# Path Reasoning over Knowledge Graph: A Multi-agent and Reinforcement Learning Based Method

**Conference Paper** · November 2018

DOI: 10.1109/ICDMW.2018.00135

**5 authors**, including:

Zixuan Li
Chinese Academy of Sciences
**38** PUBLICATIONS **918** CITATIONS
SEE PROFILE

Saiping Guan
Chinese Academy of Sciences
**35** PUBLICATIONS **1,170** CITATIONS
SEE PROFILE

Yuanzhuo Wang
Institute of Computing Technology, Chinese Academy of Sciences
**135** PUBLICATIONS **3,348** CITATIONS
SEE PROFILE

2018 IEEE International Conference on Data Mining Workshops (ICDMW)

# Path Reasoning over Knowledge Graph: A Multi-Agent and Reinforcement Learning Based Method

Zixuan Li, Xiaolong Jin, Saiping Guan, Yuanzhuo Wang, Xueqi Cheng
*CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences*
*School of Computer and Control Engineering, University of Chinese Academy of Sciences*
*Beijing, China*
{lizixuan, guansaiping}@software.ict.ac.cn; {jinxiaolong, wangyuanzhuo, cxq}@ict.ac.cn

***Abstract*—Relation reasoning over knowledge graphs is an important research problem in the fields of knowledge engineering and artificial intelligence, because of its extensive applications (e.g., knowledge graph completion and question answering). Recently, reinforcement learning has been successfully applied to multi-hop relation reasoning (i.e., path reasoning). And, a kind of practical path reasoning, in the form of query answering (e.g., (entity, relation, ?)), has been proposed and attracted much attention. However, existing methods for such type of path reasoning focus on relation selection and underestimate the importance of entity selection during the reasoning process. To solve this problem, we propose a Multi-Agent and Reinforcement Learning based method for Path Reasoning, thus called MARLPaR, where two agents are employed to carry out relation selection and entity selection, respectively, in an iterative manner, so as to implement complex path reasoning. Experimental comparison with the state-of-the-art baselines on two benchmark datasets validates the effectiveness and merits of the proposed method.**

***Index Terms*—path reasoning; relation reasoning; knowledge graph; reinforcement learning**

## I. Introduction

There is an increasing interest in reasoning over knowledge graphs (KGs) [1], due to its applications in many tasks. In the KG construction task, relation reasoning can be used to infer implicit triples in KGs from existing ones, which improves their completeness. In the more advanced applications of KG, like question answering, relation reasoning helps find connotative answers. Recently, multi-hop relation reasoning (i.e., path reasoning) has attracted extensive attention from related communities [2]–[5]. For example, we may infer that $Paul\ Allen$ is a friend of $Bill\ Gates$ through the following reasoning path on a small KG shown in Fig. 1 : $Bill\ Gates \xrightarrow{spouse} Melinda\ Gates \xrightarrow{friend} Paul\ Allen.$

So far, quite a few methods have been proposed to cope with the path reasoning problem. The Path Ranking Algorithm (PRA) [2] is a seminal and promising method for finding inference paths in KGs. Many approaches [6]–[8] were then proposed to enhance path reasoning based on the paths found by PRA. Recently, the Reinforcement Learning (RL) framework [9] has proven effective in many KG related tasks [10], [11]. Actually, the path reasoning problem is suitable to be formulated as a RL problem where the goal is to make a sequence of decisions on choosing suitable relation edges to finally reach the correct answer. For instance, DeepPath [12] formulates the process of path finding between two entities as a Markov Decision Process (MDP) and RL is employed to maximize the expected reward. During the path finding process, DeepPath adopts a random-entity-selection strategy when facing a 1-N/N-N relation, which connects the current entity to multiple different entities via the same relation. Although this strategy exempts DeepPath from complex decision making on entity selection, it may force DeepPath to perform return operations for many times until it finds a valid target entity.

The above approaches consider only the task of fact prediction, where the model/algorithm predicts a missing relation given two entities (i.e., ($entity_1$, ?, $entity_2$)), or only evaluates the truth of a given triple (i.e., ($entity_1$, $relation$, $entity_2$)?). And they find multiple relation paths which provide evidences for a prediction. Differently, MINERVA [13] applies a similar RL based method to deal with a type of relatively difficult path reasoning task in the form of query answering (i.e., ($entity$, $relation$, ?)), which finds a valid answer entity among all entities upon the given source entity and the given query relation. MINERVA learns its policy from the source entity to the target entity on a training dataset so that for an unseen pair of source entity and query relation, it is able to walk over the graph to find the correct target entity. Note that, MINERVA just picks the most promising path from all relation paths found in limit attempts as the only evidence for a prediction.

Due to the inherent features, entity selection is more important in the query answering task than in the fact prediction task. The fact prediction task concentrates on finding logistic relation paths between two given entities and performs a path constrained search based on the found logistic paths. Then supervised learning methods are used to evaluate the quality of the paths. The known entity pairs given by the fact prediction task constrain the selection of entity so that entity selection is of little account in the path finding step. However, in the query answering task, the answer entity is unknown during the whole path reasoning process. There is no idea how far the current position is to the answer entity and the model is more likely to shift from the correct path. Besides, logistic

