Neurocomputing 454 (2021) 280–290

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

SRGCN: Graph-based multi-hop reasoning on knowledge graphs

Zikang Wang [a,b], Linjing Li [a,b,c,*], Daniel Zeng [a,b,c]

[a] State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 101408, China
[c] Shenzhen Artificial Intelligence and Data Science Institute (Longhua), Shenzhen, China

ARTICLE INFO

Article history:
Received 11 October 2020
Revised 16 January 2021
Accepted 7 May 2021
Available online 11 May 2021
Communicated by Zidong Wang

Keywords:
Knowledge graph
Multi-hop reasoning
Graph convolutional network

ABSTRACT

Learning to infer missing links is one of the fundamental tasks in the knowledge graph. Instead of reasoning based on separate paths in the existing methods, in this paper, we propose a new model, Sequential Relational Graph Convolutional Network (SRGCN), which treats the multiple paths between an entity pair as a sequence of subgraphs. Specifically, to reason the relationship between two entities, we first construct a graph for the entities based on the knowledge graph and serialize the graph to a sequence. For each hop in the sequence, Relational Graph Convolutional Network (R-GCN) is then applied to update the embeddings of the entities. The updated embedding of the tail entity contains information of the entire graph, hence the relationship between two entities can be inferred from it. Compared to the existing approaches that deal with paths separately, SRGCN treats the graph as a whole, which can encode structural information and interactions between paths better. Experiments show that SRGCN outperforms path-based baselines on both link and fact prediction tasks. We also show that SRGCN is highly efficient in the sense that only one epoch of training is enough to achieve high accuracy, and even partial datasets can lead to competitive performance.

1. Introduction

Knowledge graphs have been shown very useful for various downstream artificial intelligence tasks like language modeling [1], question answering [2], machine reading comprehension [3], etc. However, even the most massive knowledge graph suffers from the problem of incompleteness which harms the downstream applications [4]. Therefore, knowledge graph reasoning is an important task to tackle this problem as it predicts missing links based on the known knowledge graphs.

Specifically, we situate our study in multi-hop reasoning, which aims at learning inference models to discover missing links based on multiple links that already existed in the knowledge graph. For example, for entities "Daniel Radcliffe" and "England", multi-hop reasoning first finds multiple links, "Daniel Radcliffe was born in Fulman", "Fulman contained in London", and "London contained in England", then infers a missing link "Daniel Radcliffe's nationality is England" from them.

Existing models in this field are all based on separate paths in knowledge graphs. These methods infer relationships between entities based on the relational information of paths connecting them. The Path Ranking Algorithm [5] is the first model reasoning based on paths. It finds paths and makes inferences through random walk in discrete feature spaces. RNN-Chain [6] and Combinatorial Reasoning [7] input paths into RNNs to semantically compose the relationship between entities. DeepPath [8], MINERVA [9], and AttnPath [10] frame the path learning process as reinforcement learning and reason with REINFORCE algorithm. DIVA [11] enhances model performance by finding better paths and learning more effective path-reasoners. All these models learn the relational information between the head entity and tail entity by either finding an optimal path based on reinforcement learning, or finding multiple paths first and then combining the relational information of each path learned separately. The graph structure of the paths between entities and the interaction between different paths are both ignored in the existing models.

Motivated by graph neural networks, we propose a novel approach, Sequential Relational Graph Convolutional Network (SRGCN), that treats paths as a sequence of subgraphs and makes inference based on graphs instead of paths. We define this problem as calculating the probability that a triple (h, r, t) is true. A triple (h, r, t) is true if the relation between entity h and t is r, and false

* Corresponding author at: State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China.

E-mail addresses: wangzikang2016@ia.ac.cn (Z. Wang), linjing.li@ia.ac.cn (L. Li), dajun.zeng@ia.ac.cn (D. Zeng).

https://doi.org/10.1016/j.neucom.2021.05.016
0925-2312/© 2021 Elsevier B.V. All rights reserved.

otherwise. For a triple candidate (h, r, t), we begin by constructing a graph for entities h and t. The graph consists of multiple paths between h and t, which is then serialized to a sequence of subgraphs. By updating the entity embeddings using Relational Graph Convolutional Network (R-GCN) [12] at each hop of the sequence, semantic information propagates along the sequence from h to t. The updated embedding of t contains semantic information of the entire graph and is used to determine the possibility of the triple candidate by comparing it with the embedding of relation r. Fig. 1 shows an example. In order to determine whether "Daniel Radcliffe's nationality is England" is a missing fact, we first extract multiple paths between entities "Daniel Radcliffe" and "England". The existing path-based methods are all about treating each path separately and then combining them together. Our graph-based approach treats the paths as a sequence of subgraphs and deals directly with the graph as a whole while propagating information.

To evaluate our model, we perform experiments on two benchmarks, NELL-995 [8] and FB15k-237 [13]. Our model outperforms all the path-based methods on both task link prediction and fact prediction. We further show that our model is highly efficient, only one epoch of training can yield competitive results. Also, a small fraction of training data is enough for the training for dataset FB15k-237.

Our contributions are threefold:

- We propose a new graph-based approach SRGCN for multi-hop reasoning on knowledge graphs, which models the information propagation on sequences of subgraphs instead of separate paths.
- The proposed model can scale up to large knowledge graphs and outperform existing methods.
- Our model is highly efficient as it achieves competitive results with limited training epochs and partial training datasets.

The rest of this paper is structured as follows. We will discuss the related works in Section 2. The explanation of the graph-based multi-hop reasoning model is in Section 3. Section 4 shows experimental settings and results. Finally, we conclude this paper in Section 5.

2. Related work

This section investigates the literature of three related fields: (1) knowledge graph reasoning, (2) multi-hop reasoning on knowledge graph, and (3) graph neural networks.

2.1. Knowledge graph reasoning

The aim of knowledge reasoning is to infer missing links based on existing knowledge graphs. It is an important issue that has gained increasing attention. Traditional methods use simple rules or statistical features for reasoning.

First-order relational learning algorithm is used to extract rules. For example, the reasoning module with Never-Ending Language Learning system (NELLs) [14] first learns probabilistic rules which are filtered by human, then learns new relational instances based on them. This method is limited by the size of knowledge graph.

Personalized PageRank (ProPPR) [15] is a first-order probabilistic method independent of the size of knowledge graphs. It reasons based on a personalized PageRank process over the proof constructed by Prolog's SLD resolution theorem-prover [15]. However, ProPPR cannot learn differentiable rules in a continuous space.

The first-order rules can also be combined with probabilistic graphical models. [16] is a Markov logic-based system that applys Markov logic network (MLN) [17] to knowledge graph reasoning.

Based on logical rules or statistical features, all these methods perform reasoning in discrete space and lack good generalization capability. Hence, the methods that perform reasoning in continuous space has drawn increasing attention. Hence more and more methods focus on reasoning in continuous spaces.

Embedding-based reasoning, or representational learning is one of the main methods of this kind. The embedding-based methods define a semantic relationship between embeddings of entities and relations in a continuous space, then reason based on the learned embeddings. Translation-based methods define the relationships as different translation functions between entity and tail embeddings, including TransE [18], TransH [19], TransR [20], and TransD [21]. RotatE [22] defines each relation as a rotation from the head entity to the tail entity in the complex vector space.

