Reasoning Like Human: Hierarchical Reinforcement Learning for Knowledge Graph Reasoning

Guojia Wan[1] , **Shirui Pan**[2] , **Chen Gong**[3,4] , **Chuan Zhou**[5] , **Gholamreza Haffari**[2]
[1]School of Computer Science, Institute of Artificial Intelligence, and National Engineering Research Center for Multimedia Software, Wuhan University, China
[2]Faculty of Information Technology, Monash University, Australia
[3]School of Computer Science and Engineering, Nanjing University of Science and Technology, China
[4]Department of Computing, Hong Kong Polytechnic University, Hong Kong, China
[5]Academy of Mathematics and Systems Science, Chinese Academy of Sciences, China
guojiawan@whu.edu.cn, shirui.pan@monash.edu, chen.gong@njust.edu.cn,
zhouchuan@amss.ac.cn, gholamreza.haffari@monash.edu

Abstract

Knowledge Graphs typically suffer from incompleteness. A popular approach to knowledge graph completion is to infer missing knowledge by multi-hop reasoning over the information found along other paths connecting a pair of entities. However, multi-hop reasoning is still challenging because the reasoning process usually experiences multiple semantic issue that a relation or an entity has multiple meanings. In order to deal with the situation, we propose a novel Hierarchical Reinforcement Learning framework to learn chains of reasoning from a Knowledge Graph automatically. Our framework is inspired by the hierarchical structure through which a human being handles cognitionally ambiguous cases. The whole reasoning process is decomposed into a hierarchy of two-level Reinforcement Learning policies for encoding historical information and learning structured action space. As a consequence, it is more feasible and natural for dealing with the multiple semantic issue. Experimental results show that our proposed model achieves substantial improvements in ambiguous relation tasks.

1 Introduction

The development of Knowledge Graphs have increasingly impacted on many downstream AI-related applications, such as question answering (QA), information retrieval, recommendation systems, etc. However, KGs are highly incomplete, which has significantly hindered the capability of KG's application [Fang *et al.*, 2020; Ji *et al.*, 2020]. Therefore, a fundamental problem for *knowledge graph reasoning* is to predict the missing knowledge.

Recently, extensive research has emerged on learning low-dimensional representations of entities and relations for missing link prediction [Bordes *et al.*, 2013; Wang *et al.*, 2017; Nickel *et al.*, 2015]. Unfortunately, these approaches are only suitable for single-hop reasoning. Meanwhile, automated multi-hop reasoning on a large-scale KG is still a challenging problem due to the presence of diverse and ambiguous semantics. In other words, multiple semantic issue where an entity or a relation may have various meanings hinders the reasoning accuracy in a KG [Xiao *et al.*, 2016; Kertkeidkachorn and Ichise, 2017]. For instance, the visualization of TransE embedding vectors with PCA dimension reduction is presented in Figure 1. In Figure 1a, the relation 'Award_nominee' has multiple latent semantics including 'Actor-award', 'Artist-award', 'Organization', etc. The non-uniform distribution indicates the presence of the multiple semantic issue. Accordingly, semantic ambiguity is accumulated in chained reasoning process, leading to more severe performance drop.

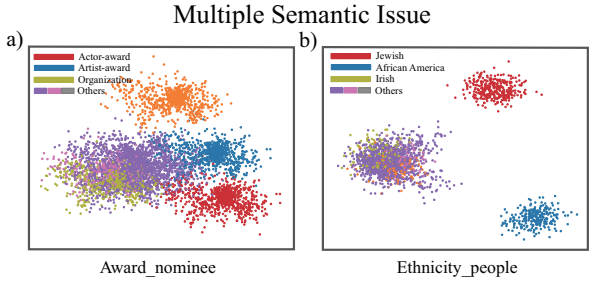

Figure 1: Visualization of TransE embeddings with PCA dimension reduction. We select two relation a) 'Award_nomee' and b) 'Ethninicy_people' from FB15K. A dot denotes a triplet, and it is depends on the difference vector between tail and head entity, i.e. $\hat{\mathbf{r}} \approx \mathbf{t} - \mathbf{h}$. The distribution of multiple clusters indicates that a relation may have multiple unknown semantics.

On the contrary, a human being can easily deal with this situation. Joshi *et al.* [2013] found that structured multiple cognitive sub-processes drive the disambiguation. One important hallmark of human cognition is that one tends to process information hierarchically [Purcell and Kiani, 2016], which divides ongoing behavior into discrete tasks that is comprised of sub-task sequences built of simple actions. Additionally, recent research [Purcell and Kiani, 2016; Sarafyazd and Jazayeri, 2019] in neuroscience and cognitive

science revealed that a human being resolves the ambiguous information of casual inference by hierarchical reasoning.

