Hierarchical Memory Networks for Answer Selection on Unknown Words

Jiaming Xu[a,*], Jing Shi[a,*], Yiqun Yao[a], Suncong Zheng[a], Bo Xu[a], Bo Xu[a,b]
[a]Institute of Automation, Chinese Academy of Sciences (CAS). Beijing, China
[b]Center for Excellence in Brain Science and Intelligence Technology, CAS. China
{jiaming.xu, shijing2014, yaoyiqun2014}@ia.ac.cn
{suncong.zheng, boxu, xubo}@ia.ac.cn

Abstract

Recently, end-to-end memory networks have shown promising results on Question Answering task, which encode the past facts into an explicit memory and perform reasoning ability by making multiple computational steps on the memory. However, memory networks conduct the reasoning on sentence-level memory to output coarse semantic vectors and do not further take any attention mechanism to focus on words, which may lead to the model lose some detail information, especially when the answers are rare or unknown words. In this paper, we propose a novel Hierarchical Memory Networks, dubbed HMN. First, we encode the past facts into sentence-level memory and word-level memory respectively. Then, k-max pooling is exploited following reasoning module on the sentence-level memory to sample the k most relevant sentences to a question and feed these sentences into attention mechanism on the word-level memory to focus the words in the selected sentences. Finally, the prediction is jointly learned over the outputs of the sentence-level reasoning module and the word-level attention mechanism. The experimental results demonstrate that our approach successfully conducts answer selection on unknown words and achieves a better performance than memory networks.

1 Introduction

With the recent resurgence of interest in Deep Neural Networks (DNN), many researchers have concentrated on using deep learning to solve natural language processing (NLP) tasks (Collobert et al., 2011; Sutskever et al., 2014; Zeng et al., 2014; Feng et al., 2015). The main merits of these representation learning based methods are that they do not rely on any linguistic tools and can be applied to different languages or domains. However, the memory of these methods, such as Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014) compressing all the external sentences into a fixed-length vector, is typically too small to accurately remember facts from the past, and may lose important details for response generation (Shang et al., 2015). Due to the drawback, these traditional DNN models encounter great limitation on Question Answering (QA), as a complex NLP task, which requires deep understanding of semantic abstraction and reasoning over facts that are relevant to a question (Hermann et al., 2015; Yu et al., 2015).

Recently, lots of deep learning methods with explicit memory and attention mechanism are explored for Question Answering (QA) task, such as Memory Networks (MemNN) (Sukhbaatar et al., 2015), Neural Machine Translation (NMT) and Neural Turing Machine (NTM) (Yu et al., 2015). These methods exploit a external memory to store the past sentences with a continuous representation and utilize attention mechanism to automatically soft-search for parts of the memory for prediction. Compared with NMT and NTM, MemNN, making multiple computational steps (termed as "hops") on the memory before making an output, is better qualified for textual reasoning tasks. However, for QA task, MemNN only conducts the reasoning on sentence-level memory and does not further take any attention mechanism to focus on words in the retrieved facts. More recently, Yu et al. (2015) constructed a *Search-Response*

*The first two authors contributed equally.

pipeline where *Search* component uses MemNN to search the supporting sentences and *Response* component uses NMT or NTM to generate answer on the selected sentences. However, that work needs the supervision of the supporting facts to guide the training of *Search* component and the combination of these two components through a separate training way may hurt the performance. Along the direction of that work, we believe that a joint learning model can achieve a better performance by designing a hierarchical architecture, with sentence-level and word-level components, which has shown promising results on document modeling (Lin et al., 2015) and document classification (Yang et al., 2016).

Besides, rare and unknown word problem as an important issue should be considered in NLP tasks, especially for QA task, where the words that we are mainly interested in are usually named entities which are mostly unknown or rare words (Marrero et al., 2013; Gulcehre et al., 2016). In order to control the computational complexity, many methods limit the trained vocabulary size, which further leads to lots of low-frequency words outside the trained vocabulary (Li et al., 2016). Traditional methods directly mask the rare or unknown words with meaningless *unk* which may lose the important information for answer selection task. For example, given a set of sentences as follows:

1. Miss, what is your name?
2. Uh, my name is *Wainwright*.
3. Please tell me your passport number.
4. Ok, it is *899917359*.

