Enhancing Question Answering over Knowledge Base Using Dynamical Relation Reasoning

1st Liao Cheng
School of Data and Computer Science
Sun Yat-Sen University
Guangzhou, Guangdong, P.R.China
chengliao@mail2.sysu.edu.cn

2nd Ziheng Chen
School of Data and Computer Science
Sun Yat-Sen University
Guangzhou, Guangdong, P.R.China
chenzh76@mail2.sysu.edu.cn

3rd Jiangtao Ren*
School of Data and Computer Science
Sun Yat-Sen University
Guangzhou, Guangdong, P.R.China
issrjt@mail.sysu.edu.cn

***Abstract*—The task of Knowledge Base Question Answering(KBQA) is to provide a convenient way for the human to more efficiently and easily answer natural language questions using the substantial and valuable knowledge in the KB. Predicate generation is the most important sub-task of KBQA, which aims to generate the predicate paths from head entity to tail entity. Existing models mostly employ a seq2seq method to handle this task. However, the seq2seq method essentially is a classification model, which needs to know the number of the categories in advance. Meanwhile, KBs often are incompleteness and questions are always unbounded, which would cause the problem that the predicates of the questions would be beyond predefined categories. Obviously, the seq2seq model cannot handle this problem. In this paper, to solve the problem above, we explore to build an scoring module with strong genelization to score the predicates not in predefined categories, and also to improve the performance of the seq2seq method.**

To bridge the gap, we carefully design a reasoning module to score the predicates through reasonably employing the attention mechanism with memory ability. Simultaneously, in order to improve the generalization of the reasoning module, we try to use multi-task learning to enrich the represent of the reasoning module based on the idea of transfer learning. Massive experiments are conducted on two popular benchmark datasets—SimpleQuestion(SimQ) and WebQuestion(WebQ). The experimental results demonstrate that the proposed relation reasoning framework outperforms the state-of-the-art methods.

***Index Terms*—KBQA, End-to-End Network, Multi-hop Reasoning, Multi-task Learning**

I. Introduction

The ever-increasing availability of data and knowledge requires significant progress on extracting the wealth information for user. In particular, Knowledge Bases(KB), such as DBpedia [1], and Freebase [2], contain vast amounts of triple facts gained from increasing information. In knowledge base, each directed edge, along with its head entity and tail entity, constitute a triple(i.e.,head entity,predicate,tail entity), which is also named as a fact. Knowledge bases are usually huge and not easily accessible for users as they need to know the query statement as well as the structure and relations in the KB.

Question answering over knowledge base provides a way for artificial intelligence systems to incorporate knowledge bases as a key ingredient to answer human questions, with applications ranging from search engine design to conversational agent building. It targets at automatically translating the end users' natural language(NL) questions into structured queries such as SPARQL [3], and returning tail entities and/or predicate paths in the KB as answers. Existing models [4]–[9] mostly contain entity linking module and relation detection module. Entity linking is responsible for detecting head entities in the question and linking them to KB. Relation detection is responsible for selecting the best from candidate predicates. Recently, some researchers [9] pay attention to the generation of the predication path via using the seq2seq [10] model, which could reduce the number of the candidate paths. However, the KBQA problem is far from solved since the domains of end users' questions are often unbounded, and any KB is far from complete. New questions might involve predicates that are different from the training set, or out of the predefined categories. The seq2seq framework can't handle this scenario since it is limited to fixed known categories and it extremely depends on the pattern of training samples. Recently, embedding methods have achieved great success in many NLP fields. The key idea is to represent each predicate as a low-dimensional vector, such that the relation information in the KB could be preserved. In addition, similar predicates tend to have similar vectors. That motivate us to to design an adaptive module based on embedding methods to score the predicates beyond predefined categories.

However, there still remains two major challenges for building an adaptive module. **First**, a predicate have various different represents. i.e., the represent of KB and NL. If the module just relies on one of these represents, that would cause the problem of the ambiguity of the predicate. For example, the predicate "people.person.nationality" is similar to "people.person.place_of_birth" in KB, since they have similar structure in KB. However, someone's nationality may not be the same as the place of birth. **Second**, the general adaptive modules often don't contain memory ability, e.g., CNN(Convolutional Neural Network) and DNN (Deep Neural Network). However, the generation of the predicate is a reasoning process. The inference of the predicate should fully consider the context of the question. For example, the corrent predicate path of the question "What is the name of Obama's father?" is ("people.person.parents","type.object.name"). Both current knowledge context(CKC, acting as a dynamic environ-

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

ment in KB, the definition in section III) at first hop and second hop have the same predicate ”type.object.name”. If following the traditional adaptive modules, the model would give the same weight to the predicate ”type.object.name” at first hop and second hop. But it is logical that the model should give the weight to the predicate according to the context of the question.