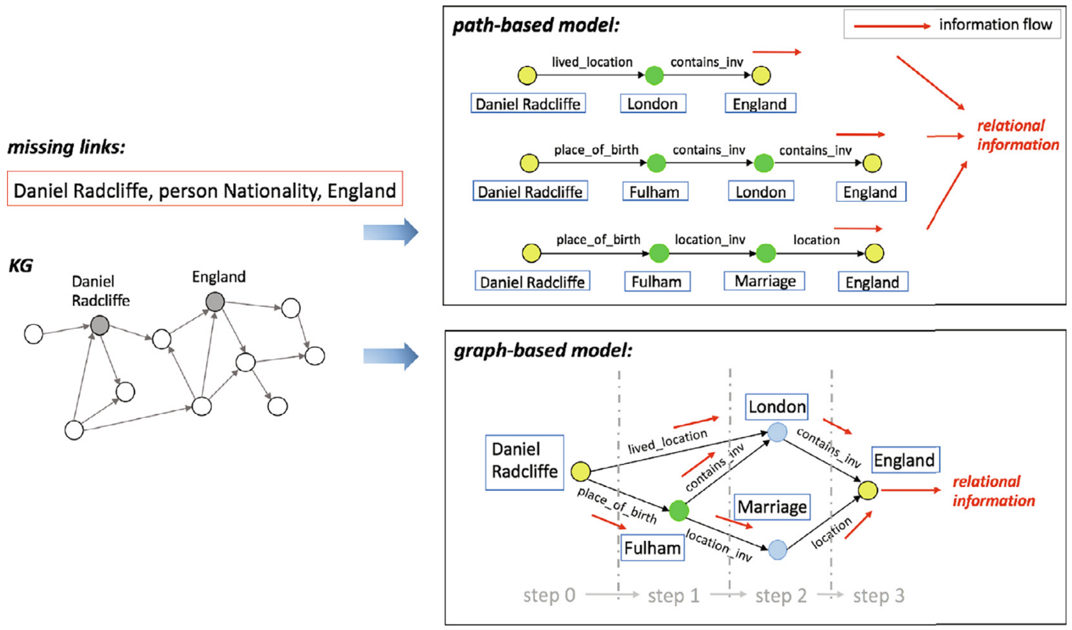


Fig. 1. To determine whether (Daniel Radcliffe, person nationality, England) is a missing link based on the known knowledge graph, existing path-based models deal with paths between "Daniel Radcliffe" and "England" separately, then combine the semantic information of each path. SRGCN serializes the paths as a sequence of subgraphs and models the information flow directly on the graph, taking into account the connections between paths. Red arrows indicate the flow of information on the graph.

ConvE [4] and ConvKB [23] apply convolutional neural network to learn embeddings.

Logic-based methods and embedding-based methods are also combined. Probabilistic Logic Neural Network (pLogicNet) combines the advantages of both kind of methods [24]. It defines the joint distribution of all possible triples by using a Markov logic network with first-order logic, and optimizes using variational EM algorithm. ExpressGNN [25] combines MLN and GNN to use graph neural networks for variational inference in MLN. It results in a good balance between the representation power and the simplicity.

2.2. Multi-hop reasoning

Embedding-based approaches though achieve very good results, they lack the interpretability. Multi-hop reasoning is proposed to address this drawback. PRA [5] made the first attempt in this direction. It finds a large set of bounded-length edge-labeled paths by performing random walk. Relational features are then combined with a logistic regression model to make inference on the relationship. It first shows that paths in the knowledge graph can be used reliably to infer missing facts. However, the generalization capability of the PRA is poor because it is performed in a discrete space.

Compositional Reasoning [7] and Chains-of-Reasoning [6] extend PRA to continuous space and bring large improvements in both performance and generalization ability. They use RNNs to encode paths into a semantic vector representing the inferred relation. Semantic information of different paths is learned separately and finally combined to make inferences.

DeepPath [8] and MINERVA [9] belong to another line of multi-hop reasoning. They frame the paths as Markov Decision Process and solve the problem using reinforcement learning. Compared to the previous models, RL-based models can control the properties of the found paths and learn long chains of reasoning. But they still deal with single paths, and the relationships between the different paths are ignored.

DIVA [11] finds that the previous methods can either be categorized as “path-finding” or “path-reasoning”, it then frames the multi-hop reasoning problem in the probabilistic graphical model to combine these two steps together as a whole.

AttnPath [10] introduces memory components by incorporating LSTM and Graph Attention Mechanism. Though it uses Graph Neural Network, it mainly uses the neighborhood information to learn the semantic features of entities. Its reasoning process is still based on reinforcement learning, which aims to find a single reasoning path. Interaction between different paths and the whole graph structure is also ignored in AttnPath. AttnPath gets state-of-the-art results on multi-hop knowledge graph reasoning.

The preceding approaches all deal with paths separately, without taking advantage of more complex patterns, like graphs. We will address this shortcoming by proposing a new model for multi-hop reasoning in Section 3, which is based on graph structure rather than individual paths.

2.3. Graph neural networks

Graph networks are a kind of neural networks that operate on graphs, which learn entity embeddings based on the structure of the graphs. They achieve good performance and interpretability effectively by the application of the expressive power of the graph structure. Graph neural networks have been used in various domains, including supervised, semi-supervised, unsupervised, and reinforcement learning settings [26].

Method in [27] firstly enables neural networks to process data in the graph structure. Many variants of the original model have been proposed in order to accommodate different types of graphs or to make use of advanced training methods. These approaches are Graph Convolutional Network (GCN) [28], Gated Graph Neural Network (GGNN) [29], FastGCN [30], GraphSAGE [31], etc.

Using GNNs to perform knowledge graph reasoning has been explored. Relation Graph Convolutional Network [12] is the first GNN model that focuses on knowledge graph. R-GCN introduces edge labeling to GCN, learns locality-sensitive embeddings, and then passes the embeddings to a decoder that predicts missing links in knowledge graphs. R-GCN shows that GNNs can be successfully applied to knowledge graphs.

Another approach on KG, SEAL [32] extracts a local subgraph around each target link and learns a function mapping the subgraph patterns to link existence. It shows that local subgraphs reserve rich information related to link existence.

Besides knowledge graphs, GNNs can also be applied to other kinds of networks, such as user-item network. Inductive Graph-based Matrix Completion (IGMC) [33] is an attempt in this direction. It trains a GNN based purely on 1-hop subgraphs around (user, item) pairs generated from the rating matrix and maps these subgraphs to their corresponding ratings [33].

The methods mentioned above generally learn the embeddings by GNNs, then reason based on either individual triples or separate paths. None of these methods uses GNNs to combine information from multiple paths while propagating information. In this paper, we use R-GCN to achieve this goal.

3. Methodology

In this section, we will describe our graph-based reasoning model in detail. Our model SRGCN is composed of three modules: graph construction module, graph serialization module, and relation inference module. For a triple candidate (h, r, t), we first construct the graph between entity h and t based on the knowledge graph, which contains multiple paths between h and t. Then we serialize the graph to a sequence of subgraphs by assigning a label for each entity in the graph, indicating the order in which they will be updated. Finally, we apply R-GCN to update the entity embeddings in order of the labels, and then determine the relationship information contained in the graph based on the updated embedding of the tail entity t. The overall architecture of the model is shown in Fig. 2.

3.1. Problem definition

We first formally define the problem of knowledge graph reasoning. A knowledge graph is a collection of triples, $\mathcal{G} = (h, r, t) \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $h, t \in \mathcal{E}$ are entities and $r \in \mathcal{R}$ is a relation. $\mathcal{E}$ and $\mathcal{R}$ are sets of entities and relations. A triple (h, r, t) indicates that head entity h and tail entity t have relation r. Knowledge graph reasoning seeks to judge whether a triple candidate (h, r, t) is true, in other words, whether the relationship between h and t is r.

In this paper, we frame the problem as calculating the probability of a triple candidate being true: $\mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$. If a triple is true, its probability of it being true should be as high as possible.

3.2. Model

Our model SRGCN consists of three modules: graph construction module, graph serialization module, and relation inference module. We describe them in the following.

3.2.1. Graph construction module

For a candidate triple (h, r, t), we first construct the graph between h and t by finding paths in the knowledge graph and combining them as one graph.

