A Pretraining Numerical Reasoning Model for Ordinal Constrained Question Answering on Knowledge Base

Yu Feng[1,2], **Jing Zhang**[1,2]*, **Gaole He**[2], **Wayne Xin Zhao**[3],
Lemao Liu[4], **Quan Liu**[2], **Cuiping Li**[1,2], **Hong Chen**[1,2]
[1]Key Laboratory of Data Engineering and Knowledge Engineering of Ministry of Education
[2]School of Information, Renmin University of China
[3]Gaoling School of Artificial Intelligence, Renmin University of China [4]Tencent AI Lab
{anniefeng,zhang-jing,hegaole,licuiping,chong}@ruc.edu.cn
{batmanfly,lemaoliu}@gmail.com

Abstract

Knowledge Base Question Answering (KBQA) is to answer natural language questions posed over knowledge bases (KBs). This paper targets at empowering the IR-based KBQA models with the ability of numerical reasoning for answering ordinal constrained questions. A major challenge is the lack of explicit annotations about numerical properties. To address this challenge, we propose a pretraining numerical reasoning model consisting of NumGNN and NumTransformer, guided by explicit self-supervision signals. The two modules are pretrained to encode the magnitude and ordinal properties of numbers respectively and can serve as model-agnostic plugins for any IR-based KBQA model to enhance its numerical reasoning ability. Extensive experiments on two KBQA benchmarks verify the effectiveness of our method to enhance the numerical reasoning ability for IR-based KBQA models. Our code and datasets are available online[1].

1 Introduction

Knowledge Base Question Answering (KBQA) aims at finding answers from the existing knowledge bases (KBs) such as freebase (Bollacker et al., 2008) and DBPedia (Lehmann et al., 2015) for the given questions expressed in natural language. KBQA has emerged as an important research topic in the last few years (Sun et al., 2018, 2019; Lan and Jiang, 2020; He et al., 2021), as the logically organized entities and relations in KBs can explicitly facilitate the QA process.

Two mainstream methods including the semantic parsing based (SP-based) models (Berant et al., 2013; Bao et al., 2016; Liang et al., 2017; Lan and Jiang, 2020) and the information retrieval based (IR-based) models (Sun et al., 2018, 2019; Saxena et al., 2020; He et al., 2021) are commonly studied to solve KBQA task. The SP-based models heavily rely on the intermediate logic query parsed from the natural language question, which turns out to be the bottleneck of performance improvement (Lan et al., 2021). On the contrary, the IR-based models directly represent and rank the entities in a question-aware subgraph based on their relevance to the question. Such an end-to-end paradigm is easier to train and more fault-tolerant. However, most of the IR-based models focus on the single- or multi-hop relation tasks. To answer the example question "Which is the largest city in China?" in Figure 1, the answer "Beijing" is supposed to encode not only the magnitude of its area but also the ordinal relationship with "largest"—the ordinal determiner in the question. Existing IR-based models are not explicitly aware of the magnitude and ordinal properties of entities, making the entity representations fall short in the ability to support such numerical reasoning.

In view of the issue, this paper targets at empowering the IR-based KBQA models with the ability of numerical reasoning to address the ordinal constrained questions. Ordinal constraint is summarized as one of the most important constraints via web query analysis (Bao et al., 2016) and ordinal is also defined as the second fundamental measurement to capture data in the forms of surveys[2].

Some efforts have been made on numerical reasoning for machine reading comprehension (MRC) (Yu et al., 2018; Ran et al., 2019; Chen et al., 2020). For example, given a question and a passage from which the answer can be inferred, NumNet (Ran et al., 2019) is an end-to-end model to learn the number embeddings and the non-numerical word embeddings together, which are encoded by graph neural network (GNN) (Kipf and Welling, 2017) and BERT (Devlin et al., 2019) respectively. QDGAT (Chen et al., 2020) further

* Corresponding author.
[1]https://github.com/RUCKBReasoning/NumKBQA
[2]https://www.questionpro.com/blog/nominal-ordinal-interval-ratio/

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 1852–1861
November 7–11, 2021.

wires the numbers and the words in a same graph and encode them together by GNN. However, most of them implicitly infer number embeddings based on the QA pairs without the explicit annotation of the magnitude and ordinal relationships of numbers. Such weak supervision signals bring difficulties to infer accurate number embeddings, which becomes more prominent when the ordinal supervision signals are rarely available in existing KBQA datasets. In fact, the three well-known KBQA benchmarks, MetaQA (Zhang et al., 2018), WebQuestionSP (WebQSP) (Yih et al., 2016) and ComplexWebQeustions (CWQ) (Talmor and Berant, 2018) only contain 0, 101 and 1821 ordinal constrained questions respectively.

To tackle the above challenge, we propose a pretraining method with additional self-supervision signals to capture two critical ingredients for ordinal constrained KBQA:

- **Relative Magnitude:** The relative magnitude between numbers, such as "$1 \prec 2 \prec 3$", is to be preserved by number embeddings[3].
- **Ordinal Relationship:** Based on the above relative magnitude, the ordinal relationship between each number and the ordinal determiner (such as "largest" in the question) is to be captured, *e.g.,* 3 in "$1 \prec 2 \prec 3$" is identified as the largest number.