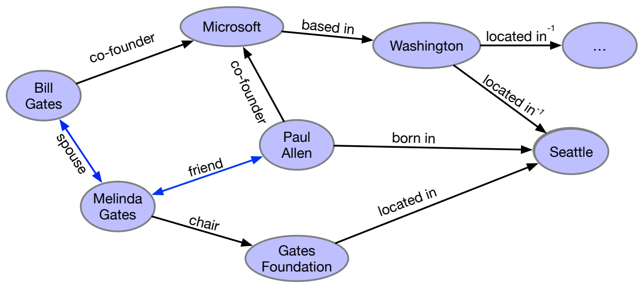

Fig. 1. A small illustrative knowledge graph.

relation path finding and answer entity reasoning are carried out jointly in the query answering task. More specifically, it finds the most suitable path to the answer entity with limited attempts, which requests the model to have relatively stronger navigation ability. Previous works (e.g. MINERVA) consider relation selection and entity selection jointly as relation-entity pair selection via a single policy network under the supervision of the query relation at each step, which underestimate the importance of entity selection when a 1-N/N-N relation appears during the path selection process. However, even the relation path is correct, the different entities filled in the path may lead to different entities. For example, to predict the airport of New York, the following paths $NY(New\ York) \xrightarrow{locatedIn^{-1}} NYC(New\ York\ City) \xrightarrow{locatedIn^{-1}} JFK(JFK\ Airport)$ and $NY \xrightarrow{locatedIn^{-1}} NYC \xrightarrow{locatedIn^{-1}} Yankee\ Stadium$ share the same relation path but only the former one is correct.

To solve the above problems, we highlight the process of entity selection as a separate module in this study. In this way, the path searching procedure is split into two sub-steps. The first sub-step is logistic relation path finding, which aims at finding the most suitable relation at each step to make up the query relation in semantics. In the query answering task, the query relation is the only information to guide the model to the target answer. Therefore, the model also decides whether it should stop at this sub-step. The second sub-step is answer entity reasoning, which aims at choosing the most suitable entity at each step until arriving at the valid answer entity. Since the answer entity is unknown, we make the selected entity semantically similar to those entities already in the reasoning path, which alleviates the shift from the correct path approaching to the answer. Correspondingly, we design a Multi-Agent and Reinforcement Learning based method for Path Reasoning, thus called MARLPaR, which contains two agents, i.e., a relation selection agent and an entity selection agent, to address relation path finding and answer entity reasoning simultaneously. More specifically, the former agent is employed to select a certain relation at each step, which may connect the current entity to a tail entity or a tail entity set. The latter one is responsible for finding a proper entity from the tail entity set, if the selected relation is a 1-N/N-N one. Particularly, these two agents collaborate with each other at each step and decide the valid action space for each other.

In general, the contributions of this paper can be summarized as follows:

- We present a Multi-Agent and Reinforcement Learning based method for Path Reasoning over knowledge graph, called MARLPaR, where two agents are employed to carry out relation selection and entity selection iteratively so as to find exactly correct relation path to the answer entity;
- Through experimental comparison with the state-of-the-art baselines on two benchmark datasets, we validate the effectiveness of the proposed method.

## II. Related Works

### A. Knowledge Graph Embedding Based Approaches

Embedding based approaches to model multi-relation data from KG have drawn increasing attention for recent years [14]–[19] and achieve improvement in many tasks [20]. Existing translation based methods like TransE [14] and its extensions [21]–[23] follow the idea that the embedding of a head entity can be translated to that of a tail entity by adding it with the embedding of the corresponding relation. Semantic matching models, like RESCAL [24] represent each relation as a matrix, which models pairwise interactions between latent factors. DistMult [25] simplifies RESCAL by restricting the relation representations to be diagonal matrices. ComplEx [16] improves DistMult by extending the representation space to a complex field in order to model asymmetric relations relatively better. ConvE [26] introduces a multi-layer convolutional network model for link prediction and gets the state-of-the-art results. Although these embedding based methods achieve impressive results in the link prediction task, most of them address this task on a latent space. Thus, they are usually unexplainable and cannot model the multi-hop relation path.

### B. Multi-Hop Link Prediction Approaches

Multi-hop link prediction approaches [2], [6], [7] address the problems of embedding based approaches mentioned above. The Path Ranking Algorithm (PRA) [2] performs depth-first search over large KGs to find relation paths between a given entity pair; Then, it adopts a supervised learning method to pick up the most promising relation paths; Finally, it uses the selected relation paths as features to predict the direct one-hop relation between the given entity pair. Later on, some research studies [6], [7] improve PRA by representing textual relations as continuous vectors and compute feature similarity in vector space. Other approaches [3], [8] employ PRA to find logistic relation paths and then a recurrent neural networks model is used to model relational paths. Recently, the RL framework is involved in multi-hop reasoning. For example, DeepPath [12] and MINERVA [13] view the whole path reasoning process as a Markov Decision Process (MDP) and get the state-of-the-art results.

*1) DeepPath:* DeepPath is the first one to use the RL based method to find relation paths between two entities in KBs. DeepPath walks from the source entity, chooses a relation and translates to any entity in the tail entity set of the relation.

