An Interpretable Reasoning Network for Multi-Relation Question Answering

Mantong Zhou, Minlie Huang, Xiaoyan Zhu
State Key Laboratory of Intelligent Technology and Systems,
National Laboratory 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.

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.

As argued by Yin et al. [2016], question answering over knowledge bases falls into two types, namely single-relation QA and multi-relation QA. 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; Bordes et al., 2014b; Yih et al., 2014]. 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 with 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 [Barkan et al., 2013; Yih et al., 2014] 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; Bordes et al., 2014a] 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 capability.

Recent reasoning models [Weston et al., 2015; Sukhbaatar et al., 2015] 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 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 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 complex questions with multiple fact 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 hop-by-hop reasoning process can make intermediate predictions, and thus provides traceable results for the predicted entities and relations.

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; Berant et al., 2013; Kwiatkowski et al., 2013; Yih et al., 2014; Yang et al., 2014]. 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 which are recently introduced into the QA community [Bordes et al., 2014b] where questions and answers are represented by distributed vectors with neural networks [Yin et al., 2016; Dong et al., 2015], and QA is formulated as a problem of matching between questions and answers. 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] 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] make interactions between a question and the corresponding document in a multi-hop manner during reasoning. MemNN [Weston et al., 2015] and KVMemN2N [Miller et al., 2016] 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] 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 [Neelakantan et al., 2015] needs predefined symbolic operations hence may not be suited for other operations, and Neural Enquirer [Yin et al., 2015] lacks explicit interpretation.

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 $e$, 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 $e$), 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 $e_1 \rightarrow r_1 \rightarrow e_2 \rightarrow r_2 \rightarrow \ldots \rightarrow a$. Relations ($r_1$) are observable (in natural language form) in the question, however, the intermediate entities ($e_1 \cdots e_{11}$) are not. For example, for question “How old is Obama’s daughter?”, the subject is Barack Obama and the answer path is Barack Obama $\rightarrow$ Children $\rightarrow$ Malia Obama $\rightarrow$ Age $\rightarrow$ 18. Note that since there are 1-to-many relations, 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: FORWARD $\rightarrow$ plays_position$^{-1}$ $\rightarrow$ Marco Reus and Borussia Dortmund $\rightarrow$ plays_in_club$^{-1}$ $\rightarrow$ 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 word “daughter”. This procedure is repeated until the Terminal relation is predicted.

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 \( \hat{r}^h \), obtains a predicted relation \( r^h \) and intermediate state \( s^h \). The predicted relation \( \hat{r}^h \) is used to update \( q^h \) and \( s^h \) is updated to state \( s^{h+1} \) which is initialized by subject \( e_x \).

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, given a question \( X = x_1, x_2, ..., x_n \), the question 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:

\[
q^0 = \sum_{i=1}^{n} x_i \tag{1}
\]

\[
q^h = q^{h-1} - M_{rq} \hat{r}^h \tag{2}
\]

where \( M_{rq} \) is a transition matrix projecting the KB relation space to the natural language question space, \( q^{h-1} \) is the question representation at hop \( h-1 \), and \( \hat{r}^h \) defined by Eq. 4 is the predicted relation at hop \( h \). The intuition of such update is that the already analyzed 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 \( (s^{h-1}) \) and the previous representation of the question \( (q^{h-1}) \), and then predicts a relation \( (\hat{r}^h) \) based on the analysis at the current hop. Once the predicted relation \( (\hat{r}^h) \) is obtained, the relation will be used to update the next question representation \( (q^h) \) and the state of the reasoning module \( (s^h) \). In this manner, the reasoning network is traceable and interpretable.