Number embeddings which satisfy the above two ingredients are capable of numerical reasoning for ordinal constrained questions. To obtain such number embeddings, we propose two pretraining modules **NumGNN** and **NumTransformer**. The former one pretrains a GNN upon the constructed number graphs by a number-aware triplet loss function to preserve the relative magnitude, and the latter one pretrains a transformer upon the constructed question-aware number graphs by a number prediction loss function to capture ordinal relationships. Compared with the weak supervision signals from QA pairs, such self-supervision signals explicitly denote the numerical properties.

After pretraining, NumGNN and NumTransformer can be attached as model-agnostic plugins into any IR-based KBQA model to infer number embeddings. By fusing the number embeddings into the entity embeddings learned by the basic model, the numerical reasoning ability of the basic model is enhanced.

Finally, we evaluate our method on two benchmarks of KBQA: WebQSP and CWQ. Experimental results demonstrate that NumGNN plus NumTransformer, serving as plugins of alternative IR-based KBQA models, can achieve substantial and consistent improvement (+2.4 -14.8% in terms of accuracy) on the ordinal constrained questions.

2 Related Work

Knowledge Base Question Answering. Methods for the KBQA task can be categorized into two groups: SP-based methods and IR-based methods. A detailed survey of the task can be referred to (Lan et al., 2021; Zhang et al., 2021). SP-based methods (Berant et al., 2013; Berant and Liang, 2014; Yih et al., 2015; Bao et al., 2016; Liang et al., 2017; Lan and Jiang, 2020) learn a semantic parser to convert natural language questions into logic queries, which are able to deal with ordinal constrained questions. However, they heavily rely on intermediate logic queries, which becomes the bottleneck of performance improvement.

IR-based methods (Bordes et al., 2015; Dong et al., 2015; Miller et al., 2016; Sun et al., 2018, 2019; Saxena et al., 2020; He et al., 2021) directly retrieve answer candidates from the KBs and represent them to encode the semantic relationships with the questions. These methods are more fault-tolerant, but are unable to deal with ordinal constrained questions. This paper aims to enhance the IR-based models for numerical reasoning.

Numerical Reasoning. Numerical Reasoning has been studied for various tasks such as word embedding (Naik et al., 2019; Wallace et al., 2019), arithmetic word problems (AWP) (Wang et al., 2018; Zhang et al., 2020), and MRC (Yu et al., 2018; Ran et al., 2019; Chen et al., 2020). Word embedding and AWP are a little far from our task. Similar to KBQA, MRC also aims to answer questions, but infers the answers from passages instead of KBs. To enable numerical reasoning, NumNet (Ran et al., 2019) adopts a numerically-aware GNN to encode numbers and QDGAT (Chen et al., 2020) further extends the number graph with additional words. However, they are all end-to-end models weakly supervised by the final answers. This paper studies the explicit supervision signals about numerical

[3]We use "$1 \prec 2 \prec 3$" to denote that the magnitude between 1 and 2 is closer than that between 1 and 3 instead of "1<2<3" because the embeddings can only reflect the relative distance rather than the absolute magnitude.

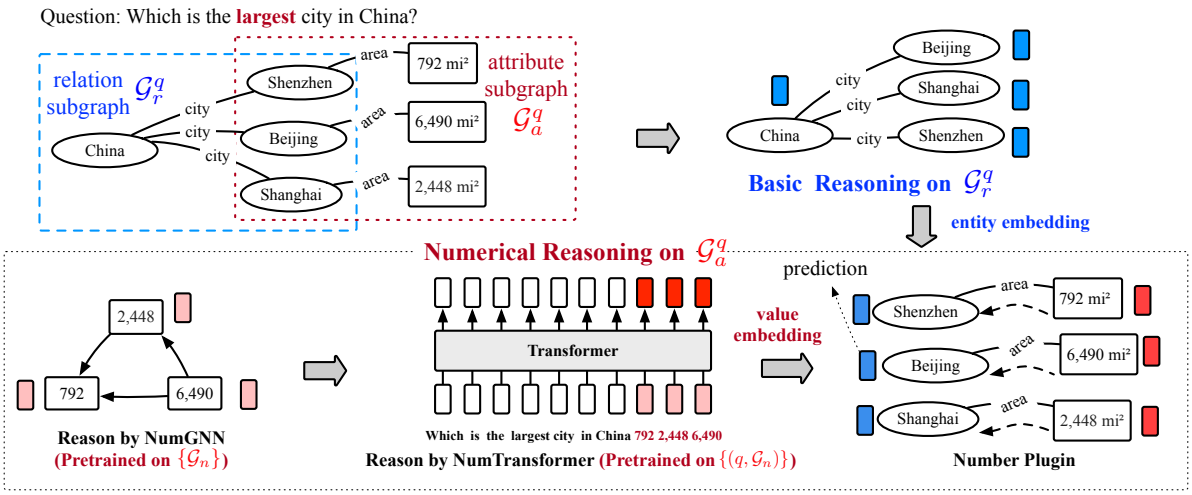


Figure 1: The whole reasoning process includes basic reasoning on the relation subgraph $\mathcal{G}_r^q$ and numerical reasoning on the attribute subgraph $\mathcal{G}_a^q$. For numerical reasoning, we first perform the pretrained NumGNN and NumTransformer to infer value embeddings and then attach them into the entity embeddings learned by the basic reasoning. The final prediction is based on the entity embeddings.

properties.