Inspired by the above observations, we formulate the mechanism of hierarchical reasoning process as Hierarchical Reinforcement Learning (HRL). HRL works on decomposing the entire problem into sub-problems, i.e, HRL splits each action into sub-actions. Some previous works have shown that not only it tackles the dimensionality curse problem [Barto and Mahadevan, 2003], but it also successfully models hierarchical decision making in robotic systems [Colin *et al.*, 2016]. By learning each sub-action of multi-hop reasoning, the agent can also learn the latent semantics of a relation through chains of reasoning. Therefore, the HRL formulation enables training an agent wtih high expressive policy networks to address the multiple semantic issue.

More recently, several RL-based KGR models have emerged as promising approaches to infer paths linking two entities in a KG [Xiong *et al.*, 2017; Das *et al.*, 2018; Shen *et al.*, 2018; Das *et al.*, 2018]. However, these approaches simply model every action in a uniform decision space. Less consideration has been given to the investigation of the hierarchical structure of knowledge reasoning process. In particular, these methods exhibit performance decrease in the tasks where multiple semantic issue exists.

In this paper, we develop a novel Hierarchical Reinforcement Learning framework, Reasoning Like Human (RLH), to imitate the thought pattern of human for applying multi-hop reasoning on knowledge graphs. By emulating hierarchical decision making, our model enables to learn chains of reasoning paths over a KG automatically. To be specific, HRL decomposes each step of reasoning into a high-level policy for encoding historical information and a low-level policy for learning to identify relation clusters. In the high-level policy, the agent trained by our model allows to encode and transfer historical information by a Gated recurrent unit (GRU). In the low-level policy, the agent follows the paradigm of hierarchical decision making, learning the concepts of relation clusters at different level of granularity. Lastly, we design a joint training method for effectively optimizing the parameters of our model. Our contributions are summarized as flows:

- We address the multiple semantic issue where a relation in knowledge graph has different meanings on multi-hop knowledge graph reasoning, which is an essential but rarely studied problem.
- We propose a novel Hierarchical Reinforcement Learning framework, Reasoning Like Human (RLH), to deal with the multiple semantic issue. The proposed model consists of a high-level policy and a low-level policy, decomposing the macro-actions into simpler sub-tasks, leading to learn the latent semantics of each relation.
- We conduct extensive experiments on three knowledge graph completion benchmarks. The results show that our model achieves competitive performance. Most importantly, our model significantly outperforms other approaches on the queries suffering from more multiple semantic issue.

2 Related Work

Knowledge Graph Reasoning. Recent developments in the field of KG have led to a renewed interest in knowledge graph reasoning. From its early days, the focus of knowledge graph reasoning has been on building systems based on symbolic logical rules [McCarthy, 1960; Quinlan, 1990]. Rule-based approaches are accurate, but suffer from poor generalization and huge complexity. Recently, *knowledge graph embedding* approaches largely superseded them [Wang *et al.*, 2017]. These methods learn topological connection information and associate entities and relations into low dimensional continuous vector spaces [Bordes *et al.*, 2013; Yih *et al.*, 2011; Dettmers *et al.*, 2018; Ye *et al.*, 2018; Nickel *et al.*, 2015]. Then, a score function or an decoder is defined to rank the target query objects with only single hop reasoning, which is a black-box system that lacks interpretability for users.

Multi-Hop Reasoning. Due to the limitations of interpretability, researchers have recently proposed multi-hop path-based approaches, such as random walks [Lao *et al.*, 2011] through a sequence of reasoning chain, further improving performance in knowledge graph completion (KGC) tasks. Unfortunately, the approaches are still computationally expensive to access the entire graph in memory. Neural LP [Yang *et al.*, 2017] is proposed to learn logical rules that can be trained in a end-to-end framework with gradient-based learning. It introduces a differential rule learning system using operators defined in TensorLog [Cohen, 2016]. Although differentiable memory allows end-to-end training, it costs expensive computation resources due to accessing the entire memory.

Deep Reinforcement Learning Reasoning. Recently, deep reinforcement learning has achieved great success in many artificial intelligence problems [Silver *et al.*, 2016]. RL shows great potential to model reasoning systems on a KG. DeepPath [Xiong *et al.*, 2017] is the first RL-based multi-hop reasoning approach for KGR. Das *et al.* [2018] further improves DeepPath by 1) avoiding pre-trained information, 2) encoding historical information using LSTM. Then, Shen *et al.* [2018] adopts Monte Carlo Tree Search (MCTS) to deal with the issue of sparse rewards to improve the efficiency of RL reasoning. However, the previous multi-hop reasoning approaches rarely consider the hierarchical structure of action space.

3 Definitions and Notations

The notation table is shown in Table 1. Then, several key definitions are given as follows.

