An Interpretable Reasoning Network for Multi-Relation Question Answering

Mantong Zhou **Minlie Huang*** **Xiaoyan Zhu**
State Key Lab. of Intelligent Technology and Systems,
National Lab. for Information Science and Technology,
Dept. of Computer Science and Technology, Tsinghua University, Beijing, PR China
zmt.keke@gmail.com, aihuang@tsinghua.edu.cn,
zxy-dcs@tsinghua.edu.cn

Abstract

Multi-relation Question Answering is a challenging task, due to the requirement of elaborated analysis on questions and reasoning over multiple fact triples in knowledge base. In this paper, we present a novel model called **Interpretable Reasoning Network** that employs an interpretable, hop-by-hop reasoning process for question answering. The model dynamically decides which part of an input question should be analyzed at each hop; predicts a relation that corresponds to the current parsed results; utilizes the predicted relation to update the question representation and the state of the reasoning process; and then drives the next-hop reasoning. Experiments show that our model yields state-of-the-art results on two datasets. More interestingly, the model can offer traceable and observable intermediate predictions for reasoning analysis and failure diagnosis, thereby allowing manual manipulation in predicting the final answer.

1 Introduction

Open-domain Question Answering (QA) has always been a hot topic in AI and this task has recently been facilitated by large-scale Knowledge Bases (KBs) such as Freebase (Bollacker et al., 2008). However, due to the variety and complexity of language and knowledge, open-domain question answering over knowledge bases (KBQA) is still a challenging task.

Question answering over knowledge bases falls into two types, namely single-relation QA and multi-relation QA, as argued by Yin et al. (2016). Single-relation questions, such as *"How old is Obama?"*, can be answered by finding one fact triple in KB, and this task has been widely studied (Bordes et al., 2015; Xu et al., 2016; Savenkov and Agichtein, 2017). In comparison, reasoning over multiple fact triples is required to answer multi-relation questions such as *"Name a soccer player who plays at forward position at the club Borussia Dortmund."* where more than one entity and relation are mentioned. Compared to single-relation QA, multi-relation QA is yet to be addressed.

Previous studies on QA over knowledge bases can be roughly categorized into two lines: semantic parsing and embedding-based models. Semantic parsing models (Yih et al., 2014; Yih et al., 2016) obtain competitive performance at the cost of hand-crafted features and manual annotations, but lack the ability to generalize to other domains. In contrast, embedding-based models (Bordes et al., 2014b; Hao et al., 2017; Yavuz et al., 2017) can be trained end-to-end with weak supervision, but existing methods are not adequate to handle multi-relation QA due to the lack of reasoning ability.

Recent reasoning models (Miller et al., 2016; Wang et al., 2017) mainly concentrate on Reading Comprehension (RC) which requires to answer questions according to a given document. However, transferring existing RC methods to KBQA is not trivial. For one reason, the focus of reasoning in RC is usually on understanding the document rather than parsing questions. For another reason, existing reasoning networks are usually designed in a black-box style, making the models less interpretable. While in multi-relation question answering, we believe that an interpretable reasoning process is essential.

In this paper, we propose a novel Interpretable Reasoning Network (IRN) to equip QA systems with the reasoning ability to answer multi-relation questions. Our central idea is to design an interpretable

*Corresponding author: Minlie Huang (aihuang@tsinghua.edu.cn).

Proceedings of the 27th International Conference on Computational Linguistics, pages 2010–2022
Santa Fe, New Mexico, USA, August 20-26, 2018.

reasoning process for a complex question: the reasoning module decides which part of an input question should be analyzed at each hop, and predicted a KB relation that corresponds to the current parsed results. The predicted relation will be used to update the question representation as well as the state of the reasoning module, and helps the model to make the next-hop reasoning. At each hop, an entity will be predicted based on the current state of the reasoning module.

Different from previous models, our model is ***interpretable*** in that the predicted relation and entity at each hop are ***traceable and observable***. At each hop our model has a specific aim to find an appropriate relation based on the iterative analysis of a question, and intermediate output at each hop can be interpreted by the corresponding linked entity. In this manner, IRN offers the ability of visualizing ***a complete reasoning path*** for a complex question, which facilitates reasoning analysis and failure diagnosis, thereby allowing manual manipulation in answer prediction as detailed in our experiments.

The contributions of this paper are in two folds:

1. We design an Interpretable Reasoning Network which can make reasoning on multi-relation questions with multiple triples in KB. Results show that our model obtains state-of-the-art performance.

2. Our model is more interpretable than existing reasoning networks in that the intermediate entities and relations predicted by the hop-by-hop reasoning process construct traceable reasoning paths to clearly reveal how the answer is derived.

2 Related Works

Recent works on QA can be roughly classified into two types: one is semantic-parsing-based and the other is embedding-based. Semantic parsing approaches map questions to logical form queries (Pasupat and Liang, 2015; Yih et al., 2016; Abujabal et al., 2017). These systems are effective but at the cost of heavy data annotation and pattern/grammar engineering. What's more, parsing systems are often constrained on a specific domain and broken down when executing logical queries on incomplete KBs.

Our work follows the line of Embedding-based models (Bordes et al., 2014b; Dong et al., 2015; Xu et al., 2016; Hao et al., 2017; Yavuz et al., 2017) which are recently introduced into the QA community where questions and KB entities are represented by distributed vectors, and QA is formulated as a problem of matching between vectors of questions and answer entities. These models need less grammars as well as annotated data, and are more flexible to deal with incomplete KBs. To make better matching, subgraphs of an entity in KB (Bordes et al., 2014a), answer aspects (Dong et al., 2015; Hao et al., 2017) and external contexts (Xu et al., 2016) can be used to enrich the representation of an answer entity. Though these methods are successful to handle simple questions, answering multi-relation questions or other complex questions is far from solved, since such a task requires reasoning or other elaborated processes.