3 Method

In this section, we first introduce the ordinal constrained KBQA. Then the framework of our model is provided, followed by detailed descriptions of its components.

3.1 Problem Definition

A **Knowledge Base** $\mathcal{G}$ is the union of a relation graph $\mathcal{G}_r$ and an attribute graph $\mathcal{G}_a$, where $\mathcal{G}_r = \{(e, r, e')\}$ and $\mathcal{G}_a = \{(e, a, v)\}$ with $e(e')$, r, a, and v denoting an entity, relation, attribute, and value respectively. Their initial embeddings $\mathbf{e}^{(0)}$, $\mathbf{r}^{(0)}$, $\mathbf{v}^{(0)}$, and $\mathbf{a}^{(0)}$ are encoded by RoBERTa (Liu et al., 2019) based on their names. Attributes are divided into the numeric attributes and non-numeric attributes, where values of the former and the later ones are presented as numbers and texts respectively.

Ordinal Constrained Question (Bao et al., 2016) denotes that the answers of such question should be selected from a ranked set based on ordinal determiners in the question as ranking criteria. This paper manually defines a list of ordinal determiners: first, last, latest, earliest, largest, biggest, most, least, warmest, tallest, highest, lowest, longest, shortest, according to (Lan and Jiang, 2020).

Ordinal Constrained KBQA: Given an ordinal constrained question q, and the topic entity e_q present in q, we aim to retrieve the question-aware relation graph $\mathcal{G}_r^q$ and attribute graph $\mathcal{G}_a^q$ from $\mathcal{G}$, perform basic reasoning on $\mathcal{G}_r^q$ and numerical reasoning on $\mathcal{G}_a^q$, and then extract the answer e_t from the two graphs based on the fused entity embeddings in them.

3.2 Overall Framework

The framework of the proposed model is depicted in Figure 1. The reasoning process consists of basic reasoning on $\mathcal{G}_r^q$ and numerical reasoning on $\mathcal{G}_a^q$. The former process infers entity embeddings that can encode the semantic relationships between entities and the question, regardless of the numerical properties. Meanwhile, the latter process infers the value embeddings by the pretrained NumGNN and NumTransformer modules, and attaches them into the entity embeddings derived by the basic reasoning module to complement the relative magnitude and ordinal properties of entities.

3.3 Number Pretraining (NumGNN)

We randomly build a large amount of number graphs from the given KB, upon which we perform GNN reasoning and optimize a number-aware triplet ranking loss to preserve the relative magnitude of numbers. Henceforth, we name a graph full of number nodes as a number graph and denote it as $\mathcal{G}_n$.

Number Graph Construction. In a number graph $\mathcal{G}_n$, the nodes are composed of the values belonging to the same numerical attribute extracted from the given KB, and the edges are directed with each one pointing from a larger number to a smaller number.

In other words, v_i points at v_j if $n(v_i) > n(v_j)$, where $n(v)$ denotes the number corresponding to the node/value v. Unlike the single "greater" edge, NumNet for MRC (Ran et al., 2019) builds both the "greater" and the "lower/equal" edges between nodes. As a result, NumNet needs to additionally incorporate weights to distinguish the effect of different relations during message passing in GNN. Given this, we only keep a single "greater" relation, as it can already distinguish the magnitude of numbers and make the latter GNN model simple. We also prove this by the empirical results shown in Figure 2(c).

We randomly sample a set of numerical attributes from the whole knowledge base $\mathcal{G}$ and extract the values of the same attributes to construct the number graphs.

Number Representation. Given a number graph $\mathcal{G}_n$, we use a GNN model to learn the number embeddings of the nodes by the following steps:

(1) Node Initialization: Nodes in a number graph $\mathcal{G}_n$ are initialized by the corresponding value embeddings $\{\mathbf{v}^{(0)}\}$.

(2) Message Passing: As we intend to preserve the relative magnitude between numbers, the role a number plays in reasoning should be affected by the surrounding numbers. Specifically, We propagate messages from each number to its neighbors by the following propagation function:

$$\widetilde{\mathbf{v}}_i^{(l-1)} = \frac{1}{|\mathcal{N}_n(i)|}\Big(\sum_{v_j \in \mathcal{N}_n(i)} \alpha_j \mathrm{MLP}(\mathbf{v}_j^{(l-1)})\Big), \quad (1)$$

where $\mathbf{v}_j$ is the number embedding of v_j and $\mathcal{N}_n(i)$ is the neighbors of v_i in $\mathcal{G}_n$. MLP mentioned in this paper is the abbreviation of multi-layer perceptron. The weight α_j is formulated as:

$$\alpha_j = \sigma(\mathrm{MLP}(\mathbf{v}_j^{(l-1)})), \quad (2)$$

where σ is the sigmoid function.