Definition 1 (Knowledge Graph)**.** *A Knowledge Graph is a directed graph $G = (\mathcal{E}, \mathcal{R}, U)$, where $\mathcal{E}$ is a set of entities, $\mathcal{R}$ is a set of relations, and U is a set of edges. $e \in \mathcal{E}$ is an entity. $r \in \mathcal{R}$ is a relation. $u \in U$ is an edge (e_o, r, e_t) that points the head entity h to the tail entity t.*

Definition 2 (Knowledge Graph Reasoning)**.** *Given a query among three cases $(h, r, ?)$, $(?, r, t)$, $(h, ?, t)$, Knowledge Graph Reasoning aims to predict the missing element of $?$*

Symbol	Meaning	Symbol	Meaning
$\mathcal{E}$	Entity set	e	Entity
$\mathcal{R}$	Relation set	r	Relation
e_0	Source entity	e_t	Target entity
G	KG	U	Edge set
S	State set	s	State
$\mathcal{A}$	Action set	a	Action
R	Reward	γ	Reward factor
π	High-level policy	θ	Parameters
τ	High-level trajectory	$\mathbf{h_t}$	History vector
ϵ	Low-level trajectory	μ	Low-level policy
ϕ	Transition function	Φ	The set of ϕ
c	Sub-action	σ	Sigmoid function
We denote a vector using a bold letter, e.g. $\mathbf{e}$ corresponding to e			

Table 1: Annotation table

through a k-hop reasoning path $e_1 \xrightarrow{r_1} e_2 \xrightarrow{r_2} \cdots \xrightarrow{r_k} e_{k+1}$.

Example: Given (Trump, isPresident,?), a possible 2-hop reasoning path is $Trump \xrightarrow{WorkAt} WhiteHouse \xrightarrow{LocatedIn} the\ US$.

Definition 3 (Markov Decision Process)**.** *A Markov decision process is a 4-tuple (S, A, P_a, R_a). Here S is a finite set of states, A is a finite set of actions (alternatively, A_s is the finite set of actions available from state s), $P_a(s, s') = \Pr(s_{t+1} = s'|s_t = s, a_t = a)$ is the probability that action a in state s at time t will lead to state s' at time $t + 1$, $R_a(s, s')$ is the immediate reward (or expected immediate reward) received after transitioning from state s to state s', due to the action a.*

Remark: The RL is formulated as a MDP. At each stage in the sequence stages, the agent observes an environment's state s, contained in a finite set S, and executes an action a selected from a finite, non-empty set, A_s, of admissible actions. The agent receives an immediate reward having expected value $R(s, a)$, and the state transition probabilities $P(s'|s, a)$.

Definition 4 (Hierarchical Reinforcement Learning)**.** *Hierarchical Reinforcement Learning is formulated as a semi-MDP $(S, A, P_a, R_{a,\Phi})$, where $P_a(s, s') = \Pr(s_{t+1} = s'|s_t = s, a_t = a) \prod_{i=1}^{K-1} \Pr(\phi_{i+1}|\Phi_i)$. Φ is a transition function space to describe K stages transiting inside the action a. Each ϕ is a sub-action of a. All of ϕ are relevant each other.*

Remark: HRL involves discrete-time and countable sub-actions inside each action [Barto and Mahadevan, 2003]. HRL delegates the optimization of the total problem to simpler sub-problems, in which knowledge can be transferred across problems and in which component solutions can be recombined to solve larger and more complicated problems.

4 Methodology

4.1 Overview of RLH

A schematic overview of our proposed approach is presented in Figure 2. For each query, the agent trained by our RL-based reasoning approach predicts a reasoning path from the source entity to the target entity. It observes the current state

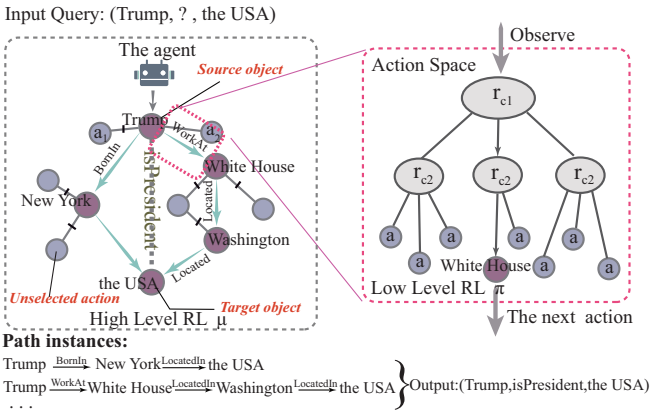

Figure 2: Schematic illustration of RLH.

and decides to move to the next entity that has the highest expectation of reaching the target entity via our well-designed hierarchical policies. The agent alternates between such observation and movement until it reaches the target entity or the maximum length. The agent's trajectory is a reasoning chain that has an attractive property that explains how the query result is obtained from the inference system.

To apply the hierarchical cognition mechanism, we propose a hierarchical policy framework $\Phi = \{\phi_1, \phi_2, \cdots, \phi_k\}$ for RL-based reasoning. The right box of Figure 2 schematically shows a 3-layers hierarchical strategy. For each interaction, the agent observes an action space $\mathcal{A}$, then it selects the most promising sub-action through Φ from hypernymy concepts to hyponymy concepts. In a knowledge graph environment, the structure of the action space is generally a hierarchical structure. As a result, the complex action space is hierarchically decomposed into sub-tasks like a search tree. Hence the multiple semantics of the relation is also decomposed into more specific representation. The details about the hierarchical policy are in Section 4.2 and 4.3.