Our work is also related to recent reasoning models which focus on Reading Comprehension where memory modules are designed to comprehend documents. State-of-the-art memory-based Reading Comprehension models (Sukhbaatar et al., 2015; Kumar et al., 2015; Shen et al., 2016; Wang et al., 2017; Celikyilmaz et al., 2017) make interactions between a question and the corresponding document in a multi-hop manner during reasoning. MemNN (Weston et al., 2015), KVMemN2N (Miller et al., 2016) and EviNet (Savenkov and Agichtein, 2017) transferred the reading comprehension framework to QA where a set of triples is treated as a document and a similar reasoning process can be applied. However, reading comprehension makes reasoning over documents instead of parsing the questions.

Other studies applying hop-by-hop inference into QA can be seen in Neural Programmer (Neelakantan et al., 2015; Neelakantan et al., 2016) and Neural Enquirer (Yin et al., 2015), where deep networks are proposed to parse a question and execute a query on tables. However, Neural Programmer needs to predefine symbolic operations, while Neural Enquirer lacks explicit interpretation. Mou et al. (2017) proposed a model coupling distributed and symbolic execution with REINFORCE algorithm, however, training such a model is challenging. Neural Module Network (Andreas et al., 2015; Andreas et al., 2016) customized network architectures for different patterns of reasoning, making the reasoning network interpretable. However, a dependency parser and the REINFORCE algorithm are required.

3 Interpretable Reasoning Network

3.1 Task Definition

Our goal is to offer an interpretable reasoning network to answer multi-relation questions. Given a question q and its topic entity or subject e_s which can be annotated by some NER tools, the task is to find an entity a in KB as the answer.

In this work, we consider two typical categories of multi-relation questions, a path question (Guu et al., 2015) and a conjunctive question (Zhang et al., 2016), while the former is our major focus.

A path question contains only one topic entity (subject e_s) and its answer (object a) can be found by walking down an answer path consisting of a few relations and the corresponding intermediate entities. We define an **answer path** as a sequence of entities and relations in KB which starts from the subject and ends with the answer like $e_s \xrightarrow{r_1} e_1 \xrightarrow{r_2} ... \xrightarrow{r_n} a$. Relations ($r_i$) are observable (in various natural language forms) in the question, however, the intermediate entities ($e_1 \cdots e_H$) are not. For example, for question *"How old is Obama's daughter?"*, the subject is *Barack_Obama* and the answer path is $Barack\_Obama \xrightarrow{Children} Malia\_Obama \xrightarrow{Age} 18$. Note that since there are 1-to-many relations[1], the range of the intermediate entities can be large, resulting in more than one answer path for a question.

A conjunctive question is a question that contains more than one subject entity and the answer can be obtained by the intersection of results from multiple path queries. For instance, the question *"Name a soccer player who plays at forward position at the club Borussia Dortmund."* has a possible answer as the intersection of results from two path queries[2] $FORWARD \xrightarrow{plays\_position^{-1}} Marco\_Reus$ and $Borussia\_Dortmund \xrightarrow{plays\_in\_club^{-1}} Marco\_Reus$. The details for dealing with conjunctive questions are shown in Fig 2.

3.2 Overview

The reasoning network has three modules: input module, reasoning module, and answer module. The input module encodes the question into a distributed representation and updates the representation hop-by-hop according to the inference results of the reasoning module. The reasoning module initializes its state by the topic entity of a question and predicts a relation on which the model should focus at the current hop, conditioned on the present question and reasoning state. The predicted relation is utilized to update the state vector and the question representation hop-by-hop. The answer module predicts an entity conditioned on the state of the reasoning module.

The process can be illustrated by the example as shown in Figure 1. For question *"How old is Obama's daughter?"*, the subject entity *Barack_Obama* is utilized to initialize the state vector. IRN predicts the first relation *"Children"* at the first hop. The *"Children"* relation is added to the state vector to encode the updated parsing result, and the corresponding natural language form of this relation in the question (here is *"daughter"*) is subtracted from the question to avoid repeatedly analyzing the relation-relevant word *"daughter"*. This procedure is repeated until the *Terminal* relation is predicted.

3.3 Input Module

The input module encodes a question to a vector representation and updates the representation of the question at each hop of the reasoning process: the predicted relation will be subtracted from the current representation to compel the reasoning process to attend to other words that should be analyzed.

Formally, the question $\boldsymbol{X} = x_1, x_2, ..., x_n$ can be initialized by the sum of the word embeddings and updated by subtracting the relation predicted by the reasoning module at the previous hop:

$$\boldsymbol{q}^0 = \sum_{i=1}^{n} \boldsymbol{x}_i \quad (1)$$

$$\boldsymbol{q}^h = \boldsymbol{q}^{h-1} - \boldsymbol{M}_{rq}\hat{\boldsymbol{r}}^h \quad (2)$$

[1]For instance, relation *Children* is one-to-many, where a person may have more than one child.

[2]Superscript -1 stands for the inverse relation.