In the original knowledge graph, only one edge exists for the same relationship between two entities, its inverse edge is generally missing. For example, for edge (h, r, t), edge (t, r^{-1}, h) is missing in most cases. Hence, before finding paths, we add a reverse edge for each edge in the knowledge graph to enable turning back while finding paths. For example, we add (t, r^{-1}, h) for edge (h, r, t).

During the pathfinding process, on the one hand, it is impractical to find all paths between h and t in the knowledge graph as knowledge graphs are enormous; on the other hand, too long or too many paths may introduce noise to our model, so we limit the length and amount of paths found when constructing graphs. We prefer shorter paths to avoid too many combinations of relation and its inverse in paths, and also to decrease the loss of information while propagation. Therefore, we set a hyperparameter L that denotes path maximum length, and a hyperparameter N that denotes path maximum number. Denote the found path set as $\mathscr{S}$. For a path p, if its length, i.e., the number of relations contained in it, is less than L, and the size of $\mathscr{S}$ is less than N, it can be added into the set $\mathscr{S}$. If there is no path of length less than L, then we relax the length limit L until a path is found. We find paths by depth-first search.

Existing path-based methods will reason directly based on the found paths, however, we will further represent these paths in the form of a graph. For each entity that in $\mathscr{S}$, we record its incoming edges and the corresponding neighbor entities and denote them as $\mathscr{N}^r$ and $\mathscr{N}^e$ respectively. For example, for entity "London" in Fig. 2, we assign a neighbor list for it, (Daniel Radcliffe, Fulham), with the corresponding relation list, (lived_location, contains_inv). All the neighbors will be used to propagate information simultaneously. This step allows different paths to be combined as one graph. In this way, the graph structure can be utilized while reasoning.

Algorithm 1: Graph Construction Module

```
Input: triple $(h, r, t)$, knowledge graph $\mathscr{G}$, path length limit $L$, path number limit $N$
Output: path set $\mathscr{S}$, neighbor relations $\mathscr{N}^r$ and neighbor entities $\mathscr{N}^e$ for each entity in the graph
1. function Paths$h, t, l$
2.     return a list of paths between $h$ and $t$ with lengths no larger than $l$
3.
4. function ConstructPathSet$h, t, L_{min}, L, N$
5.     for $l = L_{min} \rightarrow L$ do
6.         for $p$ in Paths$h, t, l$ do
7.             if len($p$) $< L$ and len($\mathscr{S}$) $< N$ then
8.                 $\mathscr{S}$.add($p$)
9.     return $\mathscr{S}$
10.
11. function ConstructGraph$\mathscr{S}, h, t, \mathscr{G}, L, N$
12.     $\mathscr{S} \leftarrow \{\}$
13.     ConstructPathSet$h, t, 1, L, N$
14.     if len($\mathscr{S}$) $= 0$ then
15.         repeat
16.             $\mathscr{S} \leftarrow ConstructPathSet\mathscr{S}, h, t, L, L+1, N$
17.             $L \leftarrow L + 1$
18.         until len($\mathscr{S}$) $> 0$
19.     $\mathscr{E}_{(h,r,t)} \leftarrow$ entities in $\mathscr{S}$
20.     for $e$ in $\mathscr{E}_{(h,r,t)}$ do
21.         $\mathscr{N}^r \leftarrow$ relations on edges incoming to e
22.         $\mathscr{N}^e \leftarrow$ head entities on edges incoming to e
```

3.2.2. Graph serialization module

Next, we serialize the constructed graph to a sequence of subgraphs. In order to propagate information from h to t through the graph by updating the embeddings of entities in order of the information flow, we give each entity a label l to mark in which order it is updated.

Define the label of head entity h that equals to 0, and the distance between two entities on a path that equals to the number of relations between them on this path. Then the label l of each entity is equal to the maximum distance between it and h on all paths.

Take the entity "London" in Fig. 2 as an example, it appears in 2 paths in the graph. On path "Daniel Radcliffe, *lived_location*, London, *contains_inv*, England", its distance from the head entity "Daniel Radcliffe" is 1, as there is only one relation "*lived_location*". On another path, "Daniel Radcliffe, *place_of_birth*, Fulham, *contains_inv*, London, *contains_inv*, England", its distance from the head entity is 2, as there are two relations "*place_of_birth*" and " *contains_inv*". We label "London" as $l = 2$, for the maximum distance between "London" and the head entity is 2.

Algorithm 2: Graph Serialization Module

```
Input: triple $(h, r, t)$, entity set $\mathscr{E}_{h,r,t}$, path set $\mathscr{S}$
Output: label $l_e$ for each entity $e \in \mathscr{E}_{h,r,t}$
1. function Distance($h, e$)
2.     $d \leftarrow []$
3.     for $p$ in $\mathscr{S}$ do
4.         $d_p \leftarrow$ number of relations between $h$ and $t$ on path $p$
5.         $d.append(d_p)$
6.     return $d$
7.
8. function Serialization($\mathscr{E}_{h,r,t}$)
9.     $l_h \leftarrow 0$
10.    for $e$ in $\mathscr{E}_{h,r,t}$ AND $e$ is not $h$ do
11.        $l_e \leftarrow MAX(Distance(h, e))$
```

3.3. Relation inference module

Finally, in the relational inference module, we use R-GCN to learn the semantic information contained in graphs. The plausibility of the triple candidate is determined by the similarity between the semantic information and r.

To encode the information of the graph, we treat the graph as a sequence. Passing semantic information from head entity h to tail entity t step-by-step in an RNN-like manner, at each step, R-GCN is used to update embeddings based on the subgraph.

We first initialize the embeddings of entities as trained by TransE. Entity embeddings are then updated in the order of the labels we marked in the previous section. We start to update embeddings from entities labeled 1. If two entities have the same label, their representations are updated at the same time. At each step, the embedding of each entity is updated as a combination of its own semantics and information that precedes the entity in the graph. R-GCN is used to update embeddings:

$$v_i^{(l+1)} = \sigma\left(\sum_{r\in\mathscr{N}_i^r}\sum_{j\in\mathscr{N}_i^e}\frac{1}{c_{i,r}}W_r^{(l)}v_j^{(l)} + W_0^{(l)}v_i^0\right), \tag{1}$$