4.2 High Level Policy for Encoding History Information

Reinforcement Learning train an agent to learn from the interactions with the environment derived from a KG through sequential exploration and exploitation. In a KG, RL is formalized with the quartuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R)$, whose elements are elaborated below.

States. The state s_i at step i is defined as a tuple (e_{i-1}, r_i, e_i, e_t), where $e_i \in \mathcal{E}$ is the current entity, e_{i-1} is the last entity, r_i denotes the relation between e_i and e_{i-1}, and e_t is the target entity. $s_i \in S$, where the state space S consists of all valid combination in $\mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{E}$. Given a pair (e_o, e_t), the starting state is represented as $('\mathrm{ST}', '\mathrm{ST}', e_0, e_t)$, where a start state indicator 'ST' was added to indicate the initial state of the agent. The final state is (e_{t-1}, r_t, e_t, e_t). Each state captures the agent's position in the KG. After taking action, the agent will move to the next state.

Actions. The action space A_{s_i} for the state $s_i = (e_i, r_{i+1}, e_{i+1}, e_t)$ is the set of outgoing edges of the current entity e_i in the KG, where $A_{s_i} = \{(r, e)|(e_i, r, e) \in G, e \notin \{e_o, e_1, \cdots, e_t\}\}$. Beginning with the source entity e_o, the

Datasets	$\|E\|$	$\|R\|$	Triplets	Tasks
FB15K-237	14505	237	310116	20
NELL995	75942	200	154231	12
WN18RR	40903	18	141422	10

Table 2: Datasets

agent uses the policy network to predict the most promising path, and it then extends its path at each step until it reaches the target entity e_t.

Transition. The transition $\mathcal{P}$ is the state transition probability used to identify the probability distribution of the next state, which is defined as a map function:$\mathcal{P} : \mathcal{S}\times\mathcal{A} \rightarrow \mathcal{S}$. The policy network encodes the current state to output a probability distribution $\mathcal{P}(s_{i+1}|s_i, a_i)$, where $a_i \in \mathcal{A}_{s_i}$. In our RL framework, the transition strategy involves selecting the action with maximum probability in $\mathcal{A}_{s_i}$.

Policy. We design a high-level policy network $\mu(s, \mathcal{A}) = P(a|s;\theta)$ based on deep learning to model the RL agent in a continuous space, where θ is the neural network parameter. Considering that the agent needs to do sequential decision making, we introduce a history vector $\mathbf{h_t}$ to keep historical information in order to better guide the agent. Given a trajectory τ at step t, the history vector is determined by the last history $\mathbf{h_{t-1}}$ and the last state $\mathbf{s_{t-1}}$, where $\mathbf{s_{t-1}} = [\mathbf{e_{t-1}}; \mathbf{r_{t-1}}; \mathbf{e_t}]$, while $\mathbf{e}, \mathbf{r} \in \mathbb{R}^d$, μ is the low-level policy,

$$\mathbf{h}_t = GRU(\mathbf{h}_{t-1}, \mathbf{s_{t-1}}). \tag{1}$$

$$a \sim \pi(a_t|s_{t-1}) = softmax(\mathbf{W}_\pi \mathbf{c}). \tag{2}$$

where c is the output sub-action (relation cluster) of the low-level policy and $\mathbf{c}$ is its vector representation. $\mathbf{W}_\pi$ is an array of $|R|$ matrices.

Rewards. Given a pair (e_o, e_t), if the agent reaches the target entity, i.e., $e_i = e_t$, the agent's trajectory is labeled as a successful finding. The reward for each hop is defined as follows:

$$R_H(\tau_i) = \begin{cases} 1 \cdot \gamma^i, & \hat{e_t} = e_t \\ 0, & \text{otherwise} \end{cases}, \tag{3}$$

4.3 Low Level Policy for Structured Action Space

The low-level policy Φ decomposes the complex action space $\mathcal{A}_s$ into structured sub-actions. The transition of these sub-actions also follows a MDP. The relations in KGs can make up relation clusters. We can build a hierarchical relation clusters by means of hierarchical-clustering relation embeddings. In this way, all states of the low-level RL are organized as a search tree. Thus the latent multiple semantics for each relation is well-expressed.

Actions. We first perform TransE [1] on a given data set and obtain the embeddings. Then, the k-means algorithm is applied to these embeddings for initializing relation clusters $C_1, C_2, C_3, \cdots, C_n$, where j-th sub-action $c_j^i \in C_i$. In this way, we enable to build hierarchical sets of relation clusters.

[1] https://github.com/thunlp/OpenKE

State. The low-level state s^l is a set containing the current valid sub-actions. For a trajectory ϵ, the starting state is the A_s. If successful, the final state is $\{a_{t+1}\}$, otherwise $\varnothing$.