The process can be formally described by the following equations:

\[
g^h_j = P(r^h = r_j | \hat{r}^{h-1}, s^{h-1}) = \text{softmax}(M_{rq} \hat{r}^{h-1} + (M_{rs}r_j)^T s^{h-1}) \tag{3}
\]

\[
\hat{r}^h = \text{argmax} g^h_j \tag{4}
\]

\[
s^h = M_{rs} \hat{r}^h \tag{5}
\]

\[
s^{h+1} = s^{h-1} + \hat{s}^h \tag{6}
\]

where \( r_j \) is the embedding vector of a relation in KB and all the relation embeddings are stored in a static memory \( R \), and \( s^h \) is the state of the reasoning module. \( g^h_j \) is the probability of selecting the \( j^{th} \) relation in KB and \( M_{rs} \) is the projection matrix mapping \( r \) from the relation space to the state space. \( M_{rq} \) is the same projection matrix used in Eq.3 to map \( r \) from the relation space to the question space. In implementation, we use the Gumbel-softmax [Jang et al., 2017] to replace the softmax+argmax operation to make the model differentiable.

We initialize the state vector with the topic entity (subject) \( s^0 = e_x \). IRN will learn to enrich the state representation hop
by hop, for instance, at the first hop $s^1 = e_s + r_1$, and at the second hop $s^2 = e_s + r_1 + r_2$, intuitively. In this manner, the state vector encodes historical information.

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 $s$.

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:

$$e^h = M_{se} s^h$$

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

$M_{se}$ is used to transfer from the state space ($s^h$) to the entity space ($e^h$) to bridge the representation gap between the two spaces. $e_j$ 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:

$$L = \sum_h C_r(h) + \lambda C_o(h)$$

$$C_r(h) = -\sum_{j=1}^{n_r} [\hat{g}_j^h \ln g_j^h]$$

$$C_o(h) = -\sum_{i=1}^{n_e} [\hat{o}_i^h \ln o_i^h]$$

where $n_r/n_e$ is the number of relations/entities in KB respectively. $\hat{g}_j^h$ is the gold distribution (one-hot) over relations at hop $h$. $g_j^h$ is the predicted distribution defined by Eq. (3). $\hat{o}$ is the gold distribution over entities, which is also one-hot representation, and $o$ is defined by Eq. (8). $\lambda$ is a hyper parameter to balance the two terms.

Note that the training data is in the form of $(q, < e_s, r_1, ..., a >)$, 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 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:

$$M_{se}(e_s + r) = e_o$$

where $e_s, r, e_o$ 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 $M_{se}$ (see Eq. (7)) as a transfer matrix to bridge the representation gap between the state space (here $e_s + r = s$) and the entity space (here $e_o = e$).

The parameters are updated with a multi-task training schema. We first learn the KB embeddings $e/r$ and the transformation matrix $M_{se}$ to fit Eq. (12) with several epochs. 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, and the process terminates after 1000 iterations.

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 $M_{se}(e_{Barack Obama} + r_{Children}) \approx e_{Malia Obama}$.

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→plays.position−1→?(Marco_Reus)”3. The output is a distribution over entities. Similarly, another IRN (IRN_2) tackles the path query “BD→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 datasets to evaluate our Interpretable Reasoning Network: one is PathQuestion, constructed by ourselves and the other is WorldCup2014, adopted from Zhang et al., 2016.

4.1 PathQuestion

We constructed a synthesis dataset by generating questions with templates. The knowledge base for PathQuestion-2/3-Hops has more than 60,000 triples which are adopted from FB13 [Scher et al., 2013] with 13 relations and thousands of entities. First, we extracted all the paths with two hops (< e_s, r1, e1, r2, a >), or three hops (< e_s, r1, e1, r2, e2, r3, a >) among these triples. Second, we crafted templates to generate natural language questions from these paths. To make the generated questions analogous 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. Last, we collected question and answer path pairs (q, < e_s, r1, e1, ..., a >) to construct the PathQuestion (PQ) dataset whose statistics are shown in Table 1.

| Dataset    | #Question | Dataset    | #Question |
|------------|-----------|------------|-----------|
| PQ-2H-V1   | 1,908     | WC-1H      | 6,482     |
| PQ-2H-V2   | 3,554     | WC-2H      | 1,472     |
| PQ-3H-V1   | 1,434     | WC-M       | 7,954     |
| PQ-3H-V2   | 7,797     | WC-C       | 2,208     |

Table 1: Statistics of PathQuestion (PQ) and WorldCup2014 (WC).