Through analyzing the problems, we aim to solve three research questions in the generation of the predicates. (i) How to construct an adaptive reasoning module to handle the problem that the categories of the predicate is unbound? (ii) How to leverage various embeddings of the predicate to solve the problem that the semantic information of similar structural predicates is different? (iii) How to leverage the reasoning context to make the adaptive module contain reasoning ability? Following these questions, we propose a novel general model with dynamical relation reasoning module. Simultaneously, inspired by the idea of transfer learning, we try to use multi-task learning to enrich the represent of the reasoning module. In conclusion, we highlight the contributions of this paper as follows:

- We construct the CKC acting as dynamic enviroment in KB. The CKC contains the KB/NL information of the predicate, which can improve the performance of the model.
- We design an adaptive reasoning module(the definition is in IV-B2) to solve the three problems raised in this paper, including (i) the categories of the predicate is unfixed, (ii) the question that similar structural predicate is indistinguishable, (iii)and the question that an adaptive module dose not have reasoning ability.
- In order to enchance the generalization of the reasoning module, based on the idea of transfer learning, we try to use multi-task learning to enrich the represent of the reasoning module.

II. Related Work

The approaches proposed to tackle the KBQA task can be roughly categorized into two groups, semantic parsing (SP) and information retrieval (IR).

SP-based methods aim to learn semantic parses which parse natural language question into logical forms and then query knowledge base to lookup answers. Earlier SP-based methods directly parse natural language questions into structured queries [11]. Recently, SP-based methods try to exploit IR-based techniques [7], [12], [13] by matching the same space between the question and semantic parsing tree. However, SP-based approachs more or less depend on hand-crafted rules as supervised information, which causes high labor costs for open-domain question answering.

IR-based methods retrieve a set of candidate answers and then conduct further analysis to rank these answers [4]–[6], [8], [14]–[17]. Most of them focus on mapping answers and questions into the same embedding space, which could query the KB independently according to its schema without any grammar or lexicon. In recent reasearchs, IR-based methods mostly are superior to SP-based methods. In these IR-based methods, there are mainly two kinds of methods based on deep neural networks, ranking methods [4]–[8] and relation generative methods [9]. Both ranking methods and relation generative methods can be described as the following processes, head entity detection, the generation of main predicate paths, filtering predicate paths and returning the tail entity. D ferently, the former filter the whole candidate predicate paths using pre-defined rules, while the latter use seq2seq [10] to generate candidate predicate paths. Bordes et al.(2014) [18] are the first to apply an embedding-based approach for KBQA. Later, Bordes et al.(2014) [19] propose a subgraph embedding method, which encodes more information of the candidate subgraghs. Golub et al.(2016) [16] propose a character-level approach based on encoder-decoder architecture with the attention mechanism. Wang et al.(2018) [9] propose an APVA(entity alignment, path label prediction, verification, object answering) architecture, which train verification and path label prediction alternately. Xu et al.(2018) [8] enrich the question represent using completing-CNN and comparing-CNN. Our method is inspired by Xu et al.(2018) and Wang et al.(2018), but differently, we propose an adaptive module, which solves the scenario that new questions might involve predicates beyond the training samples.

III. Problem Statement

Definition 1 (Current Knowledge Context): *The CKC is a set of relation edges around the entity vertex at each hop in KB. For example, as shown in Figure 1, the first hop CKC is a set of the predicates around the first entity* $e1$ *detected in the question.*

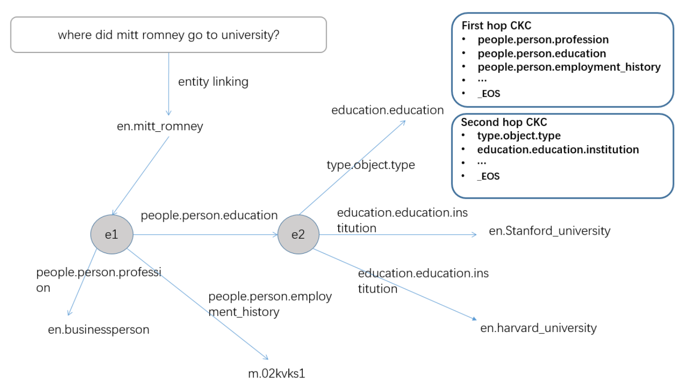

Fig. 1. An example of the multi-hop question. CKC is the set of predicates around entity.

Task Definition: We use (h,p,t) to represent a fact, which means that there exists a predicate p from a head entity h to a tail entity t. Let $\mathcal{K}$ be a knowledge base that consists of a large number of facts. The total numbers of predicates and entities are represented as $\mathcal{P}$ and $\mathcal{E}$. The answer to q may be multiple facts which form a path in the KB. For example, the answer could involve two triples (a,r1,b) and (b,r2,c) concatenated in tandem. This corresponds to the path (a,r1,b,r2,c) in the KB. Denote by $\mathcal{P}^*$ is the set of all finite-length sequences predicates. We still denote an answer, multi-hop or single-hop, as a triple (h,p,t) while considering p$\in \mathcal{P}^*$. Let $\mathcal{Q}$ be a set of questions. Given the conditions described above, we now formally define our problem.

Given a knowledge base $\mathcal{K}$ associated with all its predicates' and entities' names and the embedding representations of KB/NL, as well as a set of questions $\mathcal{Q}$ associated with corresponding head entities and predicate paths, we aim to design a multi-task learning framework that takes a new question as input and automatically returns the corresponding predicate paths and tail entity.