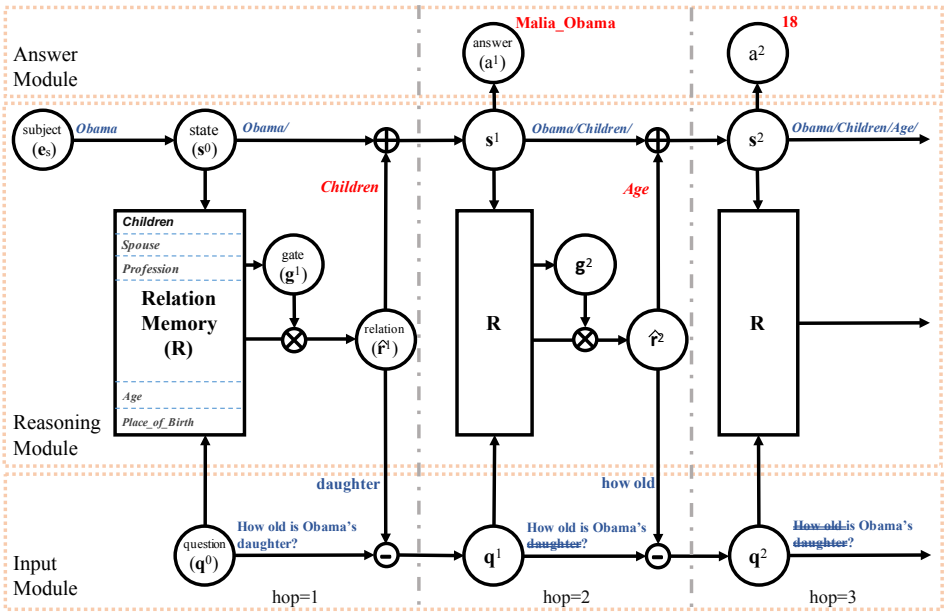

Figure 1: Interpretable Reasoning Network. At each hop IRN computes the probability of selecting the next relation as $\boldsymbol{g}^h$ and obtains a predicted relation $\hat{\boldsymbol{r}}^h$. The predicted relation $\hat{\boldsymbol{r}}^h$ is used to update the question $\boldsymbol{q}^h$ and the state $\boldsymbol{s}^h$ with different projections. The state is initialized by subject as $\boldsymbol{s}^0 = \boldsymbol{e}_s$. The answer path ($Obama \xrightarrow{Children} Malia\_Obama \xrightarrow{Age} 18$) is composed of the predicted relations and entities (in red).

where $\boldsymbol{M}_{rq}$ is a matrix projecting the KB relation space to the natural language question space, $\boldsymbol{q}^{h-1}$ is the question representation at hop $h-1$, and $\hat{\boldsymbol{r}}^h$ defined by Eq. 4 is the predicted relation at hop h. The intuition of such update is that the already analysed part of the question should not be parsed again.

Representing a question as a bag of words might be too simple. However, this method works well in our setting. Future work would consider other sophisticated encoders such as CNN or LSTM.

3.4 Reasoning Module

The reasoning module aims to attend to a particular part of the question at each hop, predict an associated relation in knowledge base, and then update its state.

The reasoning module takes as input the previous state vector ($\boldsymbol{s}^{h-1}$) and the previous question vector ($\boldsymbol{q}^{h-1}$), and then predicts a relation ($\hat{\boldsymbol{r}}^h$) based on the analysis at the current hop. Once the predicted relation ($\hat{\boldsymbol{r}}^h$) is obtained, the relation will be used to update the next question representation ($\boldsymbol{q}^h$) and the state of the reasoning module ($\boldsymbol{s}^h$). In this manner, the reasoning network is traceable and interpretable.

The process can be formally described by the following equations[3]:

$$g_j^h = P(r^h = r_j | \boldsymbol{q}^{h-1}, \boldsymbol{s}^{h-1}) = \text{softmax}((\boldsymbol{M}_{rq}\boldsymbol{r}_j)^T \boldsymbol{q}^{h-1} + (\boldsymbol{M}_{rs}\boldsymbol{r}_j)^T \boldsymbol{s}^{h-1}) \tag{3}$$

$$\hat{\boldsymbol{r}}^h = \sum_j g_j^h * \boldsymbol{r}_j \tag{4}$$

$$\boldsymbol{s}^h = \boldsymbol{s}^{h-1} + \boldsymbol{M}_{rs}\hat{\boldsymbol{r}}^h \tag{5}$$

where $\boldsymbol{r}_j$ is the embedding of a relation in KB and all the relation embeddings are stored in a static memory R, and $\boldsymbol{s}^h$ is the state of the reasoning module at hop h. g_j^h is the probability of selecting the j^{th} relation in KB and $\boldsymbol{M}_{rs}$ is the projection matrix mapping $\boldsymbol{r}$ from the relation space to the state space. $\boldsymbol{M}_{rq}$ is the same projection matrix used in Eq. 2 to map $\boldsymbol{r}$ from the relation space to the question space.

We initialize the state vector with the topic entity (subject) $\boldsymbol{s}^0 = \boldsymbol{e}_s$. IRN will learn to enrich the state representation hop by hop, for instance, at the first hop $\boldsymbol{s}^1 \approx \boldsymbol{e}_s + \boldsymbol{r}_1$, and at the second hop $\boldsymbol{s}^2 \approx \boldsymbol{e}_s + \boldsymbol{r}_1 + \boldsymbol{r}_2$, intuitively. In this manner, the state vector encodes historical information.

[3] $\boldsymbol{a}^T\boldsymbol{b}$ is the inner-product of vector $\boldsymbol{a}$ and $\boldsymbol{b}$.

In order to signify when the reasoning process should stop, we augment the relation set with the *Terminal* relation. Once the reasoning module predicts the *Terminal* relation, the reasoning process will stop, and the final answer will be the output when the last non-terminal relation is added to the state $\boldsymbol{s}$.

3.5 Answer Module

The answer module chooses the corresponding entity from KB at each hop (denoted as a^h). At the last hop, the selected entity is chosen as the final answer, while at the intermediate hops, the predictions of these entities can be inspected to help reasoning analysis and failure diagnosis.

More formally, an entity at each hop can be predicted as follows:

$$\boldsymbol{e}^h = \boldsymbol{M}_{se}\boldsymbol{s}^h \tag{6}$$

$$o_j^h = P(a^h = e_j|\boldsymbol{s}^h) = \text{softmax}(\boldsymbol{e}_j^T\boldsymbol{e}^h) \tag{7}$$

$\boldsymbol{M}_{se}$ is used to transfer from the state space ($\boldsymbol{s}^h$) to the entity space ($\boldsymbol{e}^h$) to bridge the representation gap between the two spaces. $\boldsymbol{e}_j$ is the embedding vector of the j^{th} entity in KB.