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. All the datasets are based on a KB that has 1,127 entities and 6 relations. Statistics of WorldCup2014 are listed in Table 1.

5 Experiment and Evaluation

5.1 Implementation Details

ADAM optimizer [Kingma and Ba, 2015] is used for parameter optimization with an initial learning rate of lr = 0.001, β1 = 0.9 and β2 = 0.999. The dimension of all the embeddings (words in questions, entities and relations in KB) is set as d_e = d_c = d_r = 50. Moreover, we normalized the state vector and the embeddings by constraining ∥s∥2 = ∥r_j∥2 = ∥e_i∥2 = 1. The hyper-parameter λ (see Eq. 9) is set to 1. We partitioned the entire dataset into the train/valid/test subset with a proportion of 8 : 1 : 1. We set the batch size as 50 and all models are trained with at most 1,000 epochs.

5.2 Performance of Question Answering

In this section, we evaluated the performance of multi-relation question answering on PathQuestion and WorldCup2014 respectively. 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 answer path and the entity’s subgraph. 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. 9.

The performance is measured by accuracy: correct if a predicted entity is in the answer set of input question. Since there are many 1-to-many relations in FB13 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 → Children → Malia Obama → Age → 18” or “Barack Obama → Children → Sasha Obama → 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/WC-2H) as well as 3-hop-relation questions (PQ-3H). Furthermore, assembled IRNs obtain strong performance when dealing with conjunctive questions in WC-C (the last column).

We have further observations as follows:

1. Our model can be trained end-to-end and outperforms baselines in the same configuration of weak supervision (only supervision from question-answer pairs with-
IRN-weak outperforms Embed and Subgraph, indicating that multi-hop reasoning does help to answer complex questions.

2. IRN is better than MemN2N and KV/MemN2N on most datasets, and both models are much better than other baselines with path information. Note that KV/MemN2N also utilizes the answer path by distilling fact triples from KB. In this sense, KV/MemN2N indirectly employs strong supervision from answer path. In comparison, IRN has a better (or easier) mechanism to supervise the reasoning process thanks to its interpretable framework.

3. IRN is more traceable compared with KV/MemN2N. Owing to the structure of IRN, every relation/entity predicted at each hop is a part of the answer path. By probing the intermediate outputs, we can trace the reasoning process and diagnose failures more easily (see later section).

5.3 Scalability to More Challenging Settings

To further show that IRN is able to handle larger, incomplete, and unseen knowledge bases, we evaluated the model with three challenging configurations:

- Large KB To evaluate the capacity of handling large KBs, we adopted a subset of Freebase [Bollacker et al., 2008] to construct the PathQuestion-Large dataset which includes about 400 relations and 6,000 entities.
- 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 broken) from the KB of the PQ-2H-V1 dataset.
- Unseen KB We selected 3 relations as target relations. Training pairs \(q, < e_s, r_1, ..., a >\) whose answer path involves those target relations are removed from the training set of PQ-2H-V1.

### Table 3: Accuracy on three challenging datasets

| Model        | Large     | Incomplete | Unseen   |
|--------------|-----------|------------|----------|
| MemN2N      | 0.722     | 0.899 (3.3%) | 0.558    |
| KV/MemN2N   | 0.690     | 0.902 (3.7%) | 0.554    |
| IRN-weak    | 0.600     | 0.901 (1.9%) | 0.539    |
| IRN         | **0.725** | **0.937 (2.3%)** | **0.550** |

Results with large KB verify that IRN can deal with larger datasets and also outperforms baselines. The next column indicates that IRN is more robust than baselines when dealing with incomplete KB, which is probably because auxiliary KB embedding training facilitates the prediction of missing triples. While baselines are more sensitive to the information stored in the memory units. The unseen setting simulates a real QA scenario where out-of-vocabulary words is one of the major challenges. Both IRN and baselines are decreased obviously because wrong distributed representations are influential in embedding-based QA models. Besides, the size of training set is much smaller than original PQ-2H-V1, that also leads to worse performance.