Assume that the words *Wainwright* and *899917359*[1] are rare words or outside the trained vocabulary. If these words are discarded or replaced with *unk* symbol, any models may not be able to select the correct answers for response during testing.

Based on the above observations, this paper proposes a Hierarchical Memory Networks[2] (dubbed to HMN) for answer selection. Our method first maps the sentences into a sentence-level memory and reasoning module takes multiple hops on the sentence-level memory to soft-search the related sentences. Meanwhile, all words in the sentences are encoded into a word-level memory with recurrent neural networks. Then, we exploit k-max pooling to sample the most relevant sentences and feed these selected sentences into attention mechanism on the word-level memory to focus the words. Finally, the prediction is jointly learned over the outputs of the sentence-level reasoning module and the word-level attention mechanism. Our main contributions are three-fold:

(1). We proposed a novel hierarchical memory networks for answer selection, where the reasoning module is performed on sentence-level memory to retrieve the relevant sentences and the attention mechanism is applied on word-level memory to focus the words. This hierarchical architecture allows the model to have explicit reasoning ability on sentences and also focus on more fine-grained words.

(2). k-max pooling is exploited to sample the most relevant facts based on the results of the sentence-level reasoning and then feed these facts into word-level attention mechanism, which can filter the noise information and also reduce the computational complexity on word-level attention.

(3). We release four synthetic domain dialogue datasets[3], two from air-ticket booking domain and two from hotel reservation domain, where the answers are mostly rare or unknown words, and lots of answers should be reasoned based on some supporting sentences. The experimental results show that our approach can successfully conduct answer selection on unknown words.

2   Background: Memory Networks

Here, we give a brief description of memory networks which have shown promising results on QA tasks (Weston et al., 2015; Bordes et al., 2015; Sukhbaatar et al., 2015). Memory network first introduced by Weston et al. (2015) is a new class of learning models which can easily read and write to part of a long-term memory component, and combine this seamlessly with inference for prediction. Formally, besides the explicit memory which is an array of cells to memorize the pre-trained vector representations

---

[1]Note that the personal information used in our examples and datasets throughout this paper is all synthetic and not real.

[2]It is worth noticing that the term "Hierarchical Memory Networks" has been mentioned in (Chandar et al., 2016) where the intention was to organize the memory into multi-level groups based on hashing, tree or clustering structures to make the reader efficiently access the memory, whereas in our paper the term has a different meaning.

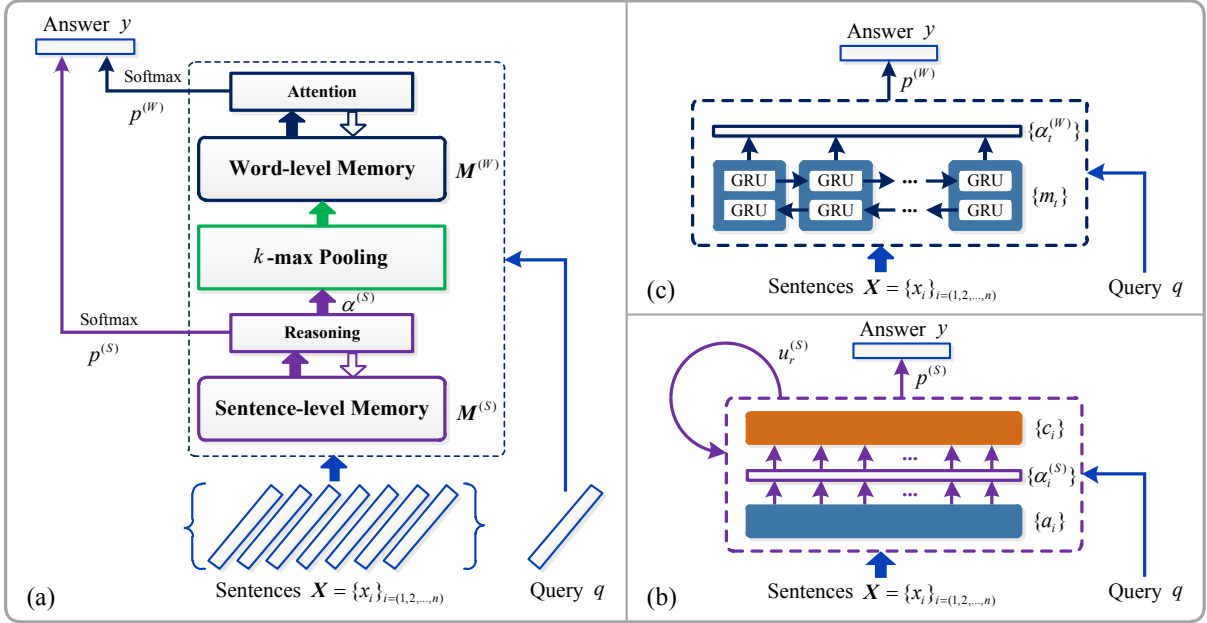[3]Our code and dataset are available: `https://github.com/jacoxu/HMN4QA`