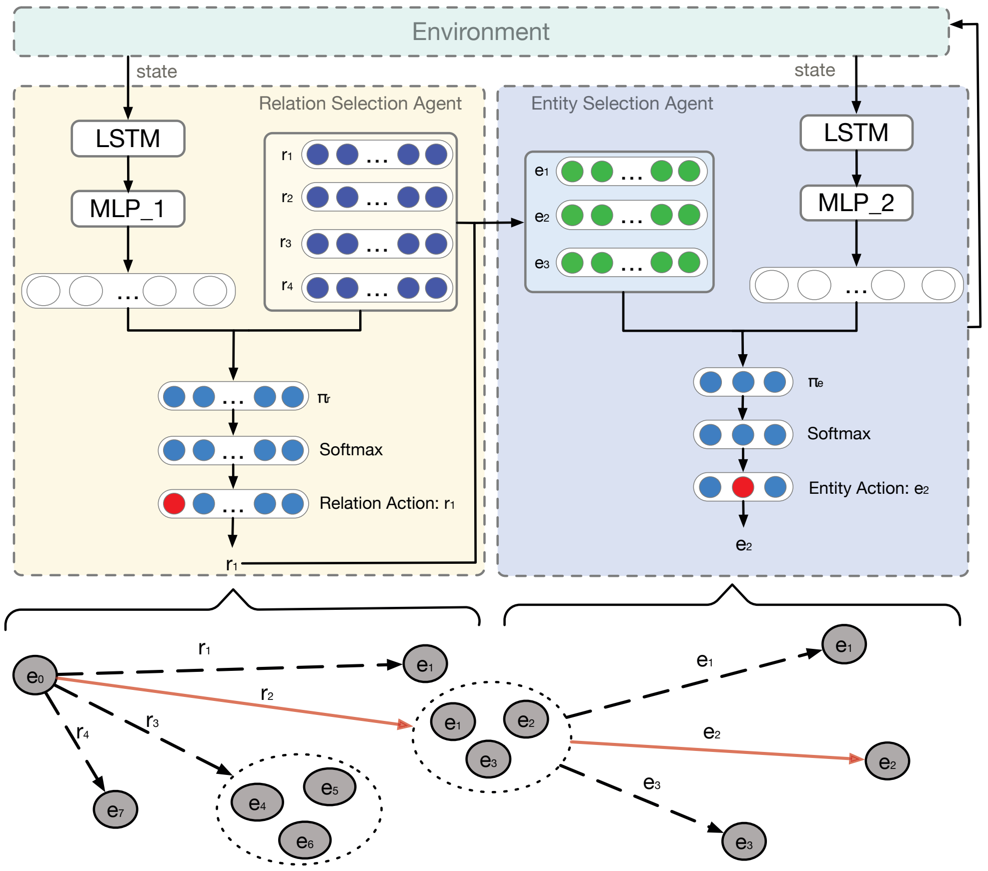

Fig. 2. An illustrative diagram of the proposed MARLPaR method.

The well-defined reward, considering the global accuracy, the path efficiency and the path diversity, encourages the model to find different paths between two entities. However, DeepPath does not take the entity selection into consideration in the path finding step and executes bi-directional path-constrained search to fill the entities into the relation path after all relation paths have been found. Then DeepPath feeds the gathered paths into a supervised learning model, the same as PRA does.

*2) MINERVA:* MINERVA views the path reasoning as an MDP, which starts from the query entity and walks over the KG by selecting an output triple from the current entity with the guidance of the query relation to find the correct target entity. Different from DeepPath, MINERVA uses a complete model to do relation path finding and answer entity reasoning. That is, MINERVA not only finds the path to the target entity, but also gives each path a confidence score. MINERVA uses the information of entities and jointly selects an entity-relation pair via a single policy network under the supervision of the query relation at each step.

## III. The MARLPaR Method

In this section, we elaborate the proposed MARLPaR method for the query answering task (i.e., ($e_s$, $r_q$, ?)). We first introduce the RL system, and then describe the three main components of MARLPaR, i.e., the relation selection agent and the entity selection agent, as well as the environment where the two agents 'reside'. The framework of MARLPaR is shown in the top part of Fig. 2. We formulate the query answering task in KGs as a sequential decision making process. The agents select their actions regularly and the environment responds to these actions, and finally the agents translate to new states. The process of choosing relation $r_2$ and entity $e_2$ from the current entity $e_0$ is demonstrated in the bottom part of Fig. 2. In the last part of this section, we introduce the training procedure of MARLPaR.

### A. The RL System

In an agent-based RL System, usually the part except the agents is defined as the environment. Under the scenario of path reasoning in this study, we formulate the RL system as a finite MDP over a KG. A KG contains a collection of triples $\{(e_1, r, e_2)|e_1, e_2 \in V, r \in R\}$, where $V$ and $R$ denote the entity and relation sets, respectively. A KG can be viewed as a directed labeled multi-relation graph $(V, E, R)$, with entities in $V$ represented as nodes and triples as edges, e.g., $(e_1, r, e_2)$ is a directed edge labeled with $r$ from $e_1$ to $e_2$, and $E$ is used to denote these edges of the graph. MDP provides a mathematical framework for modeling the sequential decision making process with the agent-environment interface, the MDP on the KG consists of the sets of states $\mathcal{S}$, actions $\mathcal{A}$, rewards $\mathcal{R}$, policies $\pi$ and a transition function$\sigma$. Here, the term 'finite' means that the sets $\mathcal{S}, \mathcal{A}, \mathcal{R}$ have a finite number of elements. Different from the ordinary RL system, MARLPaR has two agents, the sets of states, actions and rewards for each agent are introduced separately below. Policy $\pi$ is a probabilistic distribution over the possible actions for each agent. In the path reasoning problem, the transition is deterministic because whenever the agents take actions we will know beforehand which nodes the environment will transit to.