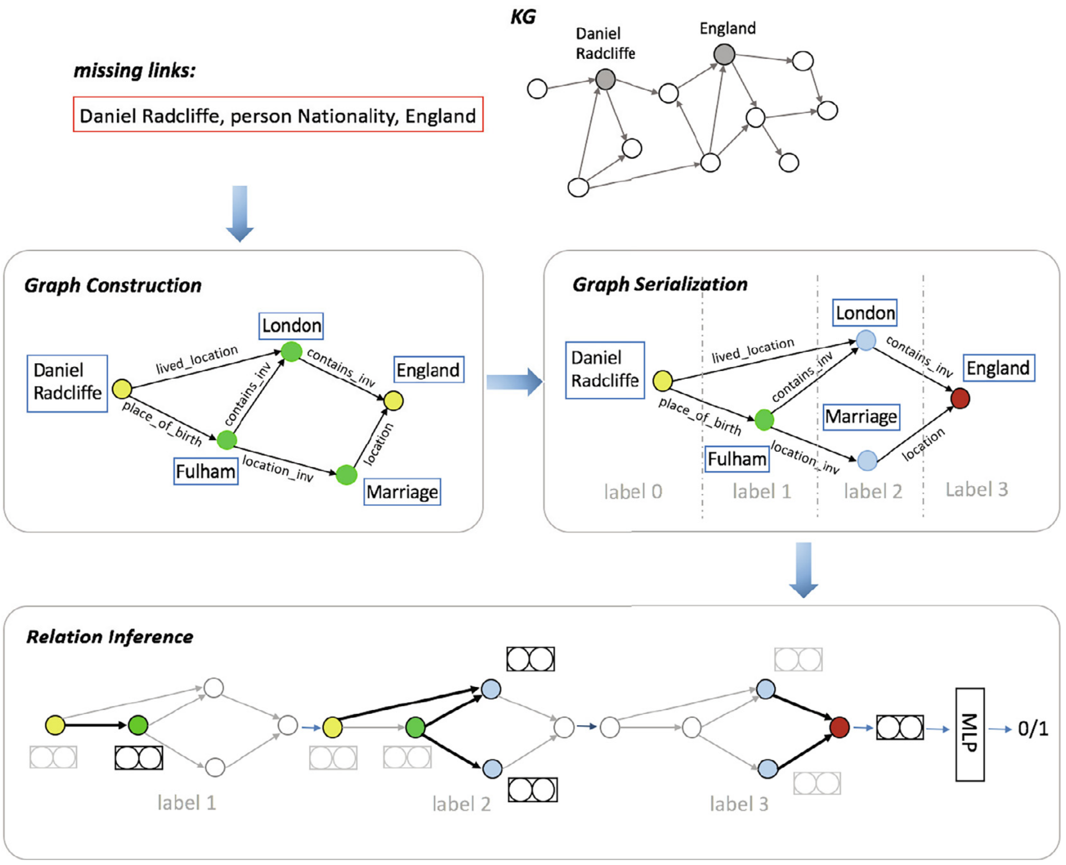


Fig. 2. The overall architecture of our model. Our model consists of three parts: the graph construction module, graph serialization module, and relation inference module. For triple candidate (Daniel Radcliffe, person Nationality, England) and the known knowledge graph, we first construct a graph between entities “Daniel Radcliffe” and “England”, then serialize it to a sequence of subgraphs, and finally, get the relational embedding of the triple. We determine whether the triple is true by feeding the relational embedding into an MLP.

where $v_i^{(l+1)}$ denotes the embedding of entity i at layer $l+1$, $\mathcal{N}_i^e$ is the set of neighbors of entity i, $\mathcal{N}_i^r$ is the corresponding relations. $c_{i,r}$ is a normalization parameter learned from the network, W_r is the relation-specific matrix of relation r, $v_j^{(l)}$ is the embedding of neighbor entity j with label l, $W_0^{(l)}$ is the weight matrix of initial embedding of entity i, v_i^0 is the initial embedding of entity i, and σ is a non-linear activation function, which is set to be ReLU in this paper.

Since each updated embedding contains information about itself and information before, we assume that the updated embedding of the tail entity contains relational information about the entire graph. Based on the relationship between it and r, we can determine whether the triple candidate is valid. Since the relation in each triple is the same in each subtask for datasets we used in the experiment, we simplify the inference into a binary classification problem by classifying the updated embedding of the tail entity to determine whether the semantic relationship of the triple is true for that subtask, i.e., whether the triple is true.

We solve this classification problem by feeding embedding v_t of tail entity t into a two-layer perceptron,

$$s_{(h,r,t)} = \mathrm{MLP}(v_t). \tag{2}$$

where $s_{(h,r,t)}$ is the possibility of the sample (h,r,t).

Then the probability that triple (h,r,t) is true is

$$p_{(h,r,t)} = \mathrm{softmax}(s_{(h,r,t)}), \tag{3}$$

where p is the possibility of triple candidate (h,r,t) being true.

Each triple in training data is labeled as 0 or 1, indicating whether it is true. We use the cross-entropy loss as loss function while training:

$$L = -(y_{(h,r,t)}\log(p_{(h,r,t)}) + (1-y_{(h,r,t)})\log(1-p_{(h,r,t)})), \tag{4}$$

where $y_{(h,r,t)}$ is the label of the triple (h,r,t) in training data, which equals 1 when the triple is true, and 0 otherwise. We train our model SRGCN using gradient descent by minimizing the loss L.

4. Experiments

In this section, we evaluate our model SRGCN, show experimental results, and make some analysis according to the results. We use two large-scale knowledge graphs as benchmarks, NELL-995 and FB15k-237. We perform two classic reasoning tasks, link prediction, and fact prediction. Link prediction is a task predicting missing entities, while fact prediction is a task determining whether a fact is true. We compare the performance of SRGCN with existing path-based methods and several classic embedding-based models. The results show that SRGCN outperforms the previous approaches. We further show that SRGCN is highly efficient, and can get competitive results even using a small fraction of training data.

4.1. Datasets

We use two widely adopted datasets to evaluate SRGCN: NELL-995 and FB15k-237.

- **NELL-995** is a sub-set of Never-Ending Language Learning datasets [34]. Proposed in [8], NELL-995 is extracted from the 995th iteration of the NELL system, which consists of the facts with top-200 relations. It contains 12 subtasks, each consisting of triples with the same relation. Subtasks include "agent belongs to organization", "athlete home stadium", "athlete plays for team", etc.
- **FB15k-237** [13] is a subset extracted from FB15K [18]. It contains 20 subtasks which have enough reasoning paths, including "birthplace", "capital of", "film country", etc.

The statistics of the two datasets in detail are shown in Table 1.

4.2. Baselines and implementation details

For baselines, we use the most widely used path-based multi-hop reasoning methods, PRA [5], RNN-Chains [6], MINERVA [9], DIVA [11], and AttnPath [10], among which AttnPath gets state-of-the-art results. We also use several classic embedding-based methods as baselines, such as TransE [18], TransR [20], etc. These embedding-based methods only use single triples for reasoning, their performance is slightly worse than path-based reasoning methods. The results of the path-based methods reported in this paper are taken directly from their original papers. For embedding-based methods, as they use different benchmarks in the original papers, we take the results reported for them in previous path-based reasoning papers.

For our model SRGCN, we use embeddings trained by TransE while initialization. One R-GCN layer is used to update embeddings. The activation function is set as ReLU. We set the length limit of paths to be 5, and the number limit of paths to be 10. We train each subtask separately and select the corresponding optimal parameters. The optimizer is Adam. For most subtasks, the dimensions of embeddings are set to be 100, and the learning rate is set to be 0.001. Training epochs are limited to 20.

4.3. Results

4.3.1. Link prediction

First proposed in [18], link prediction is a widely used evaluation criterion for knowledge graph reasoning. Given one entity and a relation, link prediction aims to rank the missing entity in a fact. In this experiment, we use the metric mean average precision (MAP) to evaluate the model performance. The overall MAP for two datasets is shown in Table 2. We further show MAP scores for different subtasks in Table 3 for NELL-995 and Table 4 for FB15k-237. The experiment shows that SRGCN outperforms all baselines on dataset FB15k-237 and gets competitive results on dataset NELL-995. The results show the effectiveness of SRGCN. Compared to R-GCN, our method shows some improvement in performance, indicating that our sequence structure is able to capture the knowledge in the graph more effectively. We notice that the performance of SRGCN on FB15k-237 is better than on NELL-995, this may be due to the fact that the length of graphs in FB15k-237 is shorter than that of NELL-995, we will discuss it in detail in the next section.

Table 1
Statistics of datasets FB15k-237 and NELL-995.

Dataset	$|\mathcal{E}|$	$|\mathcal{R}|$	#Triples	#Tasks
FB15k-237	14,505	237	310,116	20
NELL-995	75,492	200	154,213	12

Table 2
Link prediction results (MAP) on dataset NELL-995 and FB15k-237. Our model outperforms all baselines on benchmark FB15k-237, and get competitive result on NELL-995.