3.6 Loss Function

We adopt cross entropy to define the loss function. The first loss term is defined on the intermediate prediction of relations, while the second term on the prediction of entities.

The loss on one instance is defined as follows:

$$\mathcal{L} = \sum_h \mathcal{C}_r(h) + \lambda\mathcal{C}_a(h) \tag{8}$$

$$\mathcal{C}_r(h) = -\sum_{j=1}^{n_r}[\hat{g}_j^h \ln g_j^h], \ \mathcal{C}_a(h) = -\sum_{i=1}^{n_e}[\hat{o}_i^h \ln o_i^h]$$

where n_r/n_e is the number of relations/entities in KB respectively, $\hat{\boldsymbol{g}}^h$ is the gold distribution (one-hot) over relations at hop h, $\boldsymbol{g}^h$ is the predicted distribution defined by Eq. 3, $\hat{\boldsymbol{o}}$ is the gold distribution over entities, which is also one-hot representation, and $\boldsymbol{o}$ is defined by Eq. 7. λ is a hyper parameter to balance the two terms.

Note that the training data is in the form of $(q, < e_s, r_1, e_1, ..., a >)$, which indicates that the model can incorporate supervision not only from the final answer (referred to as IRN-weak), but also from the intermediate relations and entities along the answer path (referred to as IRN).

3.7 Multitask Training for KB Representation

In order to incorporate more constraints from KB[4], we learn the embeddings of entities and relations as well as the space transition matrix with a multitask training schema. For a given fact triple in KB, (e_s, r, e_o), the representations of the entities and the relation apply the following constraint:

$$\boldsymbol{M}_{se}(\boldsymbol{e}_s + \boldsymbol{r}) = \boldsymbol{e}_o \tag{9}$$

where $\boldsymbol{e}_s, r, \boldsymbol{e}_o$ are embeddings of the subject (or head) entity, relation, and the object (or tail) entity. This idea is inspired by TransE (Bordes et al., 2013), but we adopt $\boldsymbol{M}_{se}$ (see Eq. 6) as a transfer matrix to bridge the representation gap between the state space (here $\boldsymbol{e}_s + \boldsymbol{r} = \boldsymbol{s}$) and the entity space (here $\boldsymbol{e}_o = \boldsymbol{e}$).

The parameters are updated with a multi-task training schema. We first learn the KB embeddings $\boldsymbol{e}/\boldsymbol{r}$ and the transformation matrix $\boldsymbol{M}_{se}$ to fit Eq. 9 with several epoches. This is the task of KB embedding training. Then, we update all the parameters of IRN under supervision from the QA task with one epoch, which is the task of QA training. We run the two tasks iteratively.

With the help of the auxiliary KB embedding training, IRN not only utilizes the additional information from KB to make better inferences, but also has an ability to deal with incomplete answer paths. For example, even if the connection between *Barack_Obama* and *Malia_Obama* is not present in KB, our model can still make correct prediction thanks to $\boldsymbol{M}_{se}(\boldsymbol{e}_{Barack\_Obama} + \boldsymbol{r}_{Children}) \approx \boldsymbol{e}_{Malia\_Obama}$.

[4]Constraint from KB means that two entities and a relation form a triple in KB, as (e_s, r, e_o).

3.8 Dealing with Conjunctive Questions

IRN is not limited to only path questions. For a conjunctive question that contains more than one topic entity, the answer can be found by executing multiple IRNs with the same parameters in parallel and then obtaining the intersection of individual results.

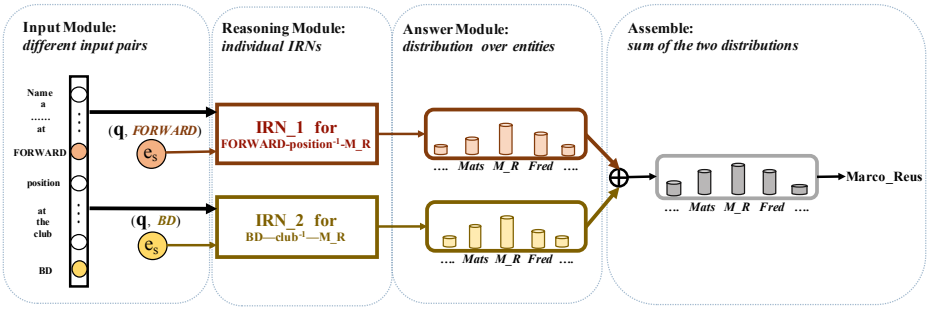

Figure 2: An assembly of two IRNs to handle a conjunctive question with two subjects. Different IRNs take as input the same question but different subjects and output the distribution over the candidate answers. The final answer is selected after summing the two distributions.

This process is exemplified by Figure 2. The input question *"Name a soccer player who plays at forward position at the club Borussia Dortmund"* has two subject entities, *"FORWARD"* and *"Borussia_Dortmund(BD)"*. One IRN (IRN_1) takes the original question and *"FORWARD"* as input, and then predicts possible objects for path query *"FORWARD* $\xrightarrow{plays\_position^{-1}}$ *?(Marco_Reus)"*.[5] The output is a distribution over entities. Similarly, another IRN (IRN_2) tackles the path query *"BD* $\xrightarrow{plays\_in\_club^{-1}}$ *?(Marco_Reus)"* where the input is the same question but another subject entity *"Borussia Dortmund(BD)"*. After summing the two output distributions, the answer *"Marco_Reus"* is chosen with the largest probability.

4 Data Preparation

We prepared two KBQA datasets to evaluate our Interpretable Reasoning Network: one is PathQuestion[6], constructed by ourselves, and the other is WorldCup2014, adopted from (Zhang et al., 2016).