## *B. The Relation Selection Agent*

The relation selection agent (the RS agent for short) is represented as a 3-tuple $\langle S^{RS}, \mathcal{A}^{RS}, \pi^{RS}\rangle$, where $S^{RS}$, $\mathcal{A}^{RS}$, and $\pi^{RS}$ denotes its state, available action set, and policy, respectively. At time step $t$, the state $S_t^{RS}$ is denoted by the current entity $e_t$ where the RS agent stays at, i.e., $S_t^{RS} = e_t$, considering the prior knowledge already known by the agent, the query relation $r_q$ and the head entity $e_s$ should also be encoded in the state; the set of available actions of the RS agent consists of the relation labels of all outgoing edges of the current entity $e_t$. Formally, $\mathcal{A}_t^{RS} = \{r \in R : (e_t, r, v) \in E, e_t, v \in V\}$. This means that the RS agent has options to select which outgoing relation it intends to take at each time step. We add $NO\_OP$ as a special action into the action set, with which the RS agent decides not to take any action at the current time step. Therefore, $\mathcal{A}_t^{RS} = \mathcal{A}_t^{RS} \cup NO\_OP$; $\pi_t^{RS}$ determines the probabilities of individual possible actions, which indicates with how much possibility the RS agent takes the corresponding action at time step $t$. Formally, $\pi_t^{RS}$ is defined as: $H_t^{RS} \rightarrow \mathcal{P}(\mathcal{A}_t^{RS})$, where the history $H_t^{RS} = (H_{t-1}^{RS}, A_{t-1}^{RS})$. Here, $A_{t-1}^{RS}$ is the action of the RS agent taken at step $t-1$, and we use the same symbol in bold to represent the randomly initialized embeddings of actions/relations, which are trained together with other parameters in the model. In this study, we adopt a Long Short-Term Memory (LSTM) network to encode $H_t^{RS}$ as a continuous vector $\mathbf{H_t^{RS}} \in \mathbb{R}^d$, where $d$ is the dimension of relation/entity embeddings. Formally, the embedding of $H_t^{RS}$ is defined as:

$$\mathbf{H_t^{RS}} = \mathrm{LSTM}(\mathbf{H_t^{RS}}, \mathbf{A_{t-1}^{RS}}), \tag{1}$$

where $\mathbf{A_{t-1}^{RS}}$ is the embedding of the action taken at time step $t-1$.

Based on the history embedding $\mathbf{H_t^{RS}}$, the RS agent further employs a policy network to select an action from all available actions $\mathcal{A}_t^r$ conditioned on the query relation. Specifically, the history embedding $\mathbf{H_t^{RS}}$ is concatenated with the embedding of the query relation $\mathbf{r_q}$, and is then fed into a feed forward network with a ReLU nonlinearity layer (i.e., the MLP_1 in Fig. 2). Finally, the MLP layer outputs the probability distribution $\pi_t^{RS}$ over all available relations after taking softmax. And, an action $A_t^{RS}$ is sampled according to $\pi_t^{RS}$. Formally, we have

$$\pi_t^{RS} = \mathrm{softmax}(\mathbf{A^{RS}}(\mathbf{W_1^{RS}}\mathrm{ReLU}(\mathbf{W_2^{RS}}[\mathbf{H_t^{RS}}; \mathbf{r_q}]))), \tag{2}$$

$$A_t^{RS} \sim Categorical(\pi_t^{RS}), \tag{3}$$

where action embedding matrix $\mathbf{A^{RS}}$ stacks the embeddings of all outgoing relations of the current entity $e_t$; $\mathbf{W_1^{RS}}$ and $\mathbf{W_2^{RS}}$ are the weights of the policy network. Note that the policy network for each agent has corresponding biases, respectively. But, they are not shown for brevity.

The RS agent does not receive rewards until time step $t = T$, where $T$ is a pre-defined maximum number of time steps. If the final entity $e_T$ is the correct answer entity, it has a reward of 1, otherwise 0. Notably, although the model might arrive at the correct answer when step $t < T$, the $NO\_OP$ operation attends to keep the agent to stay at the correct answer at the rest of time steps, the $NO\_OP$ operation is equivalent to the 'stop' operation in our experiments and equips the model with the ability of early termination. As MINERVA does, we use a moving average of the cumulative discounted reward as an additive control variant baseline to reduce the variance [27].

## *C. The Entity Selection Agent*

The entity selection agent (the ES agent for short) makes its decision after the RS agent selects a relation $r_t$ at time step $t$. It is represented as a 3-tuple $\langle S^{ES}, \mathcal{A}^{ES}, \pi^{ES}\rangle$, where $S^{ES}$, $\mathcal{A}^{ES}$, and $\pi^{ES}$ denotes its state, possible action set, and policy. At time step $t$, the state $S_t^{ES}$ is denoted by the current entity $e_t$. The ES agent takes $S_t^{ES}$ as well as the prior knowledge $(r_q, e_s)$ as input and outputs the probabilities of all actions in possible action set; the set of all possible actions is the tail entity set of edges with $e_t$ as their head entity and $r_t$ as their relation. Formally, $\mathcal{A}_t^{ES} = \{v \in V : (e_t, r_t, v) \in E\}$; $\pi_t^{ES}$ determines the probabilities of individual possible actions, which is defined as $H_t^{ES} \rightarrow \mathcal{P}(\mathcal{A}_t^{ES})$, where $H_t^{ES} = (H_{t-1}^{ES}, A_{t-1}^{ES})$. We also use LSTM to encode $H_t^{ES}$ as a continuous vector $\mathbf{H_t^{ES}} \in \mathbb{R}^d$. That is,