IV. Our Approach

Figure 2 shows the workflow of our framework. We use the freebase as our KB. As usual for traditional models, our framework have three main modules, entity detection, relation generation and relation detection.

Given a natural language question q, we employ the following steps to return the answers. (i) The entity detection module recognizes the alias of the head entity[1], and then performs the alias linking to get the entities in KB. (ii) After getting all entities, the relation generation module would generate/predict a candidate predicate paths using reasoning module on literal-level and semantic-level. (iii) Finally, with the help of the reasoning module, the relation detection returns top-k (h,p,t) triples, which can be transformed to SPARQL query language to get the final answers.

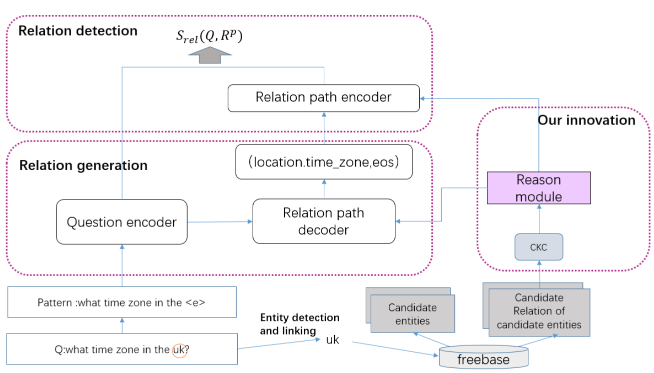

Fig. 2. The overview of the proposed KB-QA system. The detail of reason module is in Figure 3.

A. Entity Detection

Given a question, the goal of the entity detection module is to find all possible head entity names using certain selection strategies. In this part, similar to Qu et al. (2018) [17], our goal is to get all possible head entities. We employ a BIGRU [20] network and treat the entity detection as a sequential binary classification task that decides whether each word in a sentence should belong to entity mention. Getting a set of words with a positive label, we use edit distance algorithm[2] to deal with logogram of some ambiguous entities. Moreover, we use entity linking result[3] to enrich the candidate entity set.

B. Relation Generation

The architecture of this part is shown in Figure 3. This module generates the relation path from the head entity to the tail entity, and mainly contains an encode layer and a decode layer. The encode layer captures the semantic information of the question. The decode layer generates relation paths according to the context representation of the encode layer.

*1) **Encode Layer**:* Given a question Q, this part aims to effectively represent the information of question. As is shown in Figure 3.

First of all, we look up a word embedding matrix $\mathbf{W} \in \mathbb{R}^{|V_w|\times d}$ to convert the original Q into word embeddings $\{q_i^{(tok)}\}$, where $|V_w|$ denotes the vocabulary size of natural language words, and d denotes the embedding dimension. Embedding matrix is initialized using Glove [21](which is a pre-trained file), and it is fine-tuned during the training process. Then we construct a pattern of this question through replacing the keyword with e, and leverage question pattern to represent long-range dependencies between the answer and the head entity. Because the start of the query path in the question is always analogous. We simply treat the question as ascending sequence pattern, and keyword is fixed at 37. For example, as shown in figure 3 for the question "what time zone in the e", keyword subscript is 37, the subscript of the question is (32 33 34 35 36 37). Then the subscript of the question is converted into embeddings $\{q^{(p)}\}$ through the sequence embedding matrix $\mathbf{P} \in \mathbb{R}^{|V_p|\times d}$, where $|V_p|$ denotes the maximum length of the question. Finally we combine two representations as $\{q_i\} = \{q_i^{(tok)}\} + \{q_i^{(p)}\}$, which are fed into a GRU network. We use the output hidden of GRU at final timestep as the reasoning context of first hop l^1, and employ each timestep hidden $\{u_i\}$ to construct similarity matrix in reasoning module.

*2) **Decode Layer**:* Given the represent of encode layer $\{q_i\}$, this layer aim to generate the predicate path via static part and dynamic part. Static part is a classical decode layer of seq2seq model, which records information of the reasoning path to update the context. Dynamic part is a scoring module, which can score the predicate in CKC. Since the predicate is unlimited, it would face the following two scenarios in the prediction of predicates: (i) When the predicates are in predefined categories, it can improve the prediction of the static part via weighted sum. (ii) When the predicates are beyond predefined categories, it directly score the predicate, and then select the best predicate via normalization.

[1] In freebase, each entity has only one identification code. The topic words are the alias of the entity. For example, the alias of the entity "m.05v8c" is "philippines".

[2] The process is [l+1, r+1, l-1, r-1, l+2, r + 2,...], where l+1 means add one word from left side, r-1 means discard one word from right side.