4.1 PathQuestion

We adopted two subsets of Freebase (Bollacker et al., 2008) as Knowledge Bases to construct the PathQuestion (PQ) and the PathQuestion-Large (PQL) datasets. We extracted paths between two entities which span two hops ($e_s \xrightarrow{r_1} e_1 \xrightarrow{r_2} a$, denoted by -2H) or three hops ($e_s \xrightarrow{r_1} e_1 \xrightarrow{r_2} e_2 \xrightarrow{r_3} a$, denoted by -3H) and then generated natural language questions with templates. To make the generated questions analogical to real-world questions, we included paraphrasing templates and synonyms for relations by searching the Internet and two real-world datasets, WebQuestions (Berant et al., 2013) and WikiAnswers (Fader et al., 2013). In this way, the syntactic structure and surface wording of the generated questions have been greatly enriched.

PQL is more challenging than PQ in that PQL utilizes larger KB and provides less training instances. The statistics are shown in Table 1 and more details are described in the Appendix 6.

4.2 WorldCup2014

We also evaluated our model on the WorldCup2014 (WC2014) dataset constructed by (Zhang et al., 2016). The dataset contains single-relation questions (denoted by WC-1H), two-hop path questions (WC-2H), and conjunctive questions (WC-C). WC-M is the mixture of WC-1H and WC-2H. The statistics of *WorldCup2014* are listed in Table 1.

[5]The question mark indicates the entity to be predicted and the entity in bracket is the expected answer.

[6]This dataset is available at `https://github.com/zmtkeke/IRN`

Dataset	#Entity	#Relation	#Question	Exemplar Question
PQ-2H / 3H	2,215	14	1,908 / 5,198	What does the son of princess_Sophia's mom do for a living?
PQL-2H / 3H	5,035	364	1,594 / 1,031	What is the notable type of Jody_Harris's profession?
WC-1H / 2H / M / C	1,127	6	6,482 / 1,472 / 7,954 / 2,208	Name a player who plays at Forward position from Mexico?

Table 1: Statistics and exemplar questions of PathQuestion (PQ), PathQuestion-Large (PQL) and WorldCup2014 (WC).

5 Experiment and Evaluation

5.1 Implementation Details

ADAM optimizer (Kingma and Ba, 2015) was used for parameter optimization. The dimension of all the embeddings (words in question, entities and relations in KB) was set as $d_x = d_e = d_r = 50$. The hyper-parameter λ (see Eq. 8) is set to 1 . We partitioned the entire dataset into the train/valid/test subset with a proportion of $8:1:1$ and set the batch size as 50.

5.2 Performance of Question Answering

In this section, we evaluated the performance of multi-relation question answering on **PathQuestion** and **WorldCup2014** respectively. To further show that IRN is able to handle more challenging datasets, we evaluated the model with two configurations:

Incomplete KB To simulate the real KBs which are often far from complete, we removed half of the triples (entities and relations are retained but the connections between entities were removed) from the KB of the PQ-2H dataset.

Unseen KB To simulate a real QA scenario where out-of-vocabulary(OOV) words is one of the major challenges, we removed questions whose answer path includes relation *"Cause_of_Death"*, *"Gender"* or *"Profession"* from the PQ-2H training set. The models need to cope with questions related to these three OOV relations during the test.

Several baselines are included here:

Embed (Bordes et al., 2014b) deals with factoid QA over KB by matching a question with an answer in the embedding spaces.

Subgraph (Bordes et al., 2014a) improves the *Embed* model by enriching the representation of an answer entity with the entity's subgraph.

Seq2Seq (Sutskever et al., 2014) is a simplified seq2seq semantic parsing model, which adopts an LSTM to encode the input question sequence and another LSTM to decode the answer path.

MemN2N (Sukhbaatar et al., 2015) is an end-to-end memory network that can be used for reading comprehension and question answering. The memory units consist of the related triples in a local subgraph of the corresponding answer path, where the settings are the same as (Bordes et al., 2015).

KVMemN2N (Miller et al., 2016) improves the MemN2N for KBQA as it divides the memory into two parts: the key memory stores the head entity and relation while the value memory stores the tail entity.

IRN-weak is our model that employs only supervision from the final answer entity rather than the complete answer path. This can be implemented by simply ignoring the loss from the intermediate hops except the final entity in Eq. 8.

The performance is measured by accuracy[7]: correct if a predicted entity is in the answer set of input question. Since there are many 1-to-many relations in Freebase and WC2014, a question may have several possible answer paths, resulting in multiple answers. For example, given the question *"How old is Obama's daughter?"*, the original path can be *"Barack_Obama* $\xrightarrow{Children}$ *Malia_Obama* $\xrightarrow{Age}$ *18"* or *"Barack_Obama* $\xrightarrow{Children}$ *Sasha_Obama* $\xrightarrow{Age}$ *14"*, thus the answer can be either *"18"* or *"14"*. For this question, either answer is correct.

The results in Table 2 demonstrate that our system outperforms the baselines on single-relation questions (WC-1H), 2-hop-relation questions (PQ-2H/PQL-2H/WC-2H) as well as 3-hop-relation questions (PQ-3H/PQL-3H). Furthermore, assembled IRNs obtain strong performance when dealing with conjunctive questions in WC-C .

[7] Reported accuracy is the average accuracy of five repeated runs.