(3) Node Representation Update: The information carried by the neighbors is added with the node itself to update its representation:

$$\mathbf{v}_i^{(l)} = \mathrm{ReLU}(\mathrm{MLP}(\mathbf{v}_i^{(l-1)}) + \widetilde{\mathbf{v}}_i^{(l-1)}). \quad (3)$$

The above steps (2) and (3) are repeated L times, resulting in the number embeddings $\{\mathbf{v}^{(L)}\}$ which preserve the relative magnitude between numbers. To be conveniently referred in the following sections, the entire NumGNN reasoning process (Eq. (1)-(3)) is denoted as a single function:

$$\{\mathbf{v}^{(L)}\} = \mathrm{NumGNN}(\mathcal{G}_n, \{\mathbf{v}^{(0)}\}). \quad (4)$$

Loss Function. We perform a number-aware triplet ranking loss for NumGNN optimization. Specifically, from each number graph $\mathcal{G}_n$, we randomly sample a set of triplets with each consists of three numbers and assume that the small number v_s should be closer to the medium one v_m than the big one v_b. In other words, "$v_s \prec v_m \prec v_b$" should be satisfied to reflect the relative distance between numbers rather than the absolute magnitude. We minimize the following triplet ranking loss to learn the parameters of NumGNN, *i.e.*,

$$\ell = \sum_{(v_s, v_m, v_b) \in \mathcal{T}} \max(0, \epsilon + g(\mathbf{v}_s, \mathbf{v}_m) - g(\mathbf{v}_s, \mathbf{v}_b)), \quad (5)$$

where g is cosine similarity between two numbers, $\mathcal{T}$ is the set of the sampled triplets, and ϵ is a margin separating (v_s, v_m) and (v_s, v_b).

3.4 Number Pretraining (NumTransformer)

Based on the number embeddings output by NumGNN, we need further connect the numbers to the ordinal determiners to learn the ordinal properties of numbers. For example, we aim to make the embedding of 1 in "$1 \prec 2 \prec 3$" closer to the ordinal determiner "smallest" than 2 and 3. To efficiently achieve the goal, we build a set of question-aware number graphs from the ordinal constrained QA pairs, upon which we pretrain NumTransformer and optimize a number prediction loss. Other datasets that can indicate the relationship between ordinal determiners and numbers could also be chosen for pretraining.

Question-aware Number Graph Construction. For each ordinal constrained question q, we find the most relevant numerical attribute a_t of the answer entity e_t to q via measuring the cosine similarity between the attribute embeddings $\mathbf{a}^{(0)}$ and the question embedding $\mathbf{q}^{(0)}$ encoded by RoBERTa. Then we retrieve v_t in (e_t, a_t, v_t) as the ground truth value and sample other values of the same attribute a_t as the negative instances. We restrict the negative instances within three-hops of the topic entity e_q to avoid destroying the question-specific ordinal relationship.

We construct a number graph $\mathcal{G}_n$ by the ground truth and the negative values in the same way as Section 3.3. $\mathcal{G}_n$ together with the question q compose a question-aware number graph pair $(q, \mathcal{G}_n)$.

Number Representation. Given a question-aware number graph pair $(q, \mathcal{G}_n)$, we apply NumGNN on $\mathcal{G}_n$ by Eq. (4) to output the number embeddings $\{\mathbf{v}^{(L)}\}$. Then we concatenate them with the word embeddings $\mathbf{h}_q^{(0)}$ in the question q encoded by RoBERTa as the input of a transformer to update the number embeddings, *i.e.*,

$$\{\mathbf{v}^{(L')}\} = \text{Transformer}([\mathbf{h}_q^{(0)}; \{\mathbf{v}^{(L)}\}]), \quad (6)$$

where L' is the size of the fully-connection layers in Transformer. Thanks to the multi-layer self-attention, the updated number embeddings $\mathbf{v}^{(L')}$ has fully interacted with the query words such that they can encode the ordinal semantics, *e.g.*, a "largest" or "smallest" number.

Loss Function. Since the output number embeddings of NumTransformer are conjectured to encode the ordinal properties, we can predict the ground truth number based on its output embedding, and adopt cross-entropy loss to train NumTransformer. The predictive probability of the ground truth number v_t in $\mathcal{G}_n$ is formulated as:

$$p(v_t|q, \mathcal{G}_n) = \frac{\exp \sigma(\text{MLP}(\mathbf{v}_t^{(L')}))}{\sum_j \exp \sigma(\text{MLP}(\mathbf{v}_j^{(L')}))}. \quad (7)$$

3.5 Basic Reasoning

We adopt the subgraph retrieval and reasoning scheme for basic reasoning.