Models	NELL-995	FB15k-237
TransE [18]	0.737	0.532
TransR [20]	0.789	0.540
R-GCN [12]	0.795	0.597
RotatE [22]	0.823	0.638
PRA [5]	0.675	0.541
RNN-Chain [6]	0.790	0.512
DeepPath [8]	0.796	0.572
MINERVA [9]	–	0.552
DIVA [11]	**0.886**	0.598
AttnPath [10]	0.858	0.661
SRGCN	0.861	**0.702**

4.3.2. Fact prediction

Fact prediction is a task that ranks all the negative and positive samples for a particular relation [8]. The evaluation metric is also MAP. The overall results are shown in Table 5. Our graph-based model SRGCN outperforms all the baselines on this task. Note that some baselines in link prediction are not included in this task, for they only give entity ranking instead of triple tanking, such as PRA, MINERVA.

4.4. Efficiency analysis

We find that our model SRGCN is highly efficient in experiments. For most subtasks, one epoch is enough to train a competitive model. We train SRGCN for 1 to 5 epochs and show the results in Fig. 3. It shows that on FB15k-237, training more than one epoch brings little improvements to performance, On NELL-995, though more epochs are needed, we can still get competitive performance only after 4 or 5 epochs.

To compare the efficiency of SRGCN with baselines, we also train baseline MINERVA for 1 to 5 epochs. As it is not designed for task fact prediction [9], we only perform task link prediction. We choose MINERVA for comparison, partly because it's one of the state-of-the-art models, also because its code is released, allowing us to ensure that the comparison is reliable. The left sub-figure of Fig. 3 shows the performance of the link prediction of MINERVA. We can see that only 5 epochs are far from enough for it. In the source code it provided, the number of optimal training epochs is set to be 3000. Also, 500 epochs are needed for model AttnPath according to its paper. Our model SRGCN is highly efficient compared to these baselines.

After noticing that one epoch is enough for training on dataset FB15k-237 in Fig. 3, We further explore how much data is needed on this dataset while training only one epoch. We randomly select small fractions of dataset FB15k-237 and train it for one epoch. It turns out that even with a small fraction of data, SRGCN can also get competitive results.

Fig. 4 shows the model performance of two tasks on FB15k-237 when using different amounts of data. The left sub-figure shows the results of link prediction, while the right one shows the results of fact prediction. We can see that only half of the data is enough to lead to competitive results, which further shows the efficiency of SRGCN.

4.5. Influence of graph size

We define the length of a graph as equals to the label of the tail entity. It indicates how many times are needed to update entity embeddings during the information propagation. Statistics of the

Table 3
Link prediction results (MAP) on different relations of dataset NELL-995. We report the results of 10 subtasks in detail as in previous work.

Tasks	TransE	TransR	R-GCN	RotatE	PRA	DeepPath	MINERVA	SRGCN
athleteHomeStadium	0.718	0.722	0.739	0.840	0.859	0.890	0.895	0.893
athletePlaysForTeam	0.627	0.673	0.641	0.723	0.547	0.750	0.824	0.734
athletePlaysInLeague	0.773	0.912	0.897	0.925	0.841	0.960	0.970	**0.973**
athletePlaysSport	0.876	0.963	0.898	0.943	0.474	0.957	0.985	0.976
organizationHeadquarteredInCity	0.620	0.657	0.742	0.829	0.811	0.790	0.946	0.923
organizationHiredPerson	0.719	0.737	0.715	0.721	0.599	0.742	0.851	0.724
personBornInLocation	0.712	0.812	0.728	0.734	0.668	0.757	0.793	**0.829**
personLeadsOrganization	0.751	0.772	0.760	0.756	0.700	0.795	-	**0.822**
teamPlaysSport	0.761	0.814	0.803	0.821	0.791	0.738	0.846	**0.856**
worksFor	0.677	0.692	0.729	0.754	0.681	0.711	0.825	0.783

Table 4
Link prediction results (MAP) on different relations of dataset FB15k-237. We report the results of 10 subtasks in detail as in previous work.

Tasks	TransE	TransR	R-GCN	RotatE	PRA	DeepPath	AttnPath	SRGCN
teamSports	0.896	0.784	0.901	0.986	0.987	0.955	0.913	0.856
birthPlace	0.403	0.417	0.409	0.421	0.441	0.531	0.544	**0.565**
personNationality	0.641	0.720	0.744	0.818	0.846	0.823	0.846	**0.852**
filmDirector	0.386	0.399	0.392	0.450	0.349	0.441	0.437	0.372
filmWrittenBy	0.563	0.605	0.610	0.670	0.601	0.457	0.589	**0.966**
filmLanguage	0.642	0.641	0.622	0.632	0.663	0.670	0.718	0.706
tvLanguage	0.804	0.906	0.823	0.866	0.960	0.969	0.968	**0.977**
capitalOf	0.554	0.493	0.778	0.889	0.829	0.783	0.872	**0.901**
organizationFounded	0.390	0.339	0.417	0.634	0.281	0.309	0.468	**0.639**
musicianOrigin	0.361	0.379	0.405	0.460	0.426	0.514	0.526	0.524

Table 5
Fact prediction results (MAP) on dataset NELL-995 and FB15k-237. Our model outperforms all baselines on both datasets.

Models	NELL-995	FB15k-237
TransE [18]	0.383	0.277
TransH [19]	0.389	0.309
TransR [20]	0.406	0.302
TransD [21]	0.413	0.303
R-GCN [12]	0.527	0.348
RotatE [22]	0.556	0.369
DeepPath [8]	0.493	0.311
AttnPath [10]	0.693	0.379
SRGCN	**0.701**	**0.502**

length of the graphs are shown in Fig. 5 for datasets NELL-995 and dataset FB15k-237.

It shows that NELL's graphs are mostly 4 or 5 in length, while FB15k-237's graphs are mostly concentrated in 3 and 4 in length. In addition, NELL-995 has a much wider range of lengths, with a very large number of graphs with lengths larger than 10. For FB15k-237, very few subplots are with lengths greater than 5. We think that the graph length may be one of the reasons why SRGCN works better on FB15k-237 than on NELL-995, for that SRGCN may perform worse on longer sequences.

To further investigate the effect of length of graphs on the model performance, we randomly select four sub-tasks in FB15k-237 to perform link prediction and fact prediction while length equals $2, 3, 4, 5, 6$, respectively. The experimental results are shown in Fig. 6. The left figure shows the results of the link prediction, and the right one shows the results of the fact prediction. It shows that as length increases, MAP first gets higher, then gets lower on both prediction tasks. Within a certain range, the model performance improves as the length of the graph increases, we believe this is due to more knowledge being utilized as length increases. While beyond this range, the model performs worse as length increases. This can be explained by analogizing SRGCN to RNN since they propagate information along with the sequences in a similar way, they both suffer from the disadvantage of losing information along with long sequences.

We also explore how the number of paths in the graph affects the model performance. We set the number of paths as $20, 30, 50, 100, 150$, respectively, and the results are shown in Fig. 7. The left figure shows the results of link prediction, and the

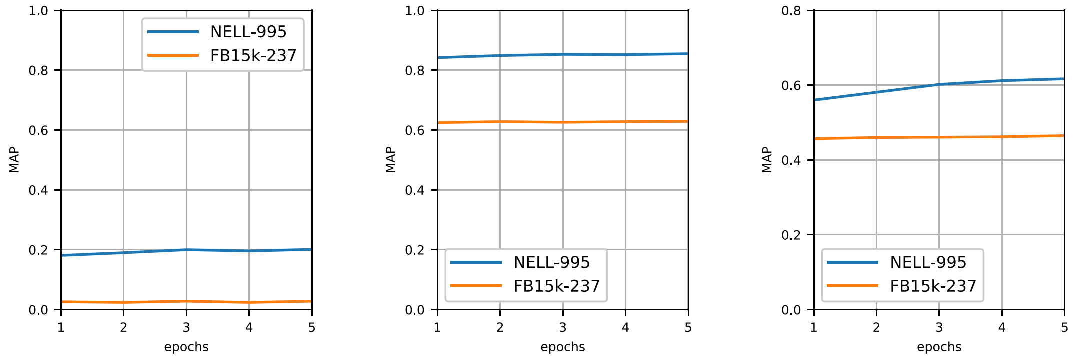


Fig. 3. The left figure shows the MAP of link prediction on baseline MINERVA while training 1 to 5 epochs. The middle figure shows the MAP of link prediction on our model. The right one shows the MAP of fact prediction on our model. It shows that for our model, only five epoch is enough to get competitive results. Especially on dataset FB15k-237, the continued training after the first epoch brings little improvements to the performance. While for baseline like MINERVA, only 5 epochs are far from enough.