$$\mathbf{H_t^{ES}} = \mathrm{LSTM}(\mathbf{H_{t-1}^{ES}}, \mathbf{A_{t-1}^{ES}}), \tag{4}$$

where $\mathbf{A_{t-1}^{ES}}$ denotes the embedding of entity chosen at step $t-1$. Similar to the RS agent, we use a feed forward network with a ReLU nonlinearity layer (i.e., the MLP_2 in Fig. 2) and take the concatenation of the history embedding and the query relation embedding as input. After taking softmax, it outputs the probability distribution $\pi_t^{ES}$ over all possible tail entities, which is defined as:

$$\pi_t^{ES} = \mathrm{softmax}(\mathbf{A^{ES}}\mathbf{W_1^{ES}}\mathrm{ReLU}(\mathbf{W_2^{ES}}[\mathbf{H_t^{ES}}; \mathbf{r_q}])), \tag{5}$$

$$A_t^{ES} \sim Categorical(\pi_t^{ES}), \tag{6}$$

where $\mathbf{W_1^{ES}}$ and $\mathbf{W_2^{ES}}$ are the parameters of the policy network. All possible action embedding matrix $\mathbf{A^{ES}}$ is formed by stacking all the embeddings of the tail entities of $r_t$. And an action $A_t^{ES}$ is sampled according to $\pi_t^{ES}$.

The ES agent shares the same reward of the RS agent at each time step. A moving average of the cumulative discounted reward is also token as an additive control variant baseline to reduce the variance.

### D. Alternate Training Strategy

For the two policy networks of our multi-agent model described above, we want to find the parameters $\theta$ that maximizes the following expected reward:

$$J(\theta) = \mathbb{E}_{(e_1,r,e_2)\sim D}\mathbb{E}_{A_1^{RS},...,A_T^{RS}\sim\pi^{RS},A_1^{ES},...,A_T^{ES}\sim\pi^{ES}} [R(S_T)|S_1=(e_1,r,e_2)], \tag{7}$$

where $\theta = (\theta^{RS}, \theta^{ES})$ is the parameters of the two agents, $D$ is a true underlying distribution that $(e_1, r, e_2) \sim D$. The large action spaces of the RS agent and the ES agent make it a big challenge to find the optimal parameters for the object function mentioned above. Especially, the two agents influence each other during path searching. It is difficult to train an agent based on the results of a divergent agent because the convergence of the model is not guaranteed.

To solve these problems, we use the alternate training strategy. Firstly, the single RS agent is trained, with selecting an entity from the tail entity set using stochastic sampling strategy at each time step. Then, the two agents are trained with the parameters of another fixed alternately. The reward of the whole model rises alternatively under this strategy, so that each agent maximizes its reward separately. For the RS agent, the parameters $\theta^{RS}$ are trained to maximize the following expected reward:

$$J(\theta^{RS}) = \mathbb{E}_{(e_1,r,e_2)\sim D}\mathbb{E}_{A_1^{RS},...,A_{T-1}^{RS}\sim\pi^{RS}}[R(S_T)|S_1=(e_1,r,e_2)]. \tag{8}$$

And for the ES agent, we find the parameters $\theta^{ES}$ to maximize the following expected reward:

$$J(\theta^{ES}) = \mathbb{E}_{(e_1,r,e_2)\sim D}\mathbb{E}_{A_1^{ES},...,A_{T-1}^{ES}\sim\pi^{ES}}[R(S_T)|S_1=(e_1,r,e_2)]. \tag{9}$$

To solve this optimization problem, the model is trained using policy gradient [9], [28]. We use Adam [29] for updating the parameters $\theta$ of the two agents. By fixing $\theta^{ES}$ to train the RS agent and fixing $\theta^{RS}$ to train the ES agent, we maximize the cumulative reward of the whole model. We approximate the second expectation in (8) and (9) by running multiple rollouts for each training sample. And an entropy regularization term is further added to encourage the agents to find diverse paths as MINERVA does.

## IV. Experiments

We evaluate MARLPaR on WN18RR [26] and NELL-995 [12] datasets in the query answering task, comparing with both embedding based methods and path based methods. To verify that MARLPaR is more powerful to solve the path reasoning problem when a 1-N/N-N relation appears in the reasoning path, we further do some statistics on the reasoning paths found by MARLPaR. Then a few case studies which compare the reasoning paths found by MARLPaR with the ones found by MINERVA in some relation tasks also validate the effectiveness of our model.

TABLE I
Statistics of the datasets used in the experiments.

| Dataset | #Entities | #Relations | #Triples | #Queries |
|---|---|---|---|---|
| WN18RR | 40943 | 11 | 86835 | 3134 |
| NELL-995 | 75492 | 200 | 154213 | 3992 |

### A. Datasets

WN18 is subset of WordNet [30], which consists of 18 relations and 40943 entities. WN18RR is an alteration of WN18, which addresses the shortcoming of WN18 that the test set contains many reversed triples appearedin the training set.

NELL-995 was developed in [12]. The relations $generalizations$ and $haswikipediaurl$ are removed from the 995th iteration of NELL system [31]. Then the dataset keeps the triples of the Top-200 relations. The statistic information of the two datasets is demonstrated in Table I.