Figure 1: An illustration of our Hierarchical Memory Networks (HMN). (a): The overall architecture of the proposed HMN. (b): Reasoning module of our approach on sentence-level memory. (c): Attention module of our approach on word-level memory.

of the external data, a general memory network consists of four major components: (1). *Input feature map* which converts the incoming input to the internal feature representation. (2). *Generalization* which updates old memories given the new input. (3). *Output feature map* which produces a new output based on the new input and the current state. (4). *Response* which converts the output into the response format desired. Along the above framework, Sukhbaatar et al. (2015) put forward end-to-end memory networks which do not require the supervision of the supporting facts and are more generally applicable in realistic setting. Thus, we choose end-to-end memory networks, denoted as MemNN throughout our paper, as the foundation of our proposed approach.

3 Hierarchical Memory Networks for Answer Selection

3.1 Approach Overview

As described in Figure 1(a), we give an illustration of our HMN for answer selection. Given a set of n sentences denoted as: $\mathbf{X} = \{x_i\}_{i=(1,2,...,n)}$ and a query q, where i is the timestep of sentence x_i in the set. We first map these sentences $\mathbf{X}$ into the sentence-level memory $\mathbf{M}^{(S)}$ and the word-level memory $\mathbf{M}^{(W)}$ with low-dimensional distributed representations respectively. Then, reasoning on the sentence-level memory is utilized to soft-search the related sentences. We further exploit k-max pooling to sample the most relevant sentences based on the soft-searching results and take attention mechanism to focus on word-level memory of the selected sentences. The target answer y is used to guide the learning of the reasoning on sentence-level memory and the attention on word-level memory learning simultaneously.

3.2 Sentence-level Memory and Reasoning

In this section, we apply reasoning module to make multiple interaction on sentence-level memory based on the adjacent weight tying scheme of MemNN (Sukhbaatar et al., 2015), as shown in Figure 1(b). Given two word embedding matrices $\mathbf{A} \in \mathbb{R}^{|V| \times d}$ and $\mathbf{C} \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the vocabulary size and d is the dimension of the word embedding, we first encode the word x_{ij} at timestep j in the sentence x_i into dual channels of word representation as $\mathbf{A}x_{ij} \in \mathbb{R}^d$ and $\mathbf{C}x_{ij} \in \mathbb{R}^d$.

In order to combine the order of the words into their representations, a positional encoding matrix $\mathbf{l}$ is applied to update the dual-channel word embeddings as $l_{gj} \cdot (\mathbf{A}x_{ij})$ and $l_{gj} \cdot (\mathbf{C}x_{ij})$, where

$$l_{gj} = (1 - j/J_i) - (g/d)(1 - 2j/J_i), \quad 1 \le j \le J_i, \ 1 \le g \le d, \tag{1}$$

and J_i is the length of the sentence x_i and g is the embedding index. This positional encoding scheme is also successfully applied in (Xiong et al., 2016).

Two temporal encoding matrices $\mathbf{T}_A \in \mathbb{R}^{n \times d}$ and $\mathbf{T}_C \in \mathbb{R}^{n \times d}$ are further utilized to encode the order of the sentences. Then, the sentence-level memory $\mathbf{M}^{(S)} = \{\{a_i\}, \{c_i\}\}_{i=(1,2,...,n)}$ is reformed as:

$$a_i = \sum_j l_j \cdot (\mathbf{A}x_{ij}) + \mathbf{T}_A(i), \quad c_i = \sum_j l_j \cdot (\mathbf{C}x_{ij}) + \mathbf{T}_C(i), \tag{2}$$

where l_j is the j-th column vector of the position encoding matrix $\mathbf{l}$ according to the sentence x_i and the operation "$\cdot$" means the element-wise multiplication.