	PathQuestion		PathQuestion Large		WorldCup2014				Challenging PQ-2H	
	PQ-2H	PQ-3H	PQL-2H	PQL-3H	WC-1H	WC-2H	WC-M	WC-C	Incomplete	Unseen
Random	0.151	0.104	0.021	0.015	0.085	0.064	0.053	0.073	-	-
Embed	0.787	0.483	0.425	0.225	0.448	0.588	0.518	0.642	-	-
Subgraph	0.744	0.506	0.500	0.213	0.448	0.507	0.513	0.692	-	-
IRN-weak(*Ours*)	0.919	0.833	0.630	0.618	0.749	0.921	0.786	0.837	-	-
Seq2Seq	0.899	0.770	0.719	0.647	0.537	0.548	0.538	0.577	-	-
MemN2N	0.930	0.845	0.690	0.617	0.854	0.915	**0.907**	0.733	0.899($\downarrow$ 3.3%)	0.558
KVMemN2N	0.937	**0.879**	0.722	0.674	**0.870**	0.928	0.905	0.788	0.902($\downarrow$ 3.7%)	0.554
IRN(*Ours*)	**0.960**	0.877	**0.725**	**0.710**	0.843	**0.981**	**0.907**	**0.910**	0.937($\downarrow$ 2.3%)	0.550

Table 2: Accuracy on different QA datasets. WC-C is for conjunctive questions while other datasets for path questions. Challenging PQ-2H are two more difficult configurations of PQ-2H. The models in the second block utilize the answer path information but those in the first block do not.

We have further observations as follows:

• *IRN-weak* outperforms *Embed* and *Subgraph*, indicating that multi-hop reasoning indeed helps to answer complex questions even when our model is trained end-to-end in the same configuration of weak supervision[8].

• The *Seq2Seq* baseline performs worse than *IRN*. Though they are both interpretable, *IRN* is more powerful when dealing with complicated KBs and questions.

• *IRN* is better than *MemN2N* and *KVMemN2N* on most of the datasets, and both models are much better than other baselines using the path information. Note that the memory in *(KV)MemN2N* consists of fact triples which are distilled from KB using answer path. In this sense, *(KV)MemN2N* indirectly employs strong supervision from answer path. In contrast, *IRN* has a better (or easier) mechanism to supervise the reasoning process thanks to its interpretable framework.

• The highest accuracy on PQL-2H/3H reveals that *IRN* performs better when faced with larger datasets. IRN deals with relations and entities separately, where the number of entities and relations is much less than that of triples. However, *(KV)MemN2N* has to handle much more triples in its memory.

• *IRN* is more robust than the baselines ($\downarrow 3.7\%$ vs. $\downarrow 2.3\%$) when dealing with incomplete KB, which is probably because auxiliary KB embedding training facilitates the prediction of missing triples. While the baselines are more sensitive to the incomplete information stored in the memory units.

• Both *IRN* and the baselines degrade remarkably (0.9→0.5) in the unseen setting because wrong distributed representations are influential in embedding-based QA models. In addition, the size of the training set is much smaller than that of the original PQ-2H, which also leads to worse performance.

• *IRN* is more interpretable compared with *(KV)MemN2N* , attributed to the structure of *IRN*. The relation/entity predicted at each hop is a part of the answer path. The intermediate outputs offer the possibility to trace the complete reasoning process, diagnose failures, and manipulate answer prediction through intermediate interactions (see Section 5.3).

5.3 Interpretable Path Reasoning

In this section, we demonstrated how IRN is interpretable by both quantitative and qualitative analysis. For the quantitative analysis, we can measure how it performs during the reasoning process by investigating the prediction accuracy of intermediate relations and entities. In this task, we collected all the relations and entities with largest probabilities (see Eq. 3 and Eq. 7) at each hop $\{r^1, a^1, r^2, .., a^H\}$ and compared these intermediate outputs with the ground truth $\{r_1, e_1, r_2, .., a\}$. As for KVMemN2N, we also fetched an entity at each hop by an answer distribution, similar to that at the final hop.

According to the structure of IRN, the relation/entity predicted at each hop constitutes an answer path. Results[9] in Table 3 indicate that IRN can predict intermediate entities more accurately than final answers, due to the cascading errors in the consecutive prediction.

Though KVMemN2N (KVM) predicts the exact answers well, it lacks interpretability. On the one hand, KVM can not predict relations to trace the answer path. On the other hand, the hops in KVM all

[8]Only supervision from question-answer pairs, but without answer path information from KB.

[9]Note here that only if an output matches the labeled entity exactly, the prediction will be judged as correct.Thus, the accuracy here has a different definition from that in Table 2.

	r_1		e_1		r_2		e_2		r_3		a	
	IRN	KVM	IRN	KVM	IRN	KVM	IRN	KVM	IRN	KVM	IRN	KVM
PQ-2H	1.000	NA	0.957	0.016	1.000	NA	NA	NA	NA	NA	0.934	0.916
PQL-2H	0.968	NA	0.722	0.083	0.836	NA	NA	NA	NA	NA	0.673	0.676
WC-2H	1.000	NA	0.531	0.000	1.000	NA	NA	NA	NA	NA	0.528	0.382
PQ-3H	1.000	NA	0.883	0.003	1.000	NA	0.772	0.001	1.000	NA	0.738	0.774
PQL-3H	0.808	NA	0.721	0.019	0.702	NA	0.721	0.007	0.683	NA	0.608	0.600

Table 3: Accuracy at different hops along the answer path from IRN and KVMemN2N (KVM). r_i indicates the relation at hop i, e_i indicates the entity at hop i. a indicates the final answer. NA means not applicable.

aim at finding the answer entity rather than the intermediate entities along the answer path.

To illustrate how our model parses a question and predicts relations hop-by-hop, we studied the distributions over all the relations ($\boldsymbol{g}^h$, see Eq. 3) and chose an example from PathQuestion as shown in Figure 3. It is clear that IRN is able to derive the relations in correct order. For question *"What does john_hays_hammond's kid do for a living?"*, IRN first detects relation *Children* (the corresponding word/phrase in the question is *kid*) and then *Profession* (*what does..do*). When detecting the *Terminal* relation, IRN will stop the analysis process.