For each dataset, we add the inverse triples following the previous works [3], [12]–[14]. That is, for each triple $(h, r, t)$ in the dataset, we append $(t, r^{-1}, h)$ to the dataset if $(t, r^{-1}, h)$ is not contained in it. This operation is suitable for path reasoning because the model is able to revise some mistakes by stepping backward through these inverse triples. After this operation, WN18RR contains 22 relations and NELL-995 contains 400 relations. Following [14], we classify a relation to 1-N/N-N relation if the average cardinality of the tail entity set to the head entity set is greater than 1.5. 7 relations, with the proportion of 32%, are 1-N/N-N relations in WN18RR and 27% relations are 1-N/N-N relations in NELL-995. The large proportion of 1-N/N-N relations also manifests entity selection is practical and thus important in path reasoning.

### B. Baseline Methods and Metrics

On WN18RR, we compare MARLPaR with recently proposed embedding based methods, i.e., ConvE [26], DistMult [25] and ComplEx [16]. Besides, the path based method MINERVA [13] is adopted as another baseline. The results of baselines we used come from [13]. For NELL-995, we measure MARLPaR on 9 relation tasks following MINERVA. For all the datasets, we adopt the widely used Hits@N, the percentage of correct entities ranked in top N among all the entities, as our metric.

### C. Implementation Details

For WN18RR, the embedding size of the entity and relation for the two agents are set to 50, the dimension of the MLP layer and LSTM layer are set to 200, the layer number of LSTM is 1 and the max path length (i.e., the max step number) $T = 3$. Similarly, we first train the single RS agent for 500 iterations, then train the two agents alternately for 1500 iterations. For NELL-995, the embedding size of entity and relation are set to 50, the dimension of the MLP layer and LSTM layer are set to 200, the layer number of LSTM is 1 and the max path length $T = 3$. We first train the single RS agent for 100 iterations, then train the two agents alternately

TABLE II
RESULTS (HITS@N) ON WN18RR.

| Model | Hits@1 | Hits@3 | Hits@10 |
|---|---|---|---|
| ConvE | 0.306 | 0.360 | 0.411 |
| DistMult | 0.389 | 0.439 | 0.491 |
| ComplEx | 0.411 | 0.458 | 0.507 |
| MINERVA | 0.413 | 0.456 | 0.513 |
| R-MARLPaR | 0.400 | 0.456 | 0.517 |
| MARLPaR | **0.421** | **0.472** | **0.520** |

TABLE III
RESULTS (HITS@N) ON NELL-995.

| Task | Hits@1 | Hits@3 | Hits@10 |
|---|---|---|---|
| athleteplaysinleague | 0.80/0.78 | 0.82/**0.85** | 0.85/**0.86** |
| worksfor | 0.57/**0.58** | 0.69/**0.71** | 0.74/**0.77** |
| organizationhiredperson | 0.55/**0.56** | 0.62/**0.64** | 0.67/0.67 |
| athleteplayssport | 0.46/**0.47** | 0.48/0.47 | 0.48/0.47 |
| teamplayssport | 0.65/**0.66** | 0.74/**0.78** | 0.79/**0.80** |
| personborninlocation | 0.57/**0.58** | 0.62/**0.64** | 0.67/**0.68** |
| athletehomestadium | 0.73/**0.80** | 0.79/**0.85** | 0.85/**0.87** |
| organizationheadquarteredincity | 0.79/**0.80** | 0.82/**0.83** | 0.84/0.83 |
| athleteplaysforteam | 0.51/0.51 | 0.58/**0.59** | 0.61/**0.62** |

for 400 iterations in each relation task. Note that the results of MINERVA are based on the implementation released by the original paper. We retrain the MINERVA model on each task and the best configurations are determined according to the validation set.

## D. Results

Tables II and III demonstrate the experimental results of the link prediction task on WN18RR and NELL-995. In Table III, the results are represented in the form of MINERVA/MARLPaR. From Table II it can be seen that MARLPaR outperforms both embedding based methods and path based one on WN18RR. Relation paths are the common characteristics between different entity pairs which have the same relation. The RS agent finds the generalities among different entity pairs which have the same relation, while the ES agent has the ability to catch the peculiarities of different entity pairs even they have the same relation. Thus, MARLPaR is capable of picking the most contextual relevant path out among all the found relation paths. It is important for a query answering task because the model should take only a single path as the evidence to infer new predictions. In Table III, the higher values of Hits@1 and Hits@3 in most tasks on NELL-995 also validate that our model distinguishes the relation-entity paths which have the same relations, and highlights the correct one to the answer.

## E. Analysis of Reasoning Paths

To further demonstrate that using only one agent cannot sufficiently model the relation and the entity selection and demonstrate the influence of the ES agent, an ablation model called Random MARLPaR (R-MARLPaR) is adopted as a baseline. More specifically, we change the ES agent of MARLPaR to a sample average method, the model is conducted to find a path $p$ with a sequence of relations, $r_1, r_2, ..., r_T$, using the uniform function as the policy function of the ES agent. The results of MINERVA, R-MARLPaR, and MARLPaR are shown in Fig. 3. As expected, the performance of MINERVA is similar to R-MARLPaR, which illustrates only a single agent cannot model the relation and entity selection well enough although MINERVA uses the information of entity and chooses entity and relation jointly (MINERVA chooses one triple from the outgoing edges of the current entity at each time step). MARLPaR performs better than R-MARLPaR for the reason that the multi-agent model is more sensible in the situation that has many possibilities of entity choices in a logistic relation path, while the unsuitable entities chosen by MINERVA and R-MARLPaR make them fail in arriving at the target entity.

