Improving Multi-hop Knowledge Base Question Answering by Learning Intermediate Supervision Signals

RUC AI Box

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao and Ji-Rong Wen

Renmin University of China, Singapore Management University

{hegaole,jrwen}@ruc.edu.cn, batmanfly@gmail.com, {yslan,jingjiang}@smu.edu.sg

SMU SINGAPORE MANAGEMENT UNIVERSITY

Introduction

Knowledge Base Question Answering.

Knowledge Base Question Answering (KBQA) is a challenging task that aims at finding answers to questions expressed in natural language from a given knowledge base (KB). A major challenge is the lack of supervision signals at intermediate steps. Therefore, multi-hop KBQA algorithms can only receive the feedback from the final answer, which makes the learning unstable or ineffective. To address this challenge, we propose a novel teacher-student approach for the multi-hop KBQA task.

What is our method ?

In our approach, the student network aims to find the correct answer to the query, while the teacher network tries to learn intermediate supervision signals for improving the reasoning capacity of the student network. The major novelty lies in the design of the teacher network, where we utilize both forward and backward reasoning to enhance the learning of intermediate entity distributions. By considering bidirectional reasoning, the teacher network can produce more reliable intermediate supervision signals, which can alleviate the issue of spurious reasoning.

Neural State Machine for Multi-hop KBQA

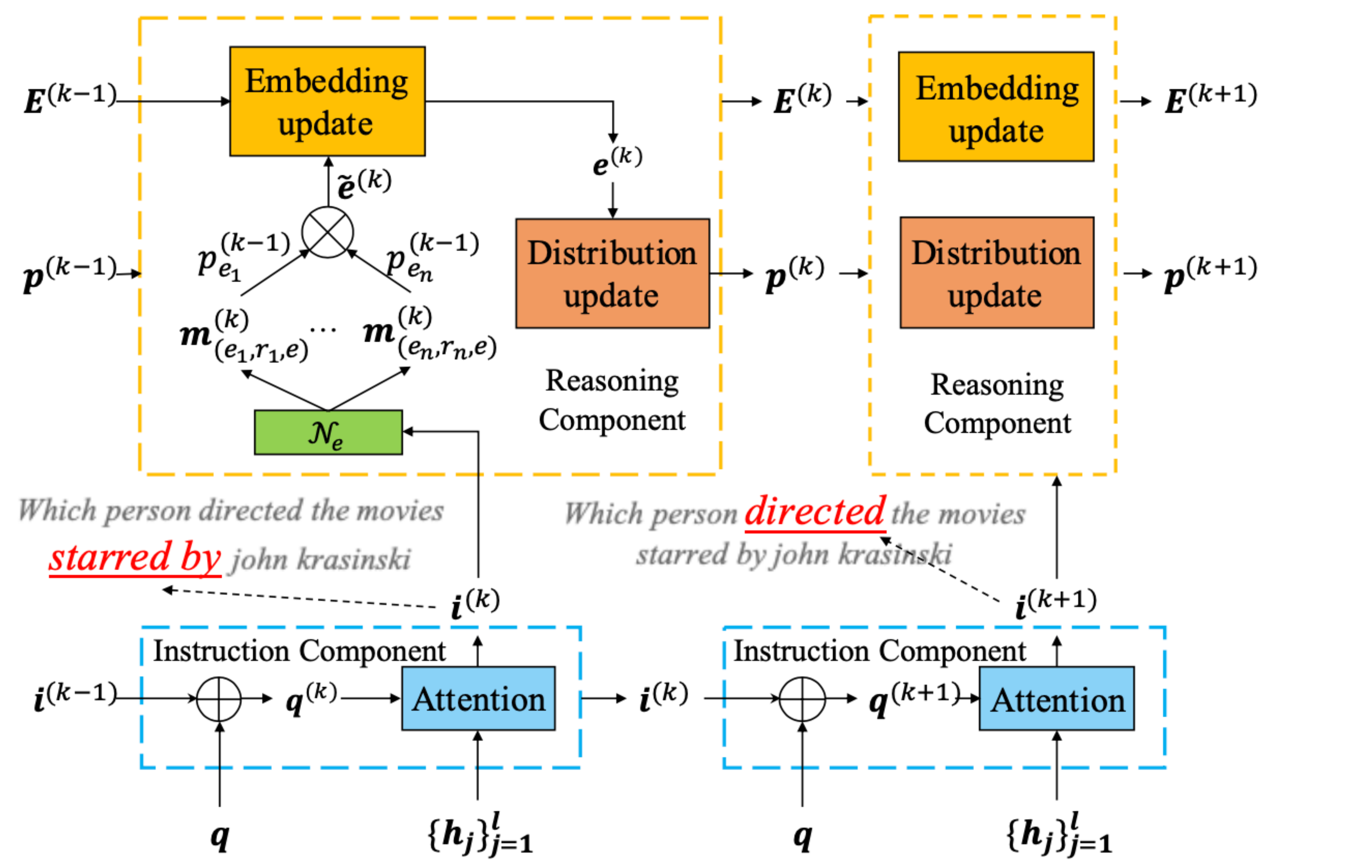

Figure 1: Illustration of the two reasoning steps for neural state machine. In different reasoning steps, the instruction vector focuses on different parts of the question.

More Details:

1. Instruction Component employs the LSTM encoder to obtain a set of hidden states $\{\boldsymbol{h}^j\}_{j=1}^{l}$, where $\boldsymbol{h}^j \in R^d$ and l is the length of the question. And the last hidden state is considered to be the question representation, $\boldsymbol{q} = \boldsymbol{h}^l$. Let $\boldsymbol{i}^{(k)} \in R^d$ denote the instruction vector at the k-th reasoning step. We adopt the following method to learn the instruction vector $\boldsymbol{i}^{(k)}$:

$$\boldsymbol{i}^{(k)} = \sum_{j=1}^{l} \alpha_j^{(k)} \cdot \boldsymbol{h}^j \qquad a_k = \boldsymbol{W}_\alpha(q^{(k)} \odot h^j) + b_\alpha$$

$$\alpha_k = \text{softmax}(a_k) \qquad \boldsymbol{q}^{(k)} = \boldsymbol{W}^{(k)}[i^{(k-1)}; \boldsymbol{q}] + \boldsymbol{b}^{(k)}$$

2. Reasoning Component infers the entity distribution and learns the entity representations. Once we obtain the instruction vector $\boldsymbol{i}^{(k)}$, we can use it as a guiding signal for the reasoning component. The input of the reasoning component consists of the instruction vector of the current step, and the entity distribution and entity embeddings obtained from the previous reasoning step. The output of the reasoning component includes the entity distribution p(k) and the entity embeddings {e(k)}

$$\boldsymbol{e}^{(0)} = \sigma\Big(\sum_{(e',r,e)\in N_e} rW_T\Big) \qquad \boldsymbol{m}^{(k)}_{<e_i,r,e>} = \sigma(\boldsymbol{i}^{(k)}\boldsymbol{W}_R\boldsymbol{r})$$

$$\boldsymbol{e}^{(k)} = FFN([\boldsymbol{e}^{(k-1)}; \tilde{\boldsymbol{e}}^{(k)}]) \qquad \tilde{\boldsymbol{e}}^{(k)} = \sum_{(e',r,e)\in N_e} \boldsymbol{p}_{e'}^{(k-1)} \cdot \boldsymbol{m}^{(k)}_{<e',r,e>}$$

$$\boldsymbol{p}^{(k)} = softmax({\boldsymbol{E}^{(k)}}^T \boldsymbol{w})$$

Teacher-student Network

1. Bidirectional Reasoning for Multi-hop KBQA: Most existing multi-hop KBQA methods start from the topic entities and then look for the possible answer entities, called ***forward reasoning***. On the other hand, the opposite search from answer entities to topic entities (which we refer to as ***backward reasoning***) has been neglected by previous studies. Our core idea is to consider the exploration in both directions and let the two reasoning processes synchronize with each other at intermediate steps. Specifically, to optimize such correspondence, we adopt constraint based loss for teacher network:

$$\mathcal{L}_C = \sum_{k=1}^{n-1} D_{JS}(\boldsymbol{p}_f^{(k)}, \boldsymbol{p}_b^{(n-k)})$$