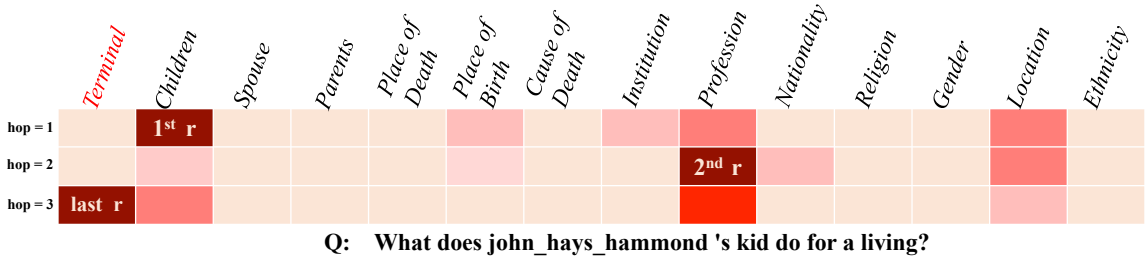

Figure 3: The predicted relations at each hop. Each row represents a probability distribution over relations. Darker color indicates larger probability. The terminal relation is highlighted in red.

Our model can map relations in KB to words in question. We sampled some relations in KB and projected them to the question space by $\boldsymbol{r}^q = \boldsymbol{M}_{rq}\boldsymbol{r}$ (see Eq. 2). We then obtained words whose embeddings are most close to $\boldsymbol{r}^q$ measured by cosine similarity $cos(\boldsymbol{r}^q, \boldsymbol{x}_i)$. The result in Table 4 indicates that IRN can establish reasonable mapping between KB relation and natural language, such as linking *Profession* to words like *working, profession*, and *occupation* . Besides mapping to single words, relation in KB can be associated with some complicate templates, such as *Profession* $\rightarrow$ *"what does ... do"* .

Relation	Similar words in natural language questions
Profession	profession, do, working, occupation
Insititution	institution, organization, work, where
Religion	faith, religion, what, belief
Cause_of_Death	died, killed, how, death
Place_of_Birth	hometown, born, city, birthplace

Table 4: Most similar words in questions for some exemplar relations.

The above analysis demonstrates that our model is interpretable. Specifically, IRN has merits at:
•Providing a traceable reasoning path for question answering. With the aid of these intermediate entities and relations, we can obtain not only the final answer but also the complete path that infers the answer.

•Facilitating failure diagnosis. For instance, IRN fails to answer the question *"Where did the child of Joseph_P_Kennedy_Sr die ?"*. The true answer path should be *"Joseph_P_Kennedy_Sr* $\xrightarrow{Children}$ *Patricia_kennedy_Lawford* $\xrightarrow{Place\_of\_Death}$ *New_York_County"*. However, the middle entity decided by IRN is *"Rosemary_Kennedy"* who is also a child of *"Joseph_P_Kennedy_Sr"*, but her death is not included in KB.

•Allowing manual manipulation in answer prediction. We updated the state (Eq. 5) and the question (Eq. 2) in IRN with the ground-truth relation vectors and compared the performance. The higher

accuracy in Table 5 implies that we can improve the final prediction by correcting intermediate predictions.

Dataset	PQ-2H	PQ-3H	PQL-2H	PQL-3H
Acc	0.980 (↑ 2.0%)	0.900 (↑ 2.3%)	0.755 (↑ 3.0%)	0.744 (↑ 3.4%)

Table 5: Accuracy when the intermediate predictions are replaced by ground truth.

6 Conclusion

We present a novel Interpretable Reasoning Network which is able to make reasoning hop-by-hop and then answer multi-relation questions. Our model is interpretable in that the intermediate predictions of entities and relations are traceable and the complete reasoning path is observable. This property enables our model to facilitate reasoning analysis, failure diagnosis, and manual manipulation in answer prediction. Results on two QA datasets demonstrate the effectiveness of the model on multi-relation question answering.

As future work, there is much room for complex question answering. For instance, answering *"How old is Obama's younger daughter?"* needs to handle arithmetic operation. Furthermore, multi-constraint questions will also be considered in this framework.

Acknowledgements

This work was partly supported by the National Science Foundation of China under grant No.61272227/61332007 and the National Basic Research Program (973 Program) under grant No. 2013CB329403.

References

Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2017. Quint: Interpretable question answering over knowledge bases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 61–66. Association for Computational Linguistics.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Klein Dan. 2015. Neural module networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 39–48.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Learning to compose neural networks for question answering. *NAACL*, pages 1545–1554.

J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic parsing on freebase from question-answer pairs. *Empirical Methods in Natural Language Processing*, pages 1533–1544.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, Bc, Canada, June*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Annual Conference on Neural Information Processing Systems*, pages 2787–2795.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014a. Question answering with subgraph embeddings. *Empirical Methods in Natural Language Processing*, pages 615–620.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014b. Open question answering with weakly supervised embedding models. *In Proceedings of ECML-PKDD*, pages 165–180.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *CoRR*, abs/1506.02075.

Asli Celikyilmaz, Li Deng, Lihong Li, and Chong Wang. 2017. Scaffolding networks for teaching and learning to comprehend.

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

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Meeting of the Association for Computational Linguistics*, pages 1608–1618.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. *Empirical Methods in Natural Language Processing*, pages 318–327.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *international conference on learning representations*.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *International Conference on Machine Learning*, pages 1378–1387.

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

Lili Mou, Zhengdong Lu, Hang Li, and Zhi Jin. 2017. Coupling distributed and symbolic execution for natural language queries. In *Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia, 6-11 August*, pages 2518–2526.

Arvind Neelakantan, Quoc V Le, and Ilya Sutskever. 2015. Neural programmer: Inducing latent programs with gradient descent. *International Conference on Learning Representations*.

Arvind Neelakantan, Quoc V Le, and Ilya Sutskever. 2016. Neural programmer: Inducing latent programs with gradient descent. *international conference on learning representations*.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. *Association for Computational Linguistics*, pages 1470–1480.