Relation Subgraph Retrieval. We follow GRAFT-Net (Sun et al., 2018) to extract the neighborhood relation triplets within two hops of the topic entity e_q. To reduce the size of the triplets, we also perform the personalized PageRank (Haveliwala, 2002) to keep the most relevant entities to q. The resultant relation triplets compose the query-relevant relation subgraph $\mathcal{G}_r^q$.

Relation Subgraph Reasoning. We perform any subgraph reasoning model such as GRAFT-Net (Sun et al., 2018), EmbedKGQA (Saxena et al., 2020) and NSM (He et al., 2021) on $\mathcal{G}_r^q$ to learn the embeddings for entities in the subgraph. This model is named as BasicReason, *i.e.*,

$$\{\mathbf{e}\} = \text{BasicReason}(\mathcal{G}_r^q, \{\mathbf{e}^{(0)}\}), \quad (8)$$

where $\{\mathbf{e}^{(0)}\}$ are the initial entity embeddings.

Loss Function. The predictive probability of the answer e_t is formulated as:

$$p(e_t|q, \mathcal{G}_r^q) = \sigma(\text{MLP}(\mathbf{e}_t)). \quad (9)$$

The cross-entropy loss is optimized on both ordinal and non-ordinal constrained questions.

3.6 Numerical Reasoning

We first retrieve an attribute subgraph $\mathcal{G}_a^q$ for q, then apply the pretrained NumGNN and NumTransformer (the parameters are frozen) to infer the value embeddings in $\mathcal{G}_a^q$, which are then attached to entity embeddings in $\mathcal{G}_r^q$ for numerical reasoning. This process can be visualized as Figure 1.

Attribute Subgraph Retrieval. We extract the numerical attribute triplets for entities in $\mathcal{G}_r^q$ to compose the attribute subgraph $\mathcal{G}_a^q$. More specifically, from all the numerical attributes of the entities in $\mathcal{G}_r^q$, we extract the top-K attributes relevant to the question q by measuring the cosine similarity between the attribute embeddings and the question embedding, and add the attribute triplets $\{(h, a, v), h \in \mathcal{G}_r^q\}$ associated with these attributes into $\mathcal{G}_a^q$.

Number Embedding Inference. The values in $\mathcal{G}_a^q$ compose multiple number graphs $\{\mathcal{G}_n\}$. Each $\mathcal{G}_n$ is composed of the values of the same attributes and is built in the same way as Section 3.3. Their value embeddings are updated by the pretrained NumGNN in Eq. (4). Then they are concatenated with the question word embeddings as the input of the pretrained NumTransformer in Eq. (6) to be further updated.

Number Embedding Plugin. The updated numerical value embeddings $\{\mathbf{v}^{(L')}\}$ from $\mathcal{G}_a^q$ can be incorporated into the entity embeddings $\{\mathbf{e}\}$, which is learned by the basic reasoning module on the relation graph $\mathcal{G}_r^q$. Specifically, we aggregate the value embeddings by attentions associated with the neighborhood attributes of the i-th entity:

$$\widetilde{\mathbf{e}}_i = \sum_{j \in \mathcal{N}_a(i)} \alpha_j \text{MLP}(\mathbf{a}_j, \mathbf{v}_j), \quad (10)$$

$$\alpha_j = \text{softmax}(\mathbf{a}_j^T \mathbf{q}), \quad (11)$$

where $\mathcal{N}_a(i)$ is the i-th entity's attribute neighbors. $\mathbf{a}_j$ and $\mathbf{v}_j$ are the attribute embedding and the value embedding of the j-th neighbor respectively. The weight α_j emphasizes the question-relevant values.

Finally, we concatenate the updated entity embedding $\widetilde{\mathbf{e}}_i$ propagated from $\mathcal{G}_a^q$ with the corresponding entity embedding $\mathbf{e}_i$ in $\mathcal{G}_r^q$ to compose the ordinal-aware entity embedding:

Algorithm 1: Training Process

Input: The KB and QA pairs $\{(q, e_t)\}$.
Output: Learned parameters θ_{NG} (NumGNN), θ_{NT} (NumTransformer), θ_{BR} (BasicReason), θ_{NR} (NumericalReason), and relation/attribute embeddings $\{\mathbf{r}, \mathbf{a}\}$.

Initialize $\{\mathbf{e}, \mathbf{v}, \mathbf{r}, \mathbf{a}, \mathbf{q}\}$ by RoBERTa;
/* Pretrain NumGNN */
Build the number graphs $\{\mathcal{G}_n\}$;
Train θ_{NG} by Eq. (5);
/* Pretrain NumTransformer */
Build the query-aware number graph pairs $\{(q, \mathcal{G}_n)\}$;
Train $\{\theta_{\text{NT}}\}$ by CrossEntropy on Eq. (7);
/* Train Basic Reasoning Module */
Retrieve a relation graph $\mathcal{G}_r^q$ for each q;
Train θ_{BR} and $\mathbf{r}$ by CrossEntropy on Eq. (9) and update $\{\mathbf{e}\}$;
/* Train Numerical Reasoning Module */
Retrieve an attribute graph $\mathcal{G}_a^q$ for each q;
Build $\{(q, \mathcal{G}_n)\}$ from $\mathcal{G}_a^q$;
Apply NumGNN and NumTransformer to update $\{\mathbf{v}\}$;
Attach $\mathbf{v}$ into corresponding $\mathbf{e}$;
Train θ_{BR}, θ_{NR}, $\{\mathbf{r}\}$, and $\{\mathbf{a}\}$ by CrossEntropy on Eq. (9) and Eq. (13) jointly.