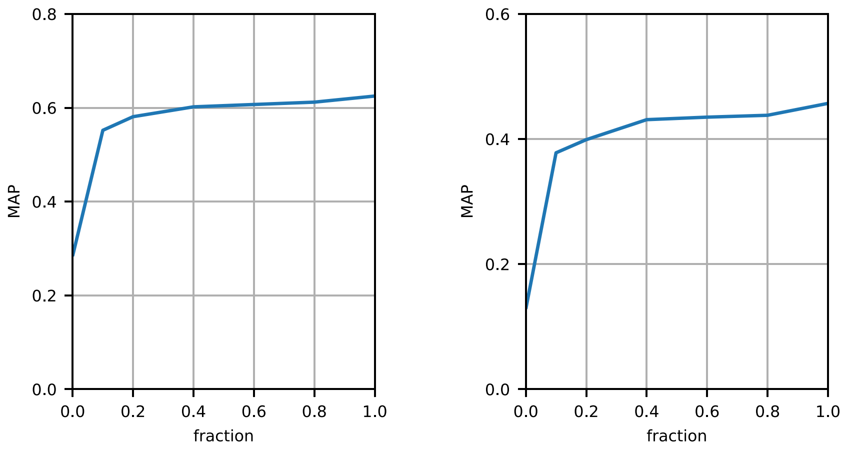


Fig. 4. MAP according to the different fraction of data used while training for only one epoch. The left sub-figure shows the link prediction results on FB15k-237, while the right one shows the fact prediction results on FB15k-237. Both figures show that we can get competitive results even use a small fraction of data, which shows the efficiency of our model. Because more than one epoch of training is required for NELL-995, this experiment is only performed on FB15k-237.

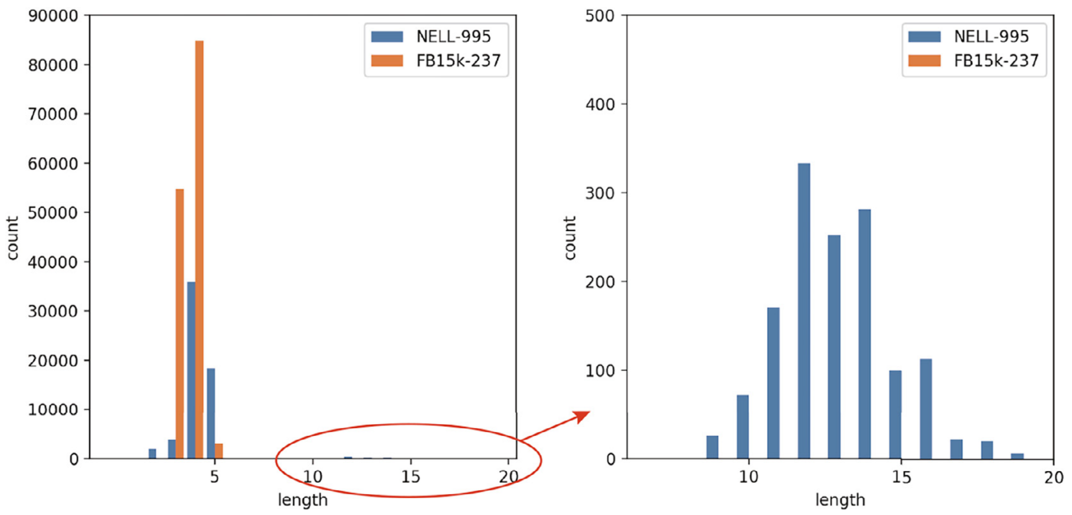


Fig. 5. Length statistics of benchmarks NELL-995 and FB15k-237. The left figure shows an overview of the length statistics, and the statistics for lengths between 6 and 20 are enlarged in the right figure. It can be seen that lengths of FB15k-237 are shorter overall and mostly 3 or 4, while NELL-995 has a larger range of lengths, with many paths longer than 10.

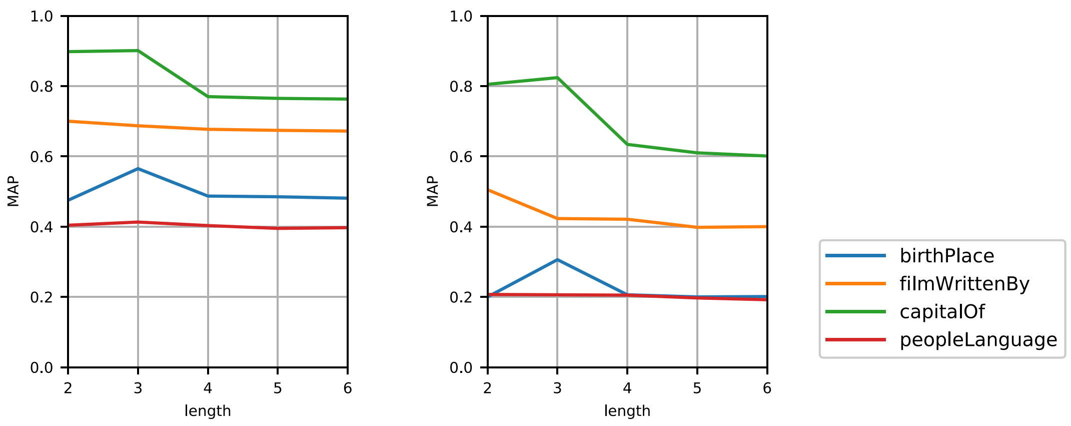


Fig. 6. We use four subtasks as examples. The left figure shows the MAP of link prediction with different lengths of graphs. The right one shows the MAP of fact prediction with different lengths. As for length increases, MAP first gets higher, then gets lower. This indicates that within a certain range, the model performance improves as the length of the graph increases, but too long will instead make the model performance worse.