Policy. When the agent observes the sub-action space under the state s_i^l, it launches the current sub-task,

$$\begin{aligned} c &\sim \mu(c_t|c_{t-1}, h_t, s_{t-1}) \\ &= Softmax(ReLU(\sigma(\mathbf{W_s}[\mathbf{h_t}; \mathbf{s}_{t-1}])[s_i^l]\mathbf{W_c}\mathbf{C_i})), \end{aligned} \tag{4}$$

where c_t is the next sub-action.

Reward. For a trajectory ϵ, a reward is obtained when the final state contains the correct action a_{next},

$$R_L(\epsilon) = \begin{cases} 1, & a_{next} \in s^l \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

4.4 Optimization and Training

In this section, we discuss how to optimize our framework. The objective function of the low-level policy network is to maximize the expectation of accumulated rewards of hierarchical decisions,

$$J^L(\theta_L) = \mathbb{E}_{\epsilon\sim p_{\theta_L}(\epsilon)}[R_L(\epsilon)], \tag{6}$$

where ϵ denotes an M-length trajectory generated from the underlying distribution $p_{\theta_L}(\epsilon)$ of the low-level policy μ, while $R(\epsilon)$ is the reward function for ϵ. The objective function of the high-level policy under the low-level policy is,

$$J^H(\theta_H) = \mathbb{E}_{\theta_L, \tau\sim p_{\theta_H}(\tau)}[R_H(\tau)], \tag{7}$$

where τ denotes an N-length trajectory generated from the underlying distribution $p_{\theta_H}(\tau)$. Similarly, $R_H(\tau)$ is the reward function for τ.

Then, we use policy gradient methods [Sutton *et al.*, 2000] with the *REINFORCE* algorithm [Williams, 1992] to optimize both high-level and low-level policies. With the likelihood ratio trick, the gradient for the policies yields:

$$\frac{\partial J^L(\theta_L)}{\partial \theta_L} \approx \frac{1}{K}\sum_{j=1}^{K}[\sum_{\tau^j, i=1}^{M} \frac{\partial}{\partial \theta_L} \log \mu(a_i^j|s_{i-1}^j, a_{i-1}^j)]. \tag{8}$$

Algorithm 1 Training Procedure

```
Initialize θ_H, θ_L
Pre-train the low-level policy μ
for 1 → episode do
    Generate a trajectory τ
    Initialize state vector h_0 ← 0
    for i ← 1 to |τ| do
        Calculate h_i ← GRU(h_{i-1}, s_{i-1})
        Generate a trajectory ϵ
        for j ← 1 to |ϵ| do
            Obtain reward R_L(ϵ_j)
            ∇J^L ← −(∂/∂θ_L) log μ(c_i^j | s_{i-1}^j, c_{i-1}^j, h_i)
        Obtain reward R_L(τ_i)
        ∇J^H ← −(∂/∂θ_H) log π(a_i^j | s_{i-1}^j, a_{i-1}^j, θ_L, h_i) γ^i
    Update θ_L,θ_L using ∇J^L,∇J^H
```

Data	Metric	TransE	ComplEX	ConvE	NeuralLP	MINRVA	RLH
NELL995	Hit@1	0.514	0.614	0.672	-	0.663	**0.692**
	Hit@3	0.678	0.784	**0.808**	-	0.773	0.768
	Hit@10	0.751	0.815	0.864	-	0.831	**0.873**
	MRR	0.456	0.652	**0.747**	-	0.725	0.723
FB15K-237	Hit@1	0.248	0.318	0.313	0.166	0.217	**0.342**
	Hit@3	0.401	0.415	0.447	0.248	0.329	**0.457**
	Hit@10	0.450	0.542	0.601	0.348	0.456	**0.648**
	MRR	0.361	0.374	0.410	0.227	0.293	**0.460**
WN18RR	Hit@1	0.289	0.319	0.402	0.376	0.413	**0.453**
	Hit@3	0.475	0.459	0.453	0.468	0.456	**0.483**
	Hit@10	0.560	0.462	**0.519**	0.657	0.513	0.516
	MRR	0.359	0.428	0.438	0.463	0.448	**0.481**

Table 3: Link Prediction on three KGC benchmarks.