$$\mathbf{e}_i^f = \text{MLP}([\mathbf{e}_i; \widetilde{\mathbf{e}}_i]). \tag{12}$$

Note $\widetilde{\mathbf{e}}_i$ is set to $\mathbf{0}$ if the i-th entity does not have numerical attributes.

Loss Function. The predictive probability of the answer e_t is formulated as:

$$p(e_t|q, \mathcal{G}_r^q, \mathcal{G}_a^q) = \sigma(\text{MLP}(\mathbf{e}_t^f)). \tag{13}$$

The cross-entropy loss is optimized on the ordinal constrained questions.

3.7 Training & Prediction

The training process is presented in Algorithm 1. θ_{NG} of NumGNN, θ_{NT} of NumTransformer, θ_{BR} of the basic reasoning module, θ_{NR} of the numerical reasoning module, as well as the relation embeddings $\{\mathbf{r}\}$ and the attribute embeddings $\{\mathbf{a}\}$ are parameters to be optimized. Note the parameters in Eq. (8) for embedding entities are shared between θ_{BR} and θ_{NR}. The parameters in Eq. (9) for basic predicting and those from Eq. (10)-(13) for numerical predicting are separated.

For each question q, we retrieve the relation subgraph $\mathcal{G}_r^q$ and the attribute subgraph $\mathcal{G}_a^q$, predict the probability of each entity candidate in $\mathcal{G}_r^q$ by Eq. (13) if the question is ordinal constrained or by Eq. (9) otherwise.

4 Experiments

4.1 Experimental Setting

Dataset. We evaluate the proposed method on two KBQA benchmarks: WebQuestionSP (WebQSP) (Yih et al., 2016) and Complex WebQuestion 1.1 (CWQ) (Talmor and Berant, 2018). Table 1 shows the statistics of the original datasets and the retrieved subgraphs.

Evaluation Metrics. We follow GRAFT-Net to rank candidate entities[4] for each question by their predictive probabilities and then evaluate Hits@1 to reflect the accuracy of the top-1 prediction.

Baselines. We compare with three IR-based KBQA models: GRAFT-Net (Sun et al., 2018), EmbedKGQA (Saxena et al., 2020) and NSM (He et al., 2021). Compared with the Vanilla GNN, GRAFT-Net and NSM incorporate questions into graph convolution. EmbedKGQA directly optimizes the triplet of (topic entity, question, answer) based on their direct embeddings. PullNet (Sun et al., 2019)—the advanced GRAFT-Net—is not evaluated due to the unreleased code.

Implementation Details. We construct a train/valid/test set of 10000/3000/4000 number graphs for NumGNN pretraining and a train/valid/test set of 500/60/80 question-aware number graphs for NumTransformer pretraining. Dataset of this scale is capable of capturing the ordinal relationships since the initial question word embeddings and the number embeddings have already been pretrained. The scale of a number graph in both NumGNN and NumTransformer is controlled within 2 to 150 nodes to balance the efficiency and the effectiveness. We unify the units of the same attribute and only compare the numbers belonging the same attribute. We extract top-K ($K = 3$) attributes relevant to q to build the attribute subgraph.

We run experiments on single Tesla V100 GPU with 32GB memory. Both number pretraining processes can be finished in 20 minutes. Take NSM as example, with our plugins, it takes around 850/76 seconds an epoch to train model on CWQ/WebQSP dataset. All the models are trained on the training set, selected on the validation set, and evaluated on the test set. Due to the scarce ordinal labels on the validation set of WebQSP (only 4 ordinal con-

[4]The candidate entities are all the entities in the relation subgraph $\mathcal{G}_r^q$.

Table 1: Data statistics. #All/Ordinal QA pairs for training, validating and testing are presented. $|\mathcal{G}_r^q|$ and $|\mathcal{G}_a^q|$ are the average number of nodes in the retrieved relation subgraph $\mathcal{G}_r^q$ and attribute subgraph $\mathcal{G}_a^q$ respectively. Coverage and coverage(O) are the coverage rate of the answers by the subgraphs over all/ordinal QA pairs respectively.

Dataset	Train	Validation	Test	$\lvert\mathcal{G}_r^q\rvert$	$\lvert\mathcal{G}_a^q\rvert$	Coverage	Coverage(O)
WebQSP	2848/58	250/4	1639/39	432	36	91.6%	97.4%
CWQ	27639/1435	3519/189	3531/197	610	52	72.3%	84.3%

Table 2: Overall performance of different methods on (all) the test instances and the (ordinal) constrained test instances (Hits@1 by %). +Num denotes the basic model is attached with the numerical reasoning module.