Denis Savenkov and Eugene Agichtein. 2017. Evinets: Neural networks for combining evidence signals for factoid question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 299–304. Association for Computational Linguistics.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. 2016. Reasonet: Learning to stop reading in machine comprehension. *SIGKDD*, pages 1047–1055.

R. Socher, D. Chen, C. D. Manning, and A. Y. Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *International Conference on Intelligent Control & Information Processing*, pages 926–934.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. *Annual Conference on Neural Information Processing Systems*, pages 2440–2448.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Annual Conference on Neural Information Processing Systems*, 4:3104–3112.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 189–198. Association for Computational Linguistics.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. *International Conference on Learning Representations*.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2326–2336.

Semih Yavuz, Izzeddin Gur, Yu Su, and Xifeng Yan. 2017. Recovering question answering errors via query revision. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 903–909. Association for Computational Linguistics.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic parsing for single-relation question answering. In *Proceedings of Association for Computational Linguistics*, pages 643–648.

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

Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. 2015. Neural enquirer: Learning to query tables with natural language. *Association for Computational Linguistics*, pages 2308–2314.

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schtze. 2016. Simple question answering by attentive convolutional neural network. *International Conference on Computational Linguistics*, pages 1746–1756.

Liwen Zhang, John Winn, and Ryota Tomioka. 2016. Gaussian attention model and its application to knowledge base embedding and question answering. *CoRR*, abs/1611.02266.

Appendix A. PathQuestion Construction and Question Templates

We constructed a synthesis dataset by generating questions with templates. The knowledge base for PathQuestion has more than 60,000 triples which are adopted from FB13 (Socher et al., 2013) with 13 relations and thousands of entities. As for PathQuestion-Large, we adopted another more complex subset of Freebase (Bollacker et al., 2008). **First**, we extracted all the paths with two hops ($< e_s, r_1, e_1, r_2, a >$), or three hops ($< e_s, r_1, e_1, r_2, e_2, r_3, a >$) among these triples. **Second**, we crafted templates to generate natural language questions from these paths. **Last**, we collected question and answer path pairs $(q, < e_s, r_1, e_1, ..., a >)$ to construct the *PathQuestion (PQ)* dataset.

We crafted templates to transfer an answer path extracted from KB to natural language questions. To make the generated questions analogical to real-world questions, those templates are firstly written manually, and then enriched by replacing synonyms. Besides, we searched for different syntactical structures and paraphrases in real-world datasets including WebQuestions (Berant et al., 2013) and WikiAnswers (Fader et al., 2013) as well as on the Internet. In this manner, the templates have been greatly diversified and are much closer to real questions.

Synonyms used in templates for PathQuestion are shown in Table 6 and templates for 2-hop paths (PQ-2H) are shown in Table 7. The datasets are available at `https://github.com/zmtkeke/IRN`.

Relation	**Synonyms**
Spouse	couple, wife, husband
	other half, darling
Children	child, offspring, kid
	daughter, son, heir
Parents	parent, father, mother
	dad, mom
Profession	job, occupation, work
Institution	organization
	educational institution
Ethnicity	race
Gender	sex
Nationality	nation, country
Location	address
Religion	faith, religious belief
	type of religion

Table 6: Natural language synonyms for relations appearing in questions in PathQuestion.

Path Pattern	**Question-templates**
Universal	"What is the r_2 of e_s's r_1?"
	"What is the e_s's r_1's r_2?"
	"What is the r_2 of r_1 of e_s?"
	"The r_2 of e_s's r_1?"
	"The e_s's r_1's r_2?"
	"The r_2 of r_1 of e_s?"
Ask about a person	"Who is the r_2 of e_s's r_1?"
	"What is the name of the r_2 of e_s's r_1?"
r_2=r_1=Parents/ r_2=r_1=Children	"Who is the grand-r_1 of e_s?"
	"What is the name of the grand-r_1 of e_s?"
r_2=Ethnicity	"What r_2 is e_s's r_1"
	"What is e_s's r_1's r_2 like?"
	"What is e_s's r_1's r_2 about?"
r_2=Institution	"Where does e_s's r_1 work?"
	"Where does e_s's r_1 work for?"
	"Which r_2 does e_s's r_1 work for?"
r_2=Nationality	"Which nationality is e_s's r_1?"
	"Where does e_s's r_1 come from?"
r_2=Religion	"What r_2 does e_s's r_1 follow?"
	"What r_2 is e_s's r_1?"
	"What r_2 does e_s's r_1 have?"
	"What r_2 is e_s's r_1 practice?"
r_2=Gender	"What r_2 is e_s's r_1?"
	"Is e_s's r_1 a man or a woman?"
r_2=Location	"Where is e_s's r_1 living?"
	"Where is e_s's r_1 staying?"
	"Please tell me e_s's r_1 present address."
r_2=Profession	"What does e_s's r_1 do?"
	"What is e_s's r_1 working on?"
	"What is e_s's r_1?"
	"What line of business is e_s's r_1 in?"
	"What does e_s's r_1 do for a living?"
r_2=Cause_of_Death	"Why e_s's r_1 died?"
	"How e_s died?"
	"What's the reason of e_s's r_1's death?"
	"What caused the death of e_s's r_1?"
	"What killed the e_s's r_1?"
	"What made the e_s's r_1 dead?"
	"What did e_s's r_1 die from?"
r_2=Place_of_Death	"Where did e_s's r_1 die?"
	"Where did the r_1 of e_s die?"
	"What city did e_s's r_1 die?"
r_2=Place_of_Birth	"Where did e_s's r_1 born?"
	"What city did e_s's r_1 born?"
	"What is the hometown of e_s's r_1?"
	"What is e_s's r_1's birthplace?"

Table 7: Templates for generating natural language questions from answer paths in PQ-2H.