5.4 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 entities. In this task, IRN is trained on question-path pairs such as \(q, < e_s, r_1, e_1, r_2, ..., a >\) but at the test stage, we collected all the outputs from the answer module at each hop \(\{a^1, a^2, ..., a^h\}\) and compared these intermediate outputs with the ground truth \(\{e_1, e_2, ..., a\}\).

### Table 4: Accuracy at different hops along the answer path.

| Dataset | \(e_1\) | \(e_2\) | \(a\) |
|---------|---------|---------|-------|
| PQ-2H-V1 | 0.957  | NA     | 0.934 |
| PQ-2H-V2 | 0.871  | NA     | 0.769 |
| PQ-3H-V1 | 0.965  | 0.868  | 0.764 |
| PQ-3H-V2 | 0.883  | 0.772  | 0.738 |
| WC-2H    | 0.531  | NA     | 0.528 |

Results are shown in Table 4. 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. Results on all the path question datasets indicate that IRN can predict intermediate entities more accurately than final answers, and that cascading errors degrade the final predictions.

Qualitative analysis is conducted with case studies. To illustrate how our model parses a question and predicts relations hop-by-hop, we studied the distributions over all relations \(\{g^i\}\), 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...
Q: What does john_hays_hammond’s kid do for a living?

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.

| hop = 1 | hop = 2 | hop = 3 |
|---------|---------|---------|
| Terminal | Children | Place of Death |
|          |          | Place of Birth |
|          |          | Cause of Death |
|          |          | Institution |
|          |          | Profession |
|          |          | Nationality |
|          |          | Religion |
|          |          | Gender |
|          |          | Location |
|          |          | Ethnicity |

Table 5: Most similar words in question for some exemplar relations.

| Relation          | Similar words in natural-language questions          |
|-------------------|-----------------------------------------------------|
| Profession        | profession, do, working, occupation                 |
| Institution       | institution, organization, work, where              |
| Religion          | faith, religion, what, belief                       |
| Nationality       | nationality, come, george_herbert, nation           |
| Location          | kikkawa_motoharu, living, location, address, staying |
| Cause_of_Death    | died, killed, how, death                            |
| Place_of_Birth    | hometown, born, city, birthplace                    |

6 Conclusion

We propose 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 and failure diagnosis. Results on two QA datasets demonstrate the effectiveness of the model on question answering.

As future work, the model will be improved to handle more complicated situations where arithmetic operation or commonsense reasoning is needed, like “How old is Obama’s younger daughter?” Furthermore, multi-constraint questions will also be considered in this framework.

References

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

[Bollacker et al., 2008] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 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, 2008.

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

[Bordes et al., 2014a] Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embed-
[Bordes et al., 2014b] Antoine Bordes, Jason Weston, and Nicolas Usunier. Open question answering with weakly supervised embedding models. In Proceedings of ECML-PKDD, Springer Berlin Heidelberg, pages 165–180, 2014.

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

[Dong et al., 2015] Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 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, 2015.

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

[Guu et al., 2015] Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. Conference on Empirical Methods in Natural Language Processing, pages 318–327, 2015.

[Jang et al., 2017] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In ICLR, 2017.

[Kingma and Ba, 2015] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. International conference on learning representations, 2015.

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

[Kwiatkowski et al., 2013] Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke S Zettlemoyer. Scaling semantic parsers with on-the-fly ontology matching. In Association for Computational Linguistics, pages 1545–1556, 2013.

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

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

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

[Shen et al., 2016] Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. Reasonet: Learning to stop reading in machine comprehension. Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 1047–1055, 2016.

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

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

[Weston et al., 2015] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. International Conference on Learning Representations, 2015.

[Xu et al., 2016] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 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, 2016.

[Yang et al., 2014] Minchul Yang, Nan Duan, Ming Zhou, and Haechang Rim. Joint relational embeddings for knowledge-based question answering. Empirical Methods in Natural Language Processing, pages 645–650, 2014.

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

[Yin et al., 2015] Pengcheng Yin, Zhengdong Lu, Hang Li, and Ben Kao. Neural enquirer: Learning to query tables with natural language. Meeting of the Association for Computational Linguistics, pages 2308–2314, 2015.

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

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