Model	WebQSP		CWQ	
	All	Ordinal	All	Ordinal
GRAFT-Net	66.4	28.4	36.8	19.3
GRAFT-Net+Num	**67.4**	**43.2**	**37.3**	**25.9**
EmbedKGQA	46.0	35.4	32.0	20.0
EmbedKGQA+Num	**47.6**	**45.4**	32.0	**22.4**
NSM	68.5	33.3	46.3	24.4
NSM+Num	**68.6**	**38.5**	**47.4**	**28.4**

Table 3: Ablation study of the pre-trained NumGNN and NumTransformer (Hits@1 by %). +NumGNN: only NumGNN is attached; +Num: both NumGNN and NumTransformer are attached; (Pre-trained): the attachment is pre-trained.

	WebQSP		CWQ	
	All	Ordinal	All	Ordinal
GRAFT-Net	66.4	28.4	36.8	19.3
+ NumGNN	66.4	32.7	36.9	21.6
+ NumGNN (Pre-trained)	66.5	37.8	36.9	22.3
+ Num	66.4	33.7	36.8	20.8
+ Num (Pre-trained)	**67.4**	**43.2**	**37.3**	**25.9**

strained questions), model selection is performed on WebQSP's training set instead. For GRAFT-Net, the embedding dimension is set as 200 on CWQ and 100 on WebQSP. On both datasets, the embedding dimension is set as 200 for EmbedKGQA and 50 for NSM. The default layer size L of NumGNN and L' of NumTransformer are both set as 2. The head size of attentions in NumTransformer is set as 8 if the embedding dimension is 200 and 5 if the embedding dimension is 50 or 100. The margin ϵ for the triplet ranking loss in NumGNN is set as 0.5. The learning rate is $1 * 10^{-4}$. The NumGNN is pretrained 5 epochs with batch size as 512. The NumTransformer is pretrained 15-20 epochs with batch size as 10. All the basic reasoning models adopt the same settings as the original papers. The numerical reasoning model is trained 50 epochs with the same batch size as the corresponding basic reasoning model.

4.2 Overall Performance

Table 2 presents Hits@1 of all the compared methods. The results show that any basic IR-based model, attached with the proposed numerical reasoning module, can obtain improved performance on both the whole test set and the specific ordinal constrained test set. This indicates that the proposed model can indeed capture the ordinal relationships of entities. The basic reasoning models ignore the numerical attributes and values of entities, which apparently underperform the corresponding number-enhanced models.

The performance improvement on WebQSP is more significant than that on CWQ, as the questions on CWQ are more complex, which results in many mistaken reasoned entities, based on which the ordinal constraints are hard to be satisfied.

4.3 Ablation Study

We perform the below model variants on GRAFT-Net to investigate the effect of different components:

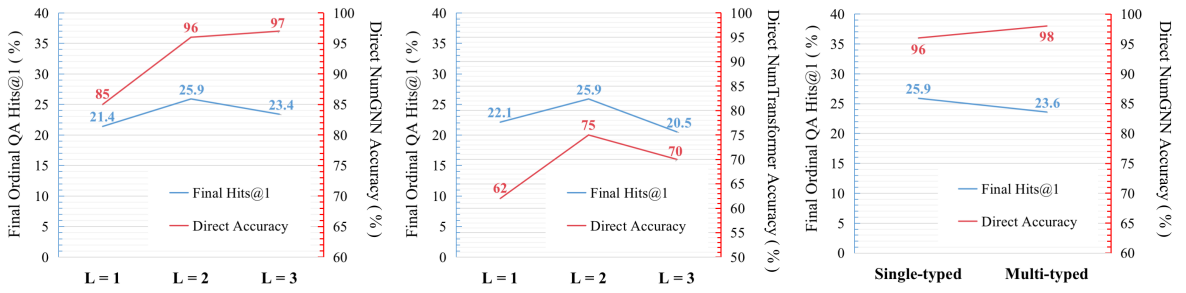
+NumGNN: with non-pretrained NumGNN, meaning that NumGNN from scratch is trained end-to-end with numerical reasoning.

+NumGNN (Pretrained): NumGNN is first pretrained and then frozen with numerical reasoning.

+Num: with non-pretrained NumGNN plus the non-pretrained NumTransformer.

+Num (Pretrained): with pretrained NumGNN and pretrained NumTransformer.

The results in Table 3 reflect 1) the effectiveness of both NumGNN and NumTransformer; 2) the positive guidance of the pretraining loss function for NumGNN and NumTransformer; 3) the inadequacy of the end-to-end QA supervision signals for NumGNN and NumTransformer.



(a) Layer L of NumGNN (b) Layer L' of NumTransformer (c) Relation Type of NumGNN

Figure 2: Direct and final evaluations of (a) NumGNN or (b) NumTransformer with different layers; (c) Direct and final evaluations of NumGNN with different relation types.