To testify that MARLPaR has the ability to find its direction when a 1-N/N-N relation is contained in a path, we figure out the proportions of paths which contain 1-N/N-N relations in Top-N reasoning paths found by MINERVA and MARLPaR on WN18RR. The results in Table IV demonstrate that

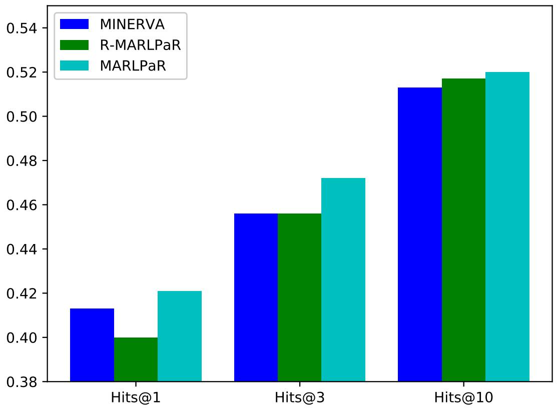

Fig. 3. Results (Hits@N) on WN18, and the three results in each bar group correspond to MINERVA, R-MARLPaR and MARLPaR, respectively.

TABLE IV
THE PROPORTIONS OF PATHS CONTAINING 1-N/N-N RELATIONS IN ALL TOP-N PATHS.

| Model | Top-10 | Top-50 |
|---|---|---|
| MINERVA | 0.256 | 0.399 |
| R-MARLPaR | 0.251 | 0.388 |
| MARLPaR | **0.274** | **0.433** |

TABLE V
THE PROPORTION OF RIGHT PATHS CONTAINING 1-N/N-N RELATIONS IN ALL RIGHT PATHS RANKED IN TOP N.

| Model | Top-10 | Top-50 |
|---|---|---|
| MINERVA | 0.270 | 0.215 |
| R-MARLPaR | 0.262 | 0.226 |
| MARLPaR | **0.338** | **0.310** |

TABLE VI
EXAMPLES OF REASONING PATHS FOUND BY THE PROPOSED MARLPaR METHOD AND MINERVA FOR SEVERAL TASKS.

| Relation | Reasoning Path |
|---|---|
| $hypernym$ | MINERVA:02314321 $\xrightarrow{member\_meronym^{-1}}$ 02313495 $\xrightarrow{member\_meronym}$ 02313709 $\xrightarrow{hypernym}$ 01905661 × <br> MARLPaR:02314321 $\xrightarrow{member\_meronym^{-1}}$ 02313495 $\xrightarrow{member\_meronym}$ 02314001 $\xrightarrow{hypernym}$ 08102555 ✓ |
| $member\_meronym$ | MINERVA:01589125 $\xrightarrow{member\_meronym^{-1}}$ 01589582 $\xrightarrow{NO\_OP}$ 01589582 $\xrightarrow{hypernym}$ 01507175 × <br> MARLPaR:01589125 $\xrightarrow{member\_meronym^{-1}}$ 01590042 $\xrightarrow{member\_meronym}$ 01590220 $\xrightarrow{hypernym}$ 01589286 ✓ |
| $has\_part$ | MINERVA:13726074 $\xrightarrow{NP\_OP}$ 13726074 $\xrightarrow{hypernym}$ 13609507 $\xrightarrow{hypernym}$ 13726947 × <br> MARLPaR:13726074 $\xrightarrow{hypernym}$ 13609507 $\xrightarrow{NO\_OP}$ 13609507 $\xrightarrow{hypernym}$ 13725726 ✓ |
| $organization\_$ <br> $headquartered\_in\_city$ | MINERVA:$bank\_prudential\_financial$ $\xrightarrow{headquarteredin}$ $city\_newark$ $\xrightarrow{buildinglocatedincity^{-1}}$ <br> $building\_prudential\_center$ $\xrightarrow{proxyfor^{-1}}$ $profession\_newark$ × <br> MARLPaR:$bank\_prudential\_financial$ $\xrightarrow{headquarteredin}$ $city\_newark$ $\xrightarrow{buildinglocatedincity^{-1}}$ <br> $building\_prudential\_center$ $\xrightarrow{proxyfor^{-1}}$ $city\_newark$ ✓ |
| $worksfor$ | MINERVA:$ceo\_brian\_moynihan$ $\xrightarrow{personleadsorganization}$ $bank\_bank\_america$ $\xrightarrow{organizationterminatedperson}$ <br> $ceo\_kenneth\_lewis$ $\xrightarrow{personleadsorganization}$ $country\_america$ × <br> MARLPaR:$ceo\_brian\_moynihan$ $\xrightarrow{personleadsorganization}$ $bank\_bank\_america$ $\xrightarrow{organizationterminatedperson}$ <br> $ceo\_brian\_moynihan$ $\xrightarrow{personleadsorganization}$ $bank\_bank\_america$ ✓ |

MARLPaR tends to find more relation paths containing 1-N/N-N relations than MINERVA does. To have a deep insight of the quality of these paths, for every test sample which has at least one path to the answer with the rank smaller than N, we take the Top-1 successful path out. The proportions of paths which contain 1-N/N-N relations in all the successful paths are shown in Table V. The result illustrates that MARLPaR does not dodge the 1-N/N-N relations in path finding step and can find the right direction to the answer when a 1-N/N-N relation appears in the path.