For the query q, the j-th word q_j is also embedded as $\mathbf{A}q_j \in \mathbb{R}^d$, where the $\mathbf{A}$ is the embedding matrix used in Eqn. (2). By encoding the word position j into the query representation, we get the probe representation of the query q as follows:

$$u_1^{(S)} = \sum_j l_j \cdot (\mathbf{A}q_j), \tag{3}$$

where l_j is the j-th column vector of the position encoding matrix $\mathbf{l}$ according to the query q. Then the attention weights of the sentences according to the query can be calculated through the inner product of the two vectors as $\alpha_i^{(S)} = softmax(a_i^T u_1^{(S)})$, and the output of the sentence-level memory based on the activation of the query can be obtained as: $o_1 = \sum_i \alpha_i^{(S)} c_i$.

In order to perform reasoning on sentence-level memory to find the most relevant sentences, we make R hops to soft-search the sentences and output the final vector o_R. To be specific, during the $r + 1$ hop of the reasoning operation, the process can be formalized as: $u_{r+1}^{(S)} = o_r + u_r^{(S)}$, $\alpha_i^{(S)} = softmax(a_i^T u_{r+1}^{(S)})$, $o_{r+1} = \sum_i \alpha_i^{(S)} c_i$, and the dual-channel memories are updated as follows:

$$a_i = \sum_j l_j \cdot (\mathbf{A}^{r+1}x_{ij}) + \mathbf{T}_A^{r+1}(i), \quad c_i = \sum_j l_j \cdot (\mathbf{C}^{r+1}x_{ij}) + \mathbf{T}_C^{r+1}(i), \tag{4}$$

where $1 \le r \le (R - 1)$. Specifically, during the $r + 1$ hop, the word embedding matrices $\mathbf{A}^{r+1}$ and $\mathbf{C}^{r+1}$ are mutually independent, so as the temporal encoding matrices $\mathbf{T}_A^{r+1}$ and $\mathbf{T}_C^{r+1}$. But during the adjacent two hops, $\mathbf{A}^{r+1} = \mathbf{C}^r$ and $\mathbf{T}_A^{r+1} = \mathbf{T}_C^r$. Finally, we can get the predicted word probability distribution by applying softmax on the output vector of the reasoning on the sentence-level memory as:

$$p^{(S)}(w) = softmax((\mathbf{C}^R)^T(o_R + u_R^{(S)})), \tag{5}$$

where $w = \{w_t\}_{t=(1,2,...,|V|)}$ is the word set with a vocabulary size of $|V|$, the weight matrix is the same as the embedding matrix $\mathbf{C}^R \in \mathbb{R}^{|V| \times d}$ on the last hop, and T is the operation of matrix transposition.

3.3 k-max Pooling

Here, we exploit a pooling operation over the top attention weights $\alpha^{(S)}$ of the reasoning module on the sentence-level memory to sample the most relevant sentences. Given a value k and the top attention weights $\alpha^{(S)}$ of length $n \ge k$, we use the k-max pooling to select a subset of sentence sequences $\hat{\mathbf{X}} = \{\hat{x}_i\}_{i=(1,2,...,k)}$, corresponds with their top-k maximum values of $\alpha^{(S)}$ on the sentences.

The k-max pooling operation makes it possible to pool the k most relevant sentences to the query and filter the noise information, which maybe more beneficial to select the correct answers. Moreover, this sampling module feeds a subset of sentences $\hat{\mathbf{X}}$ to the following attention mechanism on the word-level memory, which can reduce the computation complexity of the attention to focus on the relevant words.

3.4 Attention on Word-level Memory

For word-level memory, we first apply a Bi-directional GRU (BiGRU) to compute the hidden states of all the ordered words $\bar{w} = \{\bar{w}_t\}_{t=(1,2,...,|t|)}$ in the sentence set $\mathbf{X}$, where $|t|$ is the time steps of the words in the sentences. In particular, for the t-th word $\bar{w}_t$, the forward GRU and the backward GRU encode