4.4 Parameter and Embedding Analysis

NumGNN Layer Size L. Figure 2(a) presents the direct performance of the pre-trained NumGNN and the final ordinal constrained QA performance with various NumGNN layers. To evaluate the direct performance, we build a set of number graphs from the given KB in the same way as Section 3.3, and evaluate whether NumGNN can explicitly preserve the relative magnitude between the largest and the smallest numbers in each graph. Specifically, we reduce the number embeddings into 1-dimensional scores, calculate the sign of score difference of the two numbers and compare it with the original sign, and finally evaluate the accuracy. Considering both the direct accuracy and the final QA performance, the 2-layer NumGNN performs the best. Because 1-layer is too shallow to distinguish the number magnitude, while 3-layer over-smooths the number embeddings.

NumTransformer Layer Size L'. Figure 2(b) presents the direct performance of NumTransformer and the final ordinal constrained QA performance with various NumTransformer layers. We evaluate the ability of predicting the right number corresponding to the ordinal determiner of the questions in the same way as Section 3.4. Considering both the direct and final evaluations, 2-layer NumTransformer performs the best, which is consistent with the layer selection of NumGNN. Due to the small amount of training data for NumTransformer, the model is sensitive to L'. If the number of L' is large, there will be too many parameters in the model and will lead to overfitting. While if the number of L' is small, the parameters are not enough to capture the features of the training data and will cause underfitting.

NumGNN Graph Relation Type. We study whether the single "greater" relation in number graphs is enough to learn the numerical properties, compared with the multi-typed relations (including "greater", "equal" and "lower" types) defined by NumNet. We perform both the direct and final QA evaluations for the single-typed and multi-typed relations. The results in Figure 2(c) show that the direct performances are almost the same but the single-typed setting outperforms the multi-typed setting in terms of Hits@1 of the final ordinal QA. Moreover, considering that the multi-typed setting demands additional weights during graph convolution to distinguish the types' effect, the single-typed relation is a better choice in our model.

Number Embeddings. We visualize the reduced 1-dimensional scores of number embeddings in an example number graph in Figure 3. We can see that the relative magnitude between almost all the numbers can be maintained. Since the scores can only reflect the relative distance rather than the absolute magnitude, the absolute sort may be kept or reversed. In fact, more than 95% number graphs in our datasets can keep the relative magnitude between the largest and the smallest numbers, more than 35% can keep all the numbers' relative magnitude, which indicates NumGNN's capacity of encoding the relative magnitude.

5 Conclusion

The paper proposes a pretraining numerical reasoning model for ordinal constrained KBQA. Via pretraining by explicit supervision signals, NumGNN and NumTransformer are capable of capturing the magnitude and ordinal properties of numbers. By attaching them as plugins into any IR-based KBQA model, the numerical reasoning ability of the model

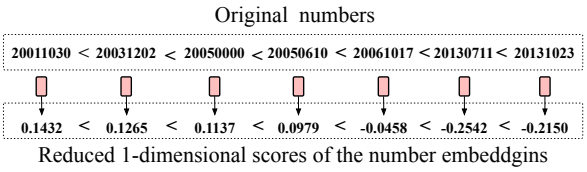


Figure 3: A case study of the number embeddings output by NumGNN . The upper and the lower bar present relative magnitude between the original numbers and between the reduced 1-dimensional scores respectively.

can be enhanced. The experimental results on two benchmarks verify the effectiveness of our model. Other types of constraints, such as multiple topic entities, type and aggregation constraints, are to be explored in the future.

Acknowledgments

This work is supported by National Natural Science Foundation of China (62076245, 62072460, 62172424); National Key Research & Develop Plan(2018YFB1004401); Beijing Natural Science Foundation (4212022); CCF-Tencent Open Fund.

References

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1415–1425.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks.

Kunlong Chen, Weidi Xu, Xingyi Cheng, Zou Xiaochuan, Yuyu Zhang, Le Song, Taifeng Wang, Yuan Qi, and Wei Chu. 2020. Question directed graph attention network for numerical reasoning over text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6759–6768.

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*, pages 4171–4186.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over Freebase with multi-column convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 260–269.

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th International Conference on World Wide Web*, page 517–526.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations*.

Yunshi Lan, Gaole He, Jing Jiang Jinhao Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and sollutions. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 969–974.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D. Forbus, and Ni Lao. 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*, pages 23–33.

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.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

Aakanksha Naik, Abhilasha Ravichander, Carolyn Rose, and Eduard Hovy. 2019. Exploring numeracy in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3374–3380.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. NumNet: Machine reading comprehension with numerical 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)*, pages 2474–2484, Hong Kong, China. Association for Computational Linguistics.

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*, pages 4498–4507.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 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*, pages 4231–4242.

Haitian Sun, Tania Bedrax Weiss, and William W. Cohen. 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*, pages 2380–2390.

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*, pages 641–651.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5306–5314.

Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. 2018. Translating a math word problem to a expression tree. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1064–1069.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1321–1331.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 201–206.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *6th International Conference on Learning Representations*.

Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. 2021. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3928–3937.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 6069–6076.