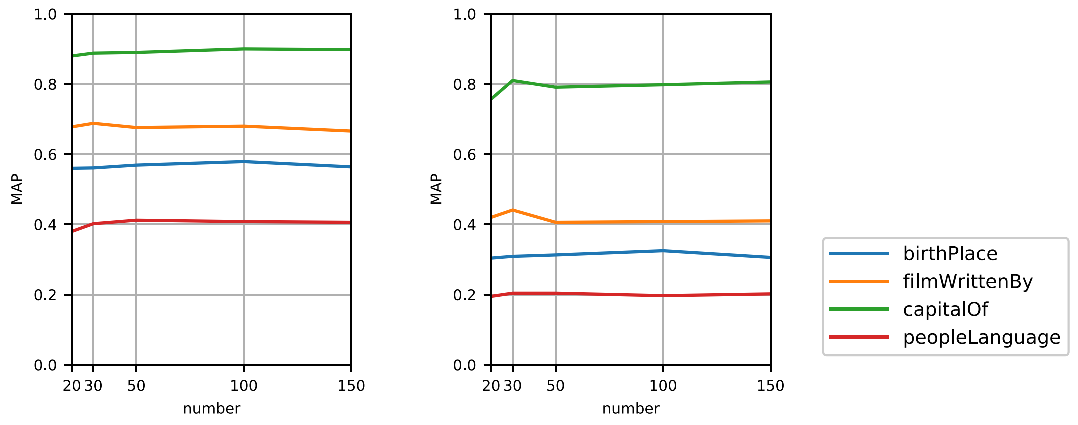
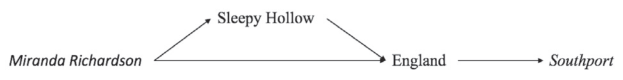
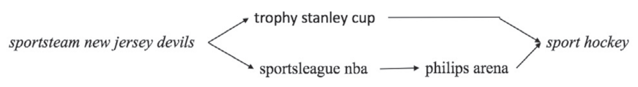
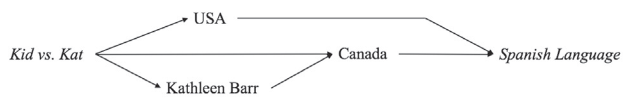
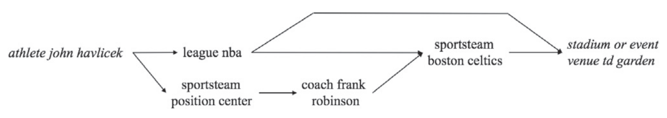


Fig. 7. Four subtasks are taken for examples. The left figure shows the MAP of link prediction with a different number of paths in graphs. The right one shows the MAP of fact prediction with a different number of paths. The two figures show that the model performance is not greatly affected by the number of paths.

Table 6
We randomly choose four relations, then show the paths used for reasoning in previous literature and the graphs used in SRGCN. In the graphs, entities aligned vertically are assigned the same label, their embeddings are updated at the same time.

Relation	Paths & Graphs used in SRGCN
birthPlace	$\mathit{Miranda\_Richardson} \xrightarrow{\text{film/performance}} \text{Sleepy\_Hollow} \xrightarrow{\text{featured\_film\_locations}} \text{England} \xrightarrow{\text{location/contains}} \mathit{Southport}$ $\mathit{Miranda\_Richardson} \xrightarrow{\text{nationality}} \text{England} \xrightarrow{\text{location/contains}} \mathit{Southport}$ 
teamPlaysSport	$\mathit{sportsteam\_new\_jersey\_devils} \xrightarrow{\text{won}} \text{trophy\_stanley\_cup} \xrightarrow{\text{championship\_of\_sport}} \mathit{sport\_hockey}$ $\mathit{sportsteam\_new\_jerse\_devils} \xrightarrow{\text{subpart\_of}} \text{sportsleague\_nba} \xrightarrow{\text{league\_stadiums}} \text{philips\_arena} \xrightarrow{\text{sport\_uses\_stadium\_inv}} \mathit{sport\_hockey}$  sportsteam new jersey devils trophy stanley cup sportsleague nba philips arena sport hockey
tvLanguage	$\mathit{Kid\_vs.\_Kat} \xrightarrow{\text{/tv/country\_of\_origin}} \text{USA} \xrightarrow{\text{countries\_spoken\_in\_inv}} \mathit{Spanish\_Language}$ $\mathit{Kid\_vs.\_Kat} \xrightarrow{\text{tv/regular\_cast}} \text{Kathleen\_Barr} \xrightarrow{\text{nationality}} \text{Canada} \xrightarrow{\text{countries\_spoken\_in\_inv}} \mathit{Spanish\_Language}$ $\mathit{Kid\_vs.\_Kat} \xrightarrow{\text{tv/country\_of\_origin)}} \text{Canada} \xrightarrow{\text{countries\_spoken\_in\_inv}} \mathit{Spanish\_Language}$  Kid vs. Kat USA Canada Kathleen Barr Spanish Language
athleteHomeStadium	$\mathit{athlete\_john\_havlicek} \xrightarrow{\text{plays\_in\_league}} \text{league\_nba} \xrightarrow{\text{league\_stadiums}} \mathit{stadium\_or\_event\_venue\_td\_garden}$ $\mathit{athlete\_john\_havlicek} \xrightarrow{\text{plays\_in\_league}} \text{league\_nba} \xrightarrow{\text{plays\_in\_league\_inv}} \text{sportsteam\_boston\_celtics} \xrightarrow{\text{team\_home\_stadiums}} \mathit{stadium\_or\_event\_venue\_td\_garden}$ $\mathit{athlete\_john\_havlicek} \xrightarrow{\text{to\_sportsteam\_position}} \text{sportsteam\_position\_center} \xrightarrow{\text{to\_sportsteam\_position\_inv}} \text{coach\_frank\_robinson} \xrightarrow{\text{plays\_for\_team}} \text{sportsteam\_boston\_celtics} \xrightarrow{\text{team\_home\_stadiums}} \mathit{stadium\_or\_event\_venue\_td\_garden}$  athlete john havlicek league nba sportsteam position center coach frank robinson sportsteam boston celtics stadium or event venue td garden

right one shows the results of fact prediction. We can see that though more paths can bring improvements to the model, the improvements are little. The model performance is not significantly affected by the number of paths.

4.6. Case study

In this section, we present a case study to show some examples directly.

As shown in Table 6, we randomly choose four relations: "birthPlace", "teamPlaysSport", "tvLanguage", and "athleteHomeStadium". Relation "birthPlace" and "tvLanguage" are in dataset FB15k-237, while "teamPlaysSport" and "athleteHomeStadium" are in dataset NELL-995. One triple is selected for each relation. For example, triple (Miranda Richardson, birthPlace, Southport) is selected for relation "birthPlace". We show the paths between the head entities and tail entities, which are used in previous literature, and also the graphs we used in our paper.

In SRGCN, multiple sequences of paths are combined into one sequence. The entities aligned vertically in Table 6 are assigned the same label, indicating their embeddings are updated simultaneously. We can see that SRGCN can combine different paths flexibly and use complicated structures, like the graph structure.

5. Conclusion

In this paper, we proposed a novel approach, SRGCN, for multi-hop reasoning on knowledge graphs. To address the shortcoming of existing models that lack effective use of graph structure, SRGCN propagates information on the sequences of subgraphs using Relational Graph Convolutional Network. Specifically, we constructed a graph for each triple candidate and serialized the graph as a sequence of subgraphs. In order to obtain the relational semantic information, we then modeled the information propagation in graphs using R-GCN. Further, we determined the credibility of the triple candidate based on the propagated information on graphs. Unlike the previous path-based models, SRGCN can utilize the graph as an effective pattern. We achieved the state-of-the-art results on link prediction and fact prediction tasks on two datasets. We also showed that SRGCN is highly efficient, it can achieve competitive results even trained by a small fraction of data on dataset FB15k-237.

In contrast to embedding-based methods, SRGCN does not rely only on separate triples, but looks for paths between head and tail entities, modelling structural information at the graph level while enhancing the interpretability of the model.

In contrast to the existing path-based approaches which learn the representations of each path separately and then combine them together, SRGCN allows the information on the paths to be combined while propagation.

Besides, existing GNN-based methods generally use GNNs to learn embeddings, then either reason based on individual triples or separate paths. SRGCN is the first to use GNNs to combine information from multiple paths while propagating information.

In the study of the future, we plan to find out how to keep all useful information even along with long sequences, like LSTM. Also, in this paper, we combine the information propagated from different entities with the same weights, which can be improved by the application of more complicated graph neural networks with attention mechanism, like Graph Attention Network [35].

CRediT authorship contribution statement

Zikang Wang: Conceptualization, Methodology, Software, Data curation, Formal analysis, Visualization, Writing - original draft. **Linjing Li:** Supervision, Conceptualization, Methodology, Validation, Writing - review & editing, Resources. **Daniel Zeng:** Conceptualization, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0103405, the National Natural Science Foundation of China under Grants 71621002, as well as the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA27030100.