Domain (Lang)	Train/Dev/Test	Vocab	Unseen Answers (Dev/Test)	Max Len (P/S)
Air-Ticket (CH)	5,400/600/6,000	8,540	409 (68.2%)/4,020 (67.0%)	16/17
Hotel (CH)	5,400/600/6,000	7,586	367 (61.2%)/3,690 (61.5%)	16/16
Air-Ticket (EN)	5,400/600/6,000	7,537	342 (57.0%)/3,489 (58.2%)	16/18
Hotel (EN)	5,400/600/6,000	7,134	357 (59.5%)/3,452 (57.5%)	16/16
Total	21,600/2,400/24,000	29,092	1,406 (58.6%)/13,872 (57.8%)	16/18

Table 1: Statistics of the datasets, including the domain and the language of the dataset (CH: Chinese and EN: English), the number of train/dev/test set entries, the vocabulary size of datasets, the number and proportion of unseen answers on dev/test set, and the max paragraph length and sentence length of the datasets. The *Total* dataset consists all the samples of the above four datasets.

it as hidden states $\vec{h}_t = \vec{GRU}(\mathbf{C}^R \bar{w}_t)$ and $\overleftarrow{h}_t = \overleftarrow{GRU}(\mathbf{C}^R \bar{w}_t)$ respectively, where $\mathbf{C}^R$ is the word embedding matrix of the last hop on the sentence-level memory, and we set the dimension of $\vec{h}_t$ and $\overleftarrow{h}_t$ equals to the dimension of the word embedding. By summing the forward hidden states and the backward hidden states, we obtain the word-level memory as $\mathbf{M}^{(W)} = \{m_t\}_{t=(1,2,...,|t|)}$, where $m_t = \vec{h}_t + \overleftarrow{h}_t$. In this way, the memory m_t contains the context information of the t-th word $\bar{w}_t$ in the sentence set $\mathbf{X}$.

Then, we perform attention on the subset of the ordered words $\hat{w} = \{\hat{w}_t\}_{t=(1,2,...,|\hat{t}|)}$ in the selected sentences $\hat{\mathbf{X}}$ by using the probe vector $u_R^{(S)}$ of the last hop on the sentence-level memory and a subset of word-level memory $\{\hat{m}_t\}_{t=(1,2,...,|\hat{t}|)}$ selected form $\mathbf{M}^{(W)}$ according to the word subset $\hat{w}$. The normalized attention weights $\alpha^{(W)} = \{\alpha_t^{(W)}\}_{t=(1,2,...,|\hat{t}|)}$ on the word-level memory are calculated as:

$$\alpha_t^{(W)} = softmax(v^T tanh(\mathbf{W}u_R^{(S)} + \mathbf{U}\hat{m}_t)), \tag{6}$$

where $v \in \mathbb{R}^{d \times 1}$, $\mathbf{W} \in \mathbb{R}^{d \times d}$ and $\mathbf{U} \in \mathbb{R}^{d \times d}$ are all learning parameters updated during the training. Inspired by Pointer Networks (Vinyals et al., 2015), we adopt the normalized attention weights $\alpha^{(W)}$ on the word collection $\hat{w}$ as the probability distribution of the output words:

$$p^{(W)}(w) = trans(p^{(W)}(\hat{w})) = trans(\alpha^{(W)}), \tag{7}$$

where $trans(\cdot)$ means the operation to map the words probability distribution $p^{(W)}(\hat{w}) \in \mathbb{R}^{|\hat{t}|}$ into the probability distribution $p^{(W)}(w) \in \mathbb{R}^{|V|}$. To be specific, the map operation makes the probability distribution $p^{(W)}(\hat{w})$ of the word subset ($\hat{w} = \{\hat{w}_t\}_{t=(1,2,...,|\hat{t}|)}$) to be added into their corresponding positions in the vocabulary ($w = \{w_t\}_{t=(1,2,...,|V|)}$), and the probabilities of the words not in selected word subset $\hat{w}$ will be set to zero[4]. Finally, we get the new probability distribution $p^{(W)}(w) \in \mathbb{R}^{|V|}$.

3.5 Joint Learning

In this paper, we combine the probability distributions of the output words both on the sentence-level memory and the word-level memory to predict the joint probability distribution $p(w)$ as follows:

$$p(w) = p^{(S)}(w) + p^{(W)}(w). \tag{8}$$

Finally, we use the target answer y to guide the learning of the reasoning module on sentence-level memory and the attention module on word-level memory simultaneously. We choose the cross entropy as the cost function and apply Stochastic Gradient Descent (SGD) (Bottou, 1991) as the optimization method to train our joint model. The learned parameters include word embedding matrices $\mathbf{A}^1$ and $\{\mathbf{C}^r\}_{r=(1,2,...,R)}$, temporal encoding matrices $\mathbf{T}_A^1$ and $\{\mathbf{T}_C^r\}_{r=(1,2,...,R)}$ in Eqn. (2) and (4), the parameters $\{\theta_{BiGRU}\}$ of the BiGRU model and the attention parameters v, $\mathbf{W}$ and $\mathbf{U}$ in Eqn. (6).