Where $\boldsymbol{p}_f^{(k)}$ and $\boldsymbol{p}_b^{(n-k)}$ denote the entity distributions from the forward reasoning at the k-th step and from the backward reasoning at the $(n-k)$-th step, respectively. Intermediate entity distribution obtained by teacher network:

$$\boldsymbol{p}_t^{(k)} = \frac{1}{2}(\boldsymbol{p}_f^{(k)} + \boldsymbol{p}_b^{(n-k)})$$

2. Teacher Network Based on the idea above, we design two kinds of neural architectures for the teacher network:

(1) Parallel Reasoning set up two separate NSMs for both forward and backward reasoning, respectively. These two NSM networks are relatively isolated, and do not share any parameters.

(2) Hybrid Reasoning share the same instruction component and arrange the two reasoning processes in a cycled pipeline. Besides the correspondence constraints, the two processes receive the same instruction vectors. Furthermore, the derived information at the final step of the forward reasoning is fed into the backward reasoning as initial values. Formally, the following equations hold in this case:

$$\boldsymbol{p}_b^{(0)} = \boldsymbol{p}_f^{(b)}, \boldsymbol{E}_b^{(0)} = \boldsymbol{E}_f^{(n)}, \boldsymbol{i}_b^{(k)} = \boldsymbol{i}_f^{(n-k)}$$

3. Student Network is implemented based on Neural State Machine (NSM). The entity distributions at intermediate reasoning steps of teacher network will be subsequently used by the student network as the supervision signals. The training loss is defined as:

$$\mathcal{L}_s = \mathcal{L}_1 + \lambda\mathcal{L}_2 \qquad \mathcal{L}_1 = D_{KL}\left(\boldsymbol{p}_s^{(n)}, \boldsymbol{p}_f^{*}\right) \qquad \mathcal{L}_2 = \sum_{k=1}^{n-1} D_{KL}(\boldsymbol{p}_s^{(k)}, \boldsymbol{p}_t^{(k)})$$

For more details, please refer to our paper.

Experiment

Our code and data are available at https://github.com/RUCAIBox/WSDM2021_NSM.

To evaluate our model on three benchmark KBQA datasets. For each test question in a dataset, a list of answers are returned by a model according to their predictive probabilities. We adopt two evaluation metrics widely used in previous works, namely Hits@1 and F1.

Table 1: Statistics of all datasets. "#entity" denotes average number of entities in subgraph, and "coverage" denotes the ratio of at least one answer in subgraph.

Datasets	Train	Dev	Test	#entity	coverage
MetaQA-1hop	96,106	9,992	9,947	487.6	100%
MetaQA-2hop	118,980	14,872	14,872	469.8	100%
MetaQA-3hop	114,196	14,274	14,274	497.9	99.0%
webqsp	2,848	250	1,639	1,429.8	94.9%
CWQ	27,639	3,519	3,531	1,305.8	79.3%

Table 2: Performance comparison of different methods for KBQA (Hits@1 in percent).

Models	Webqsp	MetaQA-1	MetaQA-2	MetaQA-3	CWQ
KV-Mem	46.7	96.2	82.7	48.9	21.1
GraftNet	66.4	97.0	94.8	77.7	32.8
PullNet	68.1	97.0	**99.9**	91.4	45.9
SRN	-	97.0	95.1	75.2	-
EmbedKGQA	66.6	**97.5**	98.8	94.8	-
NSM	68.7	97.1	**99.9**	**98.9**	47.6
NSM_{+p}	73.9	97.3	**99.9**	**98.9**	48.3
NSM_{+h}	**74.3**	97.2	**99.9**	**98.9**	**48.8**

Table 3: Ablation study of the teacher network (in percent).

Models	Webqsp		CWQ	
	Hits	F1	Hits	F1
NSM	68.7	62.8	47.6	42.4
NSM_{+f}	70.7	64.7	47.2	41.5
NSM_{+b}	71.1	65.4	47.1	42.7
$\text{NSM}_{+p,-c}$	72.5	66.5	47.7	42.7
$\text{NSM}_{+h,-c}$	73.0	66.9	47.5	42.1
NSM_{+p}	73.9	66.2	48.3	**44.0**
NSM_{+h}	**74.3**	**67.4**	**48.8**	**44.0**