[3] for SimQ, and the entity-linking result can be downloaded from https://github.com/Gorov/SimpleQuestions-EntityLinking; for WebQ, https://github.com/scottyih/STAGG

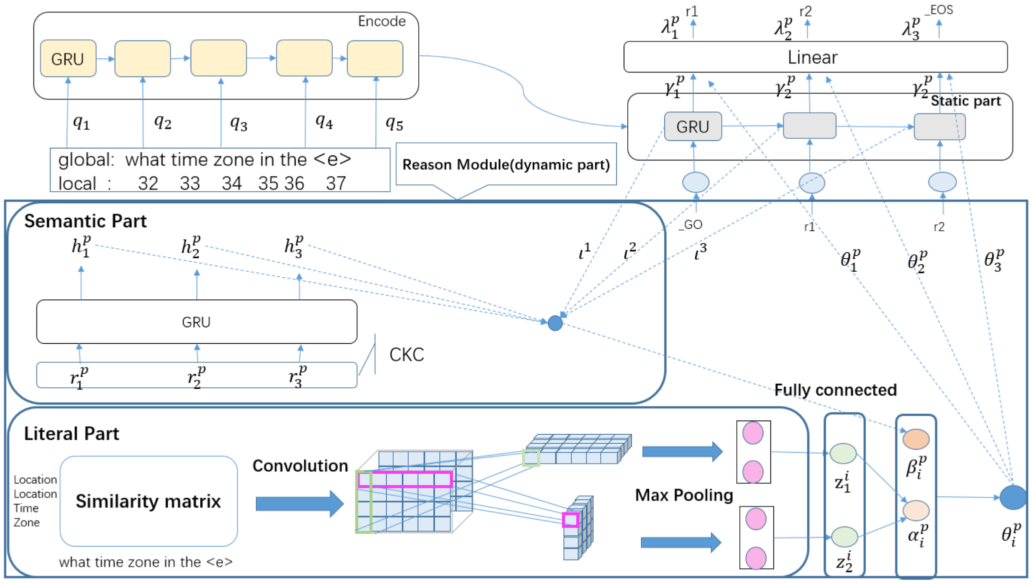

Fig. 3. The overview of relation generation. The core of this task is the reasoning module, which captures the information between question and relation in the CKC on literal-level α and semantic-level β. θ acts as dynamic probability in relation prediction, while acts as weight in relation detection. γ acts as static probability. Final probability of relation generation λ combines dynamic part and static part.

Reasoning Module(Dynamic Part): We aim to answer the three questions raised in this paper using this module. (i) The reasoning module is a component used to calculate the score of the predicate, which is not restricted by predefined predicates' categories. That can solve the question one. (ii) We construct two part in reasoning module from literal part and semantic part, so that reasoning module could understand the information of the predicate from several different perspectives. That can solve the question two. (iii)Instead of employing the gate mechanism, we directly use attention mechanism to make the module contain certain memory ability and reasoning ability. That can solve the question three.

- ***Literal Part***: Similarity matrix could better construct the representation of semantic vector space between question and predicate, and convolutional neural network can capture the features of similarity matrix from row and column. These features support reasoning module to distinguish semantic information of similar predicate.
  Given a question with length L_q and a predicate with length L_p, we first map their tokens into a sequence of word embedding vectors $\{u_i\}$ and $\{v_j\}$, based on a pre-trained embeddings such as GloVe [21]. Then we employ $\{u_i\}$ and $\{v_j\}$ to construct a similarity matrix M as the basic of the channel, and use the CNN network to extract different levels of matching patterns. Finally, we apply the max pooling to amplify the feature both from question to predicate z_1^i and from predicate to question z_2^i. A fully connected layer is then applied to hidden state $[z_1^i; z_2^i]$, its result o_i^p is the score between question and predicate from NL, the corresponding probability is α_i^p . o_i^p is computed via the following equations.

$$\alpha_i^p = \frac{exp(o_i^p)}{\sum_{t=0}^{n^k} exp(o_t^p)} \tag{1}$$

$$o_i^p = W_3[z_1^i; z_2^i] + b \tag{2}$$

$$z_1^i = W_2\sigma(W_1[y^{(1,0)}; y^{(1,K)}] + b_1) + b_2 \tag{3}$$

$$z_2^i = W_2\sigma(W_1[y^{(2,0)}; y^{(2,K)}] + b_1) + b_2 \tag{4}$$

$$y_i^{(1,k)} = \max_{0 \le t < d_1} d_{i,t}^k \tag{5}$$

$$y_j^{(2,k)} = \max_{0 \le t < d_2} d_{t,j}^k \tag{6}$$

$$d_{i,j}^k = \sigma(\sum_{s=0}^{n_k-1}\sum_{t=0}^{n_k-1} w_{s,t}^k M_{i+s,j+t} + b^k) \tag{7}$$

$$M_{ij} = u_i \otimes v_j \tag{8}$$

Where w^k is the k-th kernel. n_k denotes the size of the k-th kernel. ReLU is adopted as the active function σ. z_1^i denotes the aspect of the question to i-th relation in the p-th hop CKC. z_2^i denotes the aspect of i-th relation to the question in the p-th hop CKC. W_k is the learnable parameters of the k-th MLP layer. α_i^p is the attention weight of i-th relation in the p-th hop CKC on literal-level.