Tasks	TransE	PRA	DeepPath	MINERVA	M-Walk	RLH
AthletePlaysForTeam	62.7	54.7	72.1	82.7	84.7	**86.9**
AthletePlaysInLeague	77.3	84.1	92.7	95.2	**97.8**	94.6
AthleteHomeStadium	71.8	85.9	84.6	92.8	91.9	**93.4**
AthletePlaysSport	87.6	47.4	91.7	**98.6**	98.3	97.4
TeamPlaySports	76.1	79.1	69.6	87.5	88.4	**89.1**
OrgHeadquaterCity	62.0	79.0	79.0	94.5	**95.0**	93.6
BornLocation	67.7	81.1	69.9	82.7	84.2	**87.3**
PersonLeadsOrg	75.1	68.1	75.5	**83.0**	81.2	81.4
OrgHiredPerson	71.9	66.8	79.0	83.0	88.8	**89.5**
...						
Overall	72.3	71.8	78.41	88.4	90.0	**90.2**
adjoins	68.4	41.8	69.1	71.8	64.8	**79.1**
contains	56.7	32.5	39.8	41.5	53.8	**68.4**
personNationality	44.2	42.1	52.8	**62.1**	59.1	61.9
musicianOrigin	38.2	18.5	23.7	23.8	33.8	**46.7**
capitalOf	42.5	25.8	43.8	48.9	44.0	**53.6**
filmWritten	56.1	32.1	36.5	59.1	57.2	**72.5**
filmLanguag	61.5	45.1	52.5	58.9	62.3	**68.2**
filmDirector	41.5	32.8	45.6	38.9	31.8	**48.3**
...						
Overall	45.3	31.5	39.8	42.3	43.8	**59.2**

Table 4: The MAP scores on NELL995 and FB15K-237.

$$\frac{\partial J^H(\theta_H)}{\partial \theta_H} \approx \frac{1}{K}\sum_{j=1}^{K}[\sum_{\tau^j,i=1}^{N}\frac{\partial}{\partial \theta_H}\log \pi(a_i^j|s_{i-1}^j, a_{i-1}^j, \theta_L)\gamma^i].$$

where π and μ are respectively the high-level policy and the low-level policy.

The training process is shown in Algorithm 1. In order to improve the training stability of our model, we adopt some tricks for our models. 1) We first pre-train the low-level policy network by fixing the high-level policy. Then, these two networks are jointly trained. 2)We add regularized terms into the policy networks for controlling the over-fitting, and λ is regularization. We employ *ADAM* [Kingma and Ba, 2014] to optimize the policy network. The parameters are updated every κ episodes.

5 Experiments Settings

Datasets. We conducted experiments on three datasets: 1) NELL995 released by [Xiong *et al.*, 2017] is generated from the 995-th dump of Never Ending Language Learning [Carlson *et al.*, 2010]. 2) FB15K-237, a subset of FB15K where inverse relations are removed is a knowledge base where all entities are present in Wikilinks database. 3) WN18RR is a subset of Wordnet, which provides semantic knowledge of words. Details about these datasets are shown in Table 2.

For evaluating the performance of KGR, our experiments are mainly based on two applications:

- **Relation Link Prediction**. Given a query $(h, ?, t)$, relation prediction is to predict the relation between the head entity h and the tail entity t. First, we remove all links of the ground-truth relation r in the KG. Then the agent tries to infer and walk through the KG to reach the target entity. By collecting the path between entity pairs, we feed the path features into path ranking algorithm (PRA) [Lao *et al.*, 2011], which trains a per-relation classifier to predict the existence of the ground-truth relation r in the way of binary classification. In this way, the test set containing positive and negative query pairs is evaluated, then we report the mean average precision (MAP) scores for each task.
- **Entity Link Prediction**. Given a query $(h, r, ?)$ or $(?, r, t)$, we produce a ranking of the entities by carrying out knowledge graph reasoning, and we then do a beam search with a beam width of 50 and rank entities by the probability of the trajectory reaching the correct entity. In this way, Hit@1,3,10 and mean reciprocal rank (MRR) are calculated from the ranking process, which are standard metrics for knowledge grah completion tasks [Bordes *et al.*, 2013].

Baselines. We compared some popular knowledge graph completion approaches, TransE [Bordes *et al.*, 2013], ComplEX [Yih *et al.*, 2011], ConvE [Dettmers *et al.*, 2018], and some multi-hop reasoning methods, PRA [Lao *et al.*, 2011], NeuralLP [Yang *et al.*, 2017], DeepPath [Xiong *et al.*, 2017], MINERVA [Das *et al.*, 2018].

Hyper-parameter Settings. In the training stage, the key hyper-parameter settings are as follows. The maximum length of the high-level policy l_H is fixed to 4. The maximum length of the low-level policy l_L is fixed to 2. The reward factor γ is 1.2, and the batch size κ is 100. the vector dimension d is 100. The clustering number for i-th relation cluster is $2i-1$. The regularization λ is 0.005. The network architecture parameters is optimized by grid-search in the valid set.

6 Results and Discussion

6.1 Link Prediction

Entity Link Prediction. In this task, we conduct *link prediction* experimental based on the entity prediction metric on three standard datasets, NELL995, FB15K-237, and WN18RR. Note that NeuralLP does not scale to NELL995, therefore the results are not included. The MAP results are shown in Table 3.