References

[1] R. Logan, N.F. Liu, M.E. Peters, M. Gardner, S. Singh, Barack's wife hillary: Using knowledge graphs for fact-aware language modeling, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 5962–5971, https://doi.org/10.18653/v1/P19-1598.
[2] W. Xiong, M. Yu, S. Chang, X. Guo, W.Y. Wang, Improving question answering over incomplete KBs with knowledge-aware reader, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 4258–4264, https://doi.org/10.18653/v1/P19-1417.
[3] A. Yang, Q. Wang, J. Liu, K. Liu, Y. Lyu, H. Wu, Q. She, S. Li, Enhancing pre-trained language representations with rich knowledge for machine reading comprehension, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 2346–2357, https://doi.org/10.18653/v1/P19-1226.
[4] T. Dettmers, M. Pasquale, S. Pontus, S. Riedel, Convolutional 2d knowledge graph embeddings, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, 2018, pp. 1811–1818.
[5] N. Lao, T. Mitchell, W.W. Cohen, Random walk inference and learning in a large scale knowledge base, in: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Edinburgh, Scotland, UK, 2011, pp. 529–539.
[6] R. Das, A. Neelakantan, D. Belanger, A. McCallum, Chains of reasoning over entities, relations, and text using recurrent neural networks, in: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, Association for Computational Linguistics, Valencia, Spain, 2017, pp. 132–141. https://www.aclweb.org/anthology/E17-1013..
[7] A. Neelakantan, B. Roth, A. McCallum, Compositional vector space models for knowledge base completion, 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), Association for Computational Linguistics, Beijing, China, 2015, pp. 156–166, https://doi.org/10.3115/v1/P15-1016.
[8] W. Xiong, T. Hoang, W.Y. Wang, DeepPath: A reinforcement learning method for knowledge graph reasoning, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Copenhagen, Denmark, 2017, pp. 564–573. doi:10.18653/v1/D17-1060. https://www.aclweb.org/anthology/D17-1060..
[9] R. Das, S. Dhuliawala, M. Zaheer, L. Vilnis, I. Durugkar, A. Krishnamurthy, A. Smola, A. McCallum, Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning, in: ICLR, 2018.
[10] H. Wang, S. Li, R. Pan, M. Mao, Incorporating graph attention mechanism into knowledge graph reasoning based on deep reinforcement learning, 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), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 2623–2631, https://doi.org/10.18653/v1/D19-1264.
[11] W. Chen, W. Xiong, X. Yan, W.Y. Wang, Variational knowledge graph reasoning, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 1823–1832. doi:10.18653/v1/N18-1165. https://www.aclweb.org/anthology/N18-1165..
[12] M. Schlichtkrull, T.N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, arXiv preprint arXiv:1703.06103 (2017)..

[13] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, M. Gamon, Representing text for joint embedding of text and knowledge bases, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1499–1509, https://doi.org/10.18653/v1/D15-1174.
[14] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, J. Welling, Never-ending learning, in: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI Press, 2015, pp. 2302–2310.
[15] W.Y. Wang, K. Mazaitis, N. Lao, W.W. Cohen, Efficient inference and learning in a large knowledge base, Machine Learning 100 (1) (2015) 101–126, https://doi.org/10.1007/s10994-015-5488-x.
[16] S. Jiang, D. Lowd, D. Dou, Learning to refine an automatically extracted knowledge base using markov logic, in: 2012 IEEE 12th International Conference on Data Mining, 2012, pp. 912–917.
[17] M. Richardson, P. Domingos, Markov logic networks, Machine Learning 62 (1–2) (2006) 107–136.
[18] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 26, Curran Associates Inc, Lake Tahoe, USA, 2013, pp. 2787–2795.
[19] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, AAAI Press, Canada, 2014, pp. 1112–1119.
[20] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI Press, Austin, Texas, 2015, pp. 2181–2187.
[21] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, 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), Association for Computational Linguistics, Beijing, China, 2015, pp. 687–696.
[22] Z. Sun, Z.-H. Deng, J.-Y. Nie, J. Tang, Rotate: Knowledge graph embedding by relational rotation in complex space (2019). arXiv:1902.10197..
[23] D.Q. Nguyen, T.D. Nguyen, D.Q. Nguyen, D. Phung, A novel embedding model for knowledge base completion based on convolutional neural network, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 327–333. doi:10.18653/v1/N18-2053. https://www.aclweb.org/anthology/N18-2053..
[24] M. Qu, J. Tang, Probabilistic logic neural networks for reasoning, in: Advances in Neural Information Processing Systems 32, Curran Associates Inc, 2019, pp. 7712–7722. http://papers.nips.cc/paper/8987-probabilistic-logic-neural-networks-for-reasoning.pdf..
[25] Y. Zhang, X. Chen, Y. Yang, A. Ramamurthy, B. Li, Y. Qi, L. Song, Efficient probabilistic logic reasoning with graph neural networks (2020). arXiv:2001.11850..
[26] P.W. Battaglia, J.B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al., Relational inductive biases, deep learning, and graph networks, arXiv preprint arXiv:1806.01261 (2018)..
[27] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, Transactions on Neural Networks 20 (1) (2009) 61–80, https://doi.org/10.1109/TNN.2008.2005605.
[28] T.N. Kipf, M. Welling, Semi-Supervised Classification with Graph Convolutional Networks, in: Proceedings of the 5th International Conference on Learning Representations, ICLR '17, 2017. https://openreview.net/forum?id=SJU4ayYgl..
[29] Y. Li, R. Zemel, M. Brockschmidt, D. Tarlow, Gated graph sequence neural networks, in: Proceedings of ICLR'16, proceedings of iclr'16 Edition, 2016. https://www.microsoft.com/en-us/research/publication/gated-graph-sequence-neural-networks/..
[30] J. Chen, T. Ma, C. Xiao, Fastgcn: Fast learning with graph convolutional networks via importance sampling, 2018..
[31] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates Inc, 2017, pp. 1024–1034.
[32] M. Zhang, Y. Chen, Link prediction based on graph neural networks, in: Advances in Neural Information Processing Systems, 2018, pp. 5165–5175.
[33] M. Zhang, Y. Chen, Inductive matrix completion based on graph neural networks (2020). arXiv:1904.12058..
[34] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr., T.M. Mitchell, Toward an architecture for never-ending language learning, in: Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence AAAI'10, Atlanta, Georgia, 2010, pp. 1306–1313.
[35] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph Attention Networks, International Conference on Learning Representations Accepted as poster (2018). https://openreview.net/forum?id=rJXMpikCZ.

Zikang Wang received the B.S. degree in computer science from Central South University, Hunan, China. She is currently working towards her PhD degree at the Institute of Automation, Chinese Academy of Sciences, China. Her research interests include knowledge graph and natural language processing.

Linjing Li received the BE degree in electrical engineering and automation and the ME degree in control theory and control engineering from the Harbin Institute of Technology, Harbin, China, and the PhD degree in computer applied technology from the Graduate University of the Chinese Academy of Sciences, Beijing, China. He is currently an associate professor with the State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include game theory, mechanism design, auction theory, and machine learning.

Daniel Zeng received the BS degree in economics and operations research from the University of Science and Technology of China, Hefei, China, and the MS and PhD degrees in industrial administration from Carnegie Mellon University. He holds a Research Fellow position at the Institute of Automation, Chinese Academy of Sciences. His research interests include intelligence and security informatics, infectious disease informatics, social computing, recommender systems, software agents, and applied operations research and game theory. He has published more than 300 peer-reviewed articles. He currently serves as the editor in chief of ACM Transactions on MIS, and President of the IEEE Intelligent Transportation Systems Society. He is a Fellow of IEEE.