- ***Semantic Part***: TransE [22] is the method of KB embedding, which could capture the structural information of KB. The reasoning context l^p contain the context of the question, which could make the reasoning module have memory ability. Instead of using the gate mechanism, we employ an attention mechanism on semantic-level β_i^p, which aims to learn the different contributions between reasoning context l^{p-1} and predicates h_i^p at each hop. The dynamic probability distribution θ_i^p can be measured as followed:

$$\theta_i^p = f([\beta_i^p; \alpha_i^p]) \tag{9}$$

$$\beta_i^p = \frac{exp(w_i^p)}{\sum_{t=0}^{n} exp(w_i^p)} \tag{10}$$

$$w_i^p = V^T tanh(W^T[h_i^p; l^{p-1}]) \tag{11}$$

Where θ_i^p is dynamic probability of i-th relation in the p-th hop CKC. β_i^p denotes semantic weight of (p-1)-th reasoning context to the i-th relation in the p-th hop CKC, and n is the relation length of the CKC. $W^T \in \mathcal{R}^{2d\times d}$ and $V^T \in \mathcal{R}^{d\times 1}$ are intermediate learnable parameters. $f(.)$ is simple MLP without bias.

Final Relation Predication(Static part):We think that the final prediction result of the predicate should consider both dynamic part and static part. In the NLP task, GRU acts as memory storage. We employ the output of unidirectional GRU at p-th hop as the static probability distributions γ_i^p, and use θ_i^p in (12) as dynamic probability distributions. The final relation probability distributions λ_i^p at p-th hop could be defined as follows:

$$\lambda_i^p = h([\theta_i^p; \gamma_i]) \tag{12}$$

$$\gamma_i = GRU(\gamma_{i-1}, l_i) \tag{13}$$

Where function h(.) is MLP function. GRU(.) is GRU function, which can gerenate the hidden vector γ_i at i-th hop, l_i is the current predicate.

C. Relation Detection

After getting a set of (h,p,t) triples, we need this part to score these triples. As shown in Figure 4, this part mainly contains two parts, reasoning module and ranking. The parameters of reasoning module is the same as decode layer. The main problem of reasoning module in this part is to construct CKC. We consider the predicate path as CKC, and use the weighted sum of the predicate as the context of predicate path. Finally, the model gets the final score through a simple dot product.

*1) **Sharing Reasoning Module**:* After the previous steps, we get candidate relation path (s, r1, r2, ..., rn). As for the relation detection task, the responsibility of the reasoning module is to capture the contribution of relation in relation path. We consider the candidate relation path as the first hop CKC, and the formula of the representation h of relation path as follows:

$$h = \sum_{i=0}^{n} \theta_i h_i \tag{14}$$

Where h_i denotes i-th relation in a relation path, and θ_i acts as i-th relation weight in a relation path.

*2) **Ranking**:* Relation detection is used to calculate the score S_{rel} between question and candidate relation paths $r_{1,2,...,n}$ using reasoning module. To enhance our model, similar to Yu et al.(2017), we use entity re-ranking methods involving computations of two scores, the score of the entity S_{entity} and the score of candidate relation paths S_{rel}. The final score can be measured as follows:

$$\begin{aligned} S_{rerank}(q, r_{1,2,...,n}^p) = m * S_{rel}(q, r_{1,2,...,n}) + \\ (1-m) * S_{entity}(q, r_{1,2,...,n}^p) \end{aligned} \tag{15}$$

$$S_{rel}(q, r_{1,2,...,n}) = (q \otimes h) \tag{16}$$

Where, q is the context representation of the question. m is a hyper-parameter. The values of $S_{entity}(q, r_{1,2,...,n}^p)$ is publicly available on the website[4].

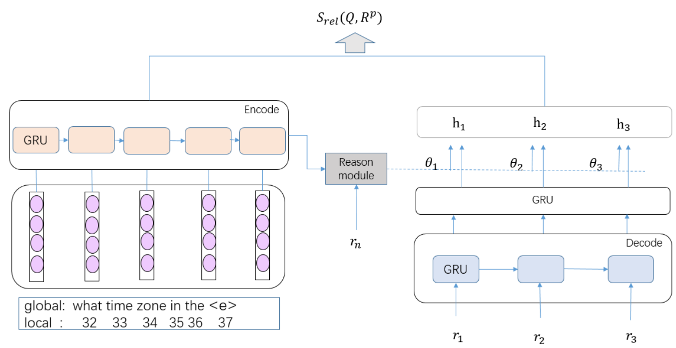

Fig. 4. The overview of relation detection. The left is the context of question sharing the same encode layer in relation generation. The right is the context of the relation path.

D. Loss Function

For our proposed model, we use masked-softmax cross entropy loss function to train relation prediction module. And we use hinge loss to train relation detection, which maximizes the margin between gold sample relation path r^{p+} and negative sample relation path r^{p-}. r^{p-} could be generated as following step:(i) firstly, we use the whole relations as negative sample set which excludes each hop relation. (ii) At each hop, we generate each hop from negative sample set randomly. $\mathcal{D}$ is a set consisting of the gold relation paths. The hinge loss formulas as follow:

$$loss1 = -\sum_{r_i \in \mathcal{R}} q(r_i) log(p(r_i)) \tag{17}$$

$$loss2(q, r^{p-}, r^{p+}) = l \tag{18}$$

$$l = \sum_{(q, r^{p+}) \in \mathcal{D}} max(0, \lambda + S(q, r^{p-}) - S(q, r^{p+})) \tag{19}$$

where $p(r_i)$ is r_i the probability of the prediction generated by the model at i-th hop, $q(r_i)$ is the predicate label at i-th hop.