On NELL995 and WN18RR, our model demonstrates competitive results compared to other approaches. Meanwhile, our model significantly outperforms other baselines on FB15K-237, which obviously differs from NELL995 and WN18RR. We further analyzed the type structure of relations on FB15K-237. Then we observed that the query number of 1-to-M is larger than the M-to-1, where respectively 54% compared to 26% on FB15K-237[Bordes *et al.*, 2013]. It indicates that FB15K-237 has many multiple-semantic relations. Accordingly, multi-hop reasoning methods (PRA, NeuraLP, MINERVA) present worse performance than single-hop reasoning (TransE, ComplEX, ConvE) on FB15K-237. The results reveal that the search process of multi-hop reasoning

ID	**Query 1**:(Texas,?,Oklahoma)→(Texas,adjoins,Oklahoma)	Prediction
	MINERVA	
1	Texas $\xrightarrow{country}$ United States of America $\xrightarrow{country^{-1}}$ Oklahoma	✓
2	Texas $\xrightarrow{country}$ United States of America $\xrightarrow{country^{-1}}$ Louisiana $\xrightarrow{adjoin^{-1}}$ Oklahoma	✓
3	Texas $\xrightarrow{adjoins}$ Louisiana $\xrightarrow{country}$ United States of America $\xrightarrow{country^{-1}}$ Arkansas	×
	RLH	
4	Texas $\xrightarrow{country}$ United States of America $\xrightarrow{contains}$ Oklahoma	✓
5	Texas $\xrightarrow{first\_level}$ United States of America $\xrightarrow{contains}$ Oklahoma	✓
6	Texas $\xrightarrow{country}$ United States of America $\xrightarrow{contains}$ University of Oklahoma $\xrightarrow{state\_province\_region}$ Oklahoma	✓
	Query 2:(Ingrid Bergman,languages,?)→ (Ingrid Bergman,languages, German)	
	MINERVA	
7	Ingrid Bergman $\xrightarrow{film}$ Casablanca $\xrightarrow{language}$ German	✓
8	Ingrid Bergman $\xrightarrow{film}$ Gaslight $\xrightarrow{film^{-1}}$ Joseph Cotten $\xrightarrow{film}$ Citizen Kane	×
9	Ingrid Bergman $\xrightarrow{award}$ Academy Award for Best Actress $\xrightarrow{award^{-1}}$ Piper Laurie $\xrightarrow{nominated\_for}$ Frasier	×
	RLH	
10	Ingrid Bergman $\xrightarrow{film}$ SportsLeague mlb $\xrightarrow{MurderontheOrientExpress}$ SportsTeam chicago cubs $\xrightarrow{language}$ German	✓
11	Ingrid Bergman $\xrightarrow{film}$ Casablanca $\xrightarrow{language}$ German	✓
12	Ingrid Bergman $\xrightarrow{award}$ Academy Award for Best Supporting Actress $\xrightarrow{nominated_for}$ Roman Holiday $\xrightarrow{language}$ German	✓

Table 5: Reasoning path cases on the two queries: (Texas,?,Oklahoma), (Ingrid Bergman,languages,?).

methods is prone to be stuck in the local nodes with high-degree centrality, resulting in a fail to reach the correct entity.

Comparing to MINERVA, our approach has superior performance, indicating that the hierarchical cognition mechanism can handle multiple semantic issues.

Relation Link Prediction. In this experiment, we perform relation prediction for two datasets, NELL995, FB15K-237. The results of MAP are reported in Table 4. As the results show for NELL995, despite of failing to achieve all improvements on each task, our approach performs better on the relations with multiple semantics. For instance, 'OrgHirePerson', 'agentBelongToOrg', 'WorksFor', 'PersonLeadOrg'. Similarly to entity prediction experiments, our model outperforms other baselines in most tasks on FB15K-237.

In summary, link prediction results demonstrate that our model succeeds multi-hop reasoning on KGs with competitive performance, therefore providing interpretable reasoning path to users, and it is also favorable to handle the multiple semantic issues.

6.2 Reasoning Path Case Study

To investigate the property of reasoning paths, we present the reasoning paths found by the approaches (MINERVA, RLH) for two typical queries on FB15K-237. Table 5 shows the top-three frequency paths found by MINERVA and RLH. For the two queries, our approach all achieves the correct target entity, but MINERVA gets 4/6 hits. Note that MINERVA adds the inverse relation of each edge, i.e., for an edge $(e_1, r, e_2) \in U$, the edge (e_2, r^{-1}, e_1) to the graph. Consequently, the agent trained by MINERVA has the ability to reversely infer the reasoning paths. However, we observe that this operation significantly suppresses the reasoning process when the intermediate entities or relations are key nodes with high degree centrality, such as 'United States of America', 'country' and 'film' in Table 5. Once the agent is at this kind of nodes, the large action space hinders the decision of the policy. As a result, we observe a high-frequency occurrence of relations with high-degree centrality in path 1-3, 7-9. Contrastively, our model learns reasoning chains with wider concepts. For instance, 'country', 'first level', and 'state province region' are all in the multiple meanings of 'sub-part of'. In brief, the reasoning cases reveal that our model can learn the structural semantics of actions to identify both specific actions and sub-actions, therefore achieves improvement.