### *F. Case Studies*

We show a few reasoning paths found by MARLPaR and MINERVA in Table VI. We choose the Top-1 path for each test sample according to the scores calculated by MINERVA and MARLPaR. In this table, the first three paths come from WN18RR and the others are from NELL-995. The results demonstrate that MINERVA may make mistakes even the logistic relation path is correct. Take the relation $organization\_headquartered\_in\_city$ for example, MINERVA finds the correct relation path $headquartered\ in \rightarrow building\ location\ in\ city^{-1} \rightarrow proxy\ for^{-1}$, but the final entity is wrong because it does not sufficiently distinguish the difference between entities in the tail entity set of 1-N relation $proxy\ for^{-1}$. The joint selection of the relation-entity pair underestimates the entity information existing in the selected nodes, although $city\_newark$ has been chosen as a previous node in the path, MINERVA brushes past the correct answer. However, MARLPaR is not only capable of finding the correct relation path, but also has the ability to reason on the relation paths and find the relation-entity paths arriving at the answer entity.

## V. CONCLUSION

In this paper, we explored the significance of entity selection on query answering task of path reasoning and proposed a Multi-Agent and Reinforcement Learning based method for Path Reasoning (i.e. MARLPaR). Specifically, we trained a relation selection agent and an entity selection agent jointly. The relation selection agent is used to find the common logistic paths for specific query relation. The entity selection agent tries to choose the most suitable entity from the tail entity set of the relations to find the answer entity accurately. Experiment results on two benchmark datasets manifest the merits and superiority of the proposed MARLPaR model.

## ACKNOWLEDGMENT

The authors would like to thank Manling Li and Long Bai for their kind help. This work is supported by National Key Research and Development Program of China under grants 2016YFB1000902 and 2017YFC0820404, and National Natural Science Foundation of China under grants 61772501, 61572473, 61572469, and 91646120.

## REFERENCES

[1] R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A study on big knowledge and its engineering issues," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[2] N. Lao, T. Mitchell, and W. W. Cohen, "Random walk inference and learning in a large scale knowledge base," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011, pp. 529–539.

[3] A. Neelakantan, B. Roth, and A. Mc-Callum, "Compositional vector space models for knowledge base inference," in *2015 AAAI spring symposium series*, 2015.

[4] K. Toutanova, V. Lin, W.-t. Yih, H. Poon, and C. Quirk, "Compositional learning of embeddings for relation paths in knowledge base and text," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2016, pp. 1434–1444.

[5] K. Guu, J. Miller, and P. Liang, "Traversing knowledge graphs in vector space," *arXiv preprint arXiv:1506.01094*, 2015.

[6] M. Gardner, P. P. Talukdar, B. Kisiel, and T. Mitchell, "Improving learning and inference in a large knowledge-base using latent syntactic cues," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 833–838.

[7] M. Gardner, P. Talukdar, J. Krishnamurthy, and T. Mitchell, "Incorporating vector space similarity in random walk inference over knowledge bases," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 397–406.

[8] R. Das, A. Neelakantan, D. Belanger, and A. McCallum, "Chains of reasoning over entities, relations, and text using recurrent neural networks," *arXiv preprint arXiv:1607.01426*, 2016.

[9] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[10] J. Feng, M. Huang, L. Zhao, Y. Yang, and X. Zhu, "Reinforcement learning for relation classification from noisy data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[11] P. Qin, W. Xu, and W. Y. Wang, "Robust distant supervision relation extraction via deep reinforcement learning," *arXiv preprint arXiv:1805.09927*, 2018.

[12] W. Xiong, T. Hoang, and W. Y. Wang, "Deeppath: A reinforcement learning method for knowledge graph reasoning," *arXiv preprint arXiv:1707.06690*, 2017.

[13] R. Das, S. Dhuliawala, M. Zaheer, L. Vilnis, I. Durugkar, A. Krishnamurthy, A. Smola, and A. McCallum, "Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning," *arXiv preprint arXiv:1711.05851*, 2017.

[14] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in neural information processing systems*, 2013, pp. 2787–2795.

[15] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in neural information processing systems*, 2013, pp. 926–934.

[16] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *International Conference on Machine Learning*, 2016, pp. 2071–2080.

[17] Y. Jia, Y. Wang, X. Jin, and X. Cheng, "Path-specific knowledge graph embedding," *Knowledge-Based Systems*, vol. 151, pp. 37–44, 2018.

[18] Y. Jia, Y. Wang, X. Jin, H. Lin, and X. Cheng, "Knowledge graph embedding: A locally and temporally adaptive translation-based approach," *ACM Transactions on the Web*, vol. 12, no. 2, p. 8, 2017.

[19] Y. W. Saiping Guan, Xiaolong Jin and X. Cheng, "Shared embeddings based neural network for knowledge graph completion," in *The 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 22–26.

[20] S. Guan, X. Jin, Y. Jia, Y. Wang, H. Shen, and X. Cheng, "Self-learning and embedding based entity alignment," in *IEEE International Conference on Big Knowledge*. IEEE, 2017, pp. 33–40.

[21] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes." in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 14, 2014, pp. 1112–1119.

[22] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion." in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 15, 2015, pp. 2181–2187.

[23] Y. Jia, Y. Wang, H. Lin, X. Jin, and X. Cheng, "Locally adaptive translation for knowledge graph embedding." in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, pp. 992–998.

[24] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Advances in Neural Information Processing Systems*, 2012, pp. 3167–3175.

[25] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv preprint arXiv:1412.6575*, 2014.

[26] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," *arXiv preprint arXiv:1707.01476*, 2017.

[27] M. Evans and T. Swartz, *Approximating integrals via Monte Carlo and deterministic methods*. OUP Oxford, 2000, vol. 20.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[31] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell, "Toward an architecture for never-ending language learning." in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 5. Atlanta, 2010, p. 3.