[4]for SimQ, https://github.com/Gorov/SimpleQuestions-EntityLinking; for WebQ, https://github.com/scottyih/STAGG

V. EXPERIMENT

This section describes an extensive evaluation of our model against state-of-the-art KBQA methods. We use the Freebase as our KB, and use the SimpleQuestion and WebQuestion as our dataset.

A. Dataset

SimpleQuestion(SimQ): SimQ consists of 108,442 questions written by human English-speaking annotators. The dataset is partitioned randomly into 75,910 training question, 10,845 validation question, and 21,687 test questions. The questions in SimQ are all single-hop and single-answer.

WebQuestion(WebQ): WebQ[5] contains 3,778 training example and 2,032 test examples. Each question can have multiple correct answers. In our experiment, we use a development version of the dataset [6], which contains potential noise, such as incorrect relation paths and missing of relation paths.

FreeBase: The answer triples or paths in SimQ and WebQ are all in FreeBase. For SimQ, we use a subset of FreeBase—FB2M, which includes 2,150,604 entities, 6701 relations and 14,180,937 triples. The answer triples in SimQ can all be found in FB2M, whereas not the all answer paths in WebQ are accessible in FB2M. Thus, for the KBQA tasks, the KB associated with SimQ is chosen as FB2M, and that with WebQ is chosen as the entire Freebase[7].

B. Setting

We initialize word embedding using pre-trained GloVe [21] vector with word embedding of size 300, and the global freebase model is also trained separately. In this model, TransE[8] [23] uses FB2M to construct the schame graph for both SimQ and WebQ datasets. The dimension of the relation embedding space is chosen as 100. The hidden layer of GRU has size 300, and hinge loss margin is set to 2. The number of CNN channel is 12 and the kernel size is set to 3. For optimization, parameters are trained using Adam [24] with a learning rate of 0.0005 in mini-batch setting with batch size 64. Drouput is used to regularize CNN and linear-layer in our experiment and is set to 0.3.

For negative sample, the sampleing size of SimQ is set to 5, and the sampleing size of WebQ is set to 50.

C. Baseline

- Yih et al.(2015) use query graph generation method, where KBQA task is formulated as a staged search problem converting the question into query language with constraints and aggregations.
- Jain et al.(2016) use memory network to solve multi-hop reasoning problem through treating the triple (s,r,o) as memory slot at each hop, which creates probability distribution for candidate triple.

[5]nlp.stanford.edu/software/sempre

[6]https://github.com/brmson/dataset-factoid-webquestions

[7]the freebase version can be downloaded in https://github.com/percyliang/sempre

[8]https://github.com/thunlp/KB2E

TABLE I
KBQA RESULTS ON SIMQ AND WEBQ TEST SETS.ACC=ACCURACY

Model	**WebQ(F1 %)**	**SimQ(Acc %)**
(Yih et al.,2015)	52.5	76.4
(Jain et al.2016)	55.6	63.9
(Yu et al.,2017)	-	77.0
(Hao et al.,2017)	42.9	-
(Qu et al.,2018)	-	77.9
(Wang et al.,2018)	-	81.5
(Chen et al.,2019)	55.7	-
(Tong et al.,2019)	44.1	-
Our Method	**57.6**	**82.3**

- Yu et al.(2019) employ hierarchical residual BiLSTM relation detector to improve the performance of relation detection, and make use of S-mart([25]) to re-ranking the result of the entity linking.
- The best model (Wang et al.,2018) refines the training framework of multi-components, and proposes a kind of novel SGD iterative training.
- Qu et al.(2018) prove the effectiveness between question words and relation words based on CNN in SimQ, but their model is limited to only solve single hop question.
- Chen et al.(2019) employ complex bidirectional attention and memory network to capture interactions between the questions and the KB.
- Tong et al.(2019) propose a new KG-QA approach by leveraging the domain context, with the help of a cross-attention model. And they also parse the question tree and utilize meta-path to enrich the representation for answers, which enhances the performance of KBQA.

D. Result

The results of our experiment are presented in Table I. From the results, compared with the best baseline, our model makes improvement of 0.8% accuracy on SimQ, and 1.9% F1 on WebQ. In Wang et al.(2018) researcher, they conclude that the bottleneck of existing model is relation detection. Following their findings, we imporve the accuracy of the predicates before relation detection, such that the model achieve the best result.

However, we have to illustrate the environment of WebQ. There are not predicate paths on original WebQ, so predicate paths must be generated by search algorithm. That may cause different experimental environments. For the sake of fairness, we use the public dataset about predicate paths on WebQ[9], which contains potential noise. And we compared with the model of the same experimental environment on WebQ.