7 Conclusions

In this paper, we study multiple semantic issue where an entity or relation has multiple meanings during multi-hop knowledge graph reasoning. In order to address the problem, we consider the way that a human being handles ambiguous situations as a promising solution. We therefore design a HRL framework with a hierarchical decision making mechanism. The mechanism is implemented by a hierarchy of the high-level policy and the low-level policy. The high-level policy learns historical information. Meanwhile, the low-level policy is responsible for learning sub-actions as well as dividing each entire action space into a smaller action space. As a result, the multiple semantics of each relation can also be learned. In this way, our proposed approach can deal with multiple semantic issue in the multi-hop reasoning tasks. Experimental results show our model demonstrates competitive performance compared with existing knowledge graph completion methods. In particular, our approach achieves significant improvements on the tasks with multiple semantic issue.

Acknowledgements

This work was supported in part by NSFC under Grants 61822113, 62041105, 61872360, 61973162, STMP of Hubei Province under Grant 2019AEA170, NSF of Hubei Province under Grants 2018CFA050, FRFCU under Grant 413000092, 413000082, and 30918011319, Supercomputing Center of Wuhan University, Youth Innovation Promotion Association CAS (2017210), NSF of Jiangsu Province (BK20171430), and Australian Research Council (FT190100039).

References

[Barto and Mahadevan, 2003] Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77, 2003.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NeurIPS*, pages 2787–2795, 2013.

[Carlson *et al.*, 2010] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010.

[Cohen, 2016] William W Cohen. Tensorlog: A differentiable deductive database. *arXiv:1605.06523*, 2016.

[Colin *et al.*, 2016] Thomas R Colin, Tony Belpaeme, Angelo Cangelosi, and Nikolas Hemion. Hierarchical reinforcement learning as creative problem solving. *Robotics and Autonomous Systems*, 86:196–206, 2016.

[Das *et al.*, 2018] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *ICLR*, 2018.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, 2018.

[Fang *et al.*, 2020] Yixiang Fang, Xin Huang, Lu Qin, Ying Zhang, Wenjie Zhang, Reynold Cheng, and Xuemin Lin. A survey of community search over big graphs. *The VLDB Journal*, 29(1):353–392, 2020.

[Ji *et al.*, 2020] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. A survey on knowledge graphs: Representation, acquisition and applications. *arXiv preprint arXiv:2002.00388*, 2020.

[Joshi *et al.*, 2013] Salil Joshi, Diptesh Kanojia, and Pushpak Bhattacharyya. More than meets the eye: Study of human cognition in sense annotation. In *NAACL*, pages 733–738, 2013.

[Kertkeidkachorn and Ichise, 2017] Natthawut Kertkeidkachorn and Ryutaro Ichise. T2kg: An end-to-end system for creating knowledge graph from unstructured text. In *AAAI*, 2017.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2014.

[Lao *et al.*, 2011] Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *EMNLP*, pages 529–539. Association for Computational Linguistics, 2011.

[McCarthy, 1960] John McCarthy. *Programs with common sense*. RLE and MIT computation center, 1960.

[Nickel *et al.*, 2015] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015.

[Purcell and Kiani, 2016] Braden A Purcell and Roozbeh Kiani. Hierarchical decision processes that operate over distinct timescales underlie choice and changes in strategy. *PNAS*, 113(31):E4531–E4540, 2016.

[Quinlan, 1990] J. Ross Quinlan. Learning logical definitions from relations. *Machine learning*, 5(3):239–266, 1990.

[Sarafyazd and Jazayeri, 2019] Morteza Sarafyazd and Mehrdad Jazayeri. Hierarchical reasoning by neural circuits in the frontal cortex. *Science*, 364(6441):eaav8911, 2019.

[Shen *et al.*, 2018] Yelong Shen, Jianshu Chen, Po-Sen Huang, Yuqing Guo, and Jianfeng Gao. M-walk: Learning to walk over graphs using monte carlo tree search. In *NeurIPS*, pages 6786–6797, 2018.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[Sutton *et al.*, 2000] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NeurIPS*, pages 1057–1063, 2000.

[Wang *et al.*, 2017] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *TKDE*, 29(12):2724–2743, 2017.

[Williams, 1992] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[Xiao *et al.*, 2016] Han Xiao, Minlie Huang, and Xiaoyan Zhu. Transg: A generative model for knowledge graph embedding. In *ACL*, volume 1, pages 2316–2325, 2016.

[Xiong *et al.*, 2017] Wenhan Xiong, Thien Hoang, and William Yang Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *EMNLP*, pages 564–573, 2017.

[Yang *et al.*, 2017] Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *NeurIPS*, pages 2319–2328, 2017.

[Ye *et al.*, 2018] Mang Ye, Zheng Wang, Xiangyuan Lan, and Pong C Yuen. Visible thermal person re-identification via dual-constrained top-ranking. In *IJCAI*, volume 1, page 2, 2018.

[Yih *et al.*, 2011] Wen-tau Yih, Kristina Toutanova, John C Platt, and Christopher Meek. Learning discriminative projections for text similarity measures. In *CoNLL*, pages 247–256, 2011.