1) ***Ablation Study:*** Both the model Yu et al.(2017) and Yih et al.(2015) add extra features in generating relation paths, e.g., constraints and aggregation. Since the original dataset don't have these extra information, these works need to reannotate the training sample using human cost. However, before starting their work, they all need to generate the main predicate path, just like relation generation module does. In other words, these

[9]https://github.com/brmson/dataset-factoid-webquestions

TABLE II
EXPERIMENTAL RESULTS ON WEBQ AND SIMQ. THIS IS THE ACCURACY OF THE TOP-1 RELATION PATH ON WEBQ AND SIMQ

Model	WebQ(Acc %)	SimQ(Acc %)
(Yih et al.,2015)	63.9	76.4
(Yu et al.,2017)	63.9	77.0
Our Method(All)	**65.02**	**82.3**
Our Method(w/o reasoning module)	60.12	79.4
Our Method(w/o literal part)	63.32	80.5
Our Method(w/o semantic part)	62.59	80.2
Our Method(w/o jointing training)	64.21	81.4

extra features can be embed to our model to improve the performance.

Ablation study of our model is shown in Table II. First, supposed that leaving out the reasoning module, our framework would become the simplest model for handling KBQA. The result of the second line shows that the drop is more obvious on WebQ when leaving out the reasoning module(from 65.02% to 60.12%). That demonstrates the effectiveness of our reasoning module. Second, the reasoning module without jointing training yields a better result on WebQ(from 60.12% to 64.21). Both literal part and semantic part also improve the performence of our model in different degrees (the result of the line 3 and the reault of the line 4). Third, the multi-task learning method improves our system (from 64.21% to 65.02%), because the multi-task learning helps reasoning module to learn more different feature from different tasks.

Comparing the result of the drop rate on SimQ and WebQ when leaving out the reasoning module(4.9% vs 2.9%), our reasoning module has a greater impact on WebQ than SimQ, because the longer the length of the predicate path, the greater the influence of the reasoning module on the performance of the model. That also illustrates the effectiveness of our module.

*2) **Interpretability Analysis**:* We evaluate the effectiveness of the proposed framework. In this section, we aim to answer the following two research questions via using case study:

- How to solve the question that similar structural predicate is indistinguishable.
- How to leverage the reasoning context to make the adaptive module contain reasoning ability?

To answer question one, as showed in Figure 5. The heatmap of the weight generated by a example question ”where was blessed kateri born”. Obviously, our model successfully capture the semantic information of the correct predicate ”people.person.place_of_birth” and the similar predicate ”people.person.nationality”. However, in the final prediction of predicates, the weight of the predicate ”people.person.place_of_birth” is higher than the predicate ”people.person.nationality”, because the predicate ”people.person.place_of_birth” is closer to the question in semantic information. That proves that our model can distinguish subtle difference in the similar structural predicates.

To answer question two, figure 6 shows the heatmap of the weight generated by a example the question ”What is the name of Obama’s father”. The weight of the predicate

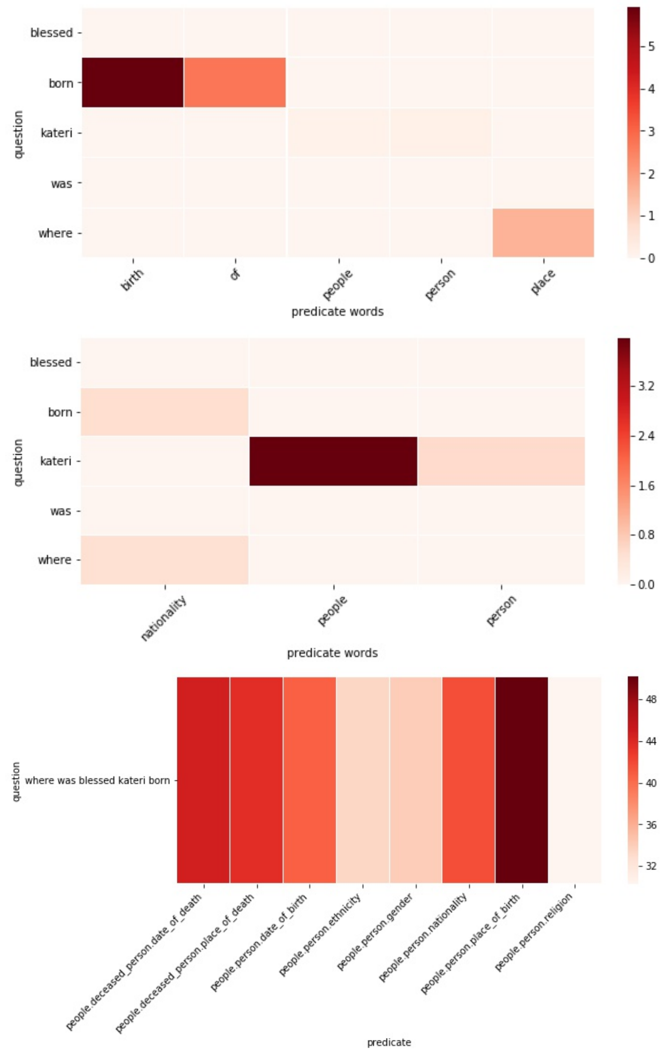

Fig. 5. The weight between the question and the predicate. Red is higher weight. The top picture is the heatmap of the correct predicate ”people.person.place_of_birth”, the bottom picture is the heatmap of other predicates in common CKC. These similarity matrix are extracted from one of CNN cores.

”type.object.name” in first hop CKC is lower than sencod hop CKC, because the reasoning context impacts the weight of the predicate. That illustrates that reasoning module have reasoning ability, such that it can generate the corresponding weights of the predicates according to the reasoning context.

VI. CONCLUSION

In this paper, to our best of knowledge, we are the first to model the interactions between questions and the underlying KB for the KBQA task at each hop dynamically. We proposed an adaptive module with powerful generalization, which could help existing model to solve the scenario of dynamic enviroment KB through scoring the predicates beyond the training samples. Finally, we use multi-task learning to help the reasoning module to reduce variance and improve our model

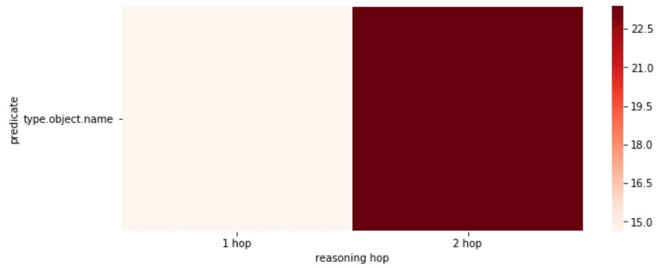

Fig. 6. Weight heatmap generated by the reasoning module. The weight of the predicate "type.object.name" at different hop in question "What is the name of Obama's father". Red is higher weight

fault-tolerant ability. The experimental results demonstate the effectiveness of our model.

In future work, we intend to add the constrains of question into predicate path, which can is translated into the problem of the matching of the query graph. And we would try better pre-trained model to enhance our model. Simultaneously, the challenge of this problem is changed to how to model graph embedding.

VII. Acknowledgment

This research is partially supported by National Natural Science Foundation of China(No.U1711263).

References

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*. Springer, 2007, pp. 722–735.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, 2008, pp. 1247–1250.

[3] E. Prud, A. Seaborne *et al.*, "Sparql query language for rdf," 2006.

[4] S. Jain, "Question answering over knowledge base using factual memory networks," in *Proceedings of the NAACL Student Research Workshop*, 2016, pp. 109–115.

[5] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao, "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)*, 2017, pp. 221–231.

[6] Y. Chen, L. Wu, and M. J. Zaki, "Bidirectional attentive memory networks for question answering over knowledge bases," *arXiv preprint arXiv:1903.02188*, 2019.

[7] M. Yu, W. Yin, K. S. Hasan, C. d. Santos, B. Xiang, and B. Zhou, "Improved neural relation detection for knowledge base question answering," *arXiv preprint arXiv:1704.06194*, 2017.

[8] Z. Xu, H.-T. Zheng, Z. Fu, and W. Wang, "Enhancing question understanding and representation for knowledge base relation detection," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 1362–1367.

[9] Y. Wang, R. Zhang, C. Xu, and Y. Mao, "The apva-turbo approach to question answering in knowledge base," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 1998–2009.

[10] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[11] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on freebase from question-answer pairs," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1533–1544.

[12] W.-t. Yih, M.-W. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, 2015, pp. 1321–1331.

[13] K. Luo, F. Lin, X. Luo, and K. Zhu, "Knowledge base question answering via encoding of complex query graphs," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2185–2194.

[14] X. Yao and B. Van Durme, "Information extraction over structured data: Question answering with freebase," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2014, pp. 956–966.

[15] Z. Dai, L. Li, and W. Xu, "Cfo: Conditional focused neural question answering with large-scale knowledge bases," *arXiv preprint arXiv:1606.01994*, 2016.

[16] D. Golub and X. He, "Character-level question answering with attention," *arXiv preprint arXiv:1604.00727*, 2016.

[17] Y. Qu, J. Liu, L. Kang, Q. Shi, and D. Ye, "Question answering over freebase via attentive rnn with similarity matrix based cnn," *arXiv preprint arXiv:1804.03317*, 2018.

[18] A. Bordes, J. Weston, and N. Usunier, "Open question answering with weakly supervised embedding models," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2014, pp. 165–180.

[19] A. Bordes, S. Chopra, and J. Weston, "Question answering with subgraph embeddings," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 615–620.

[20] R. Dey and F. M. Salemt, "Gate-variants of gated recurrent unit (gru) neural networks," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2017, pp. 1597–1600.

[21] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[22] 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.

[23] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

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

[25] Y. Yang and M.-W. Chang, "S-MART: Novel tree-based structured learning algorithms applied to tweet entity linking," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, Jul. 2015, pp. 504–513. [Online]. Available: https://www.aclweb.org/anthology/P15-1049