---

[4]For example, if the word "airport", at two different timesteps in the ordered word subset $\hat{w}$, has two probabilities "0.1" and "0.3", the map operation would add up these probabilities and set "0.4" as the probability of the word "it" in the vocabulary.

	Air-Ticket (CH)	Hotel (CH)	Air-Ticket (EN)	Hotel (EN)	Total
MemNN-H1	4,727.8±79.2	3,680.8±48.3	4,051.8±53.8	3,067.4±71.3	14,492.8±282.8
MemNN-NT	3,424.0±106.6	2,816.2±35.9	2,984.2±93.0	2,304.0±70.3	10,250.4±152.1
MemNN	67.5±9.1	84.0±9.8	55.5±9.4	56.2±10.7	225.6±41.1
HMN-Sent	99.8±29.9	175.0±19.3	112.8±51.2	129.2±15.3	125.6±27.4
HMN-Word	22.2±7.5	31.8±7.8	24.8±9.6	12.8±4.9	27.4±4.8
HMN-Joint	**4.2±2.5**	**9.8±1.9**	**4.4±2.7**	**3.0±1.2**	**4.2±1.8**

Table 2: Comparison of predicted test error numbers of our HMN and MemNN with different components on four domain datasets (Test: 6,000 samples) and the *Total* dataset (Test: 24,000 samples). MemNN-H1: Memory network with 1 hop and temporal encoding, MemNN-NT: Memory network with 3 hops but without temporal encoding. MemNN: Memory network with 3 hops and temporal encoding. Note that HMN-Sent and HMN-Word, as the parts of HMN-Joint, are joint learning but give their predicted answers separately.

4 Experiments

4.1 Datasets and Setup

We conduct answer selection tasks on four synthetic domain dialogue datasets, two from air-ticket booking domain and two from hotel reservation domain. One complete dialogue history of each dataset has eight round responses. Besides greeting and ending sentences, one dialogue history consists six round responses to query and answer client's personal information, such as name, phone and passport number. The datasets contain hundreds of response patterns and thousands of entity information. More detailed descriptions can be found in our released datasets. The statistics of the datasets are summarized in Table 1. We use 45% of the data for training, 5% for validation and the remaining 50% for test. From the statistics, we can see that the proportions of the unseen answers on dev/test sets all overtake 57%.

In our experiments, the most of hyper parameters are set uniformly for the datasets as described in Table 3. The training gradients with an $l2$ norm larger than 40 are clipped to 40 and the learning rate is annealed every 15 epochs by $\lambda/2$ until 60 epochs are reached. The learned parameters are all initialized randomly from a Gaussian distribution with zero mean and 0.1 standard deviation. In order to make the comparison more intuitive, we use the number of predicted error samples on each dataset to evaluate the performance for answer selection and calculate the average result by repeating each experiment 5 times.

Hyperparameter	Hidden dim.	Hops	Max pooling	Learning rate	Batch size (*Total*)
Value	$d = 100$	$R = 3$	$k = 4$	$\lambda = 0.01$	30 (32)

Table 3: Hyperparameters used in our experiments. The dimension of word embeddings and the dimension of GRU hidden states are set equally to 100, batch sizes are 32 for *Total* and 30 for the others.

4.2 Comparison with Memory Networks

In order to evaluate the effect of multiple hops for reasoning module and temporal encoding on sentence-level memory, we design three MemNN (Sukhbaatar et al., 2015) based variants: MemNN-H1 (1 hop and temporal encoding), MemNN-NT (3 hops but not temporal encoding) and MemNN (3 hops and temporal encoding) on the datasets. We further evaluate the prediction performance of our HMN on different level memory components via HMN-Sent (prediction of reasoning module on sentence-level memory as Eqn. (5)), HMN-Word (prediction of attention module on word-level memory as Eqn. (7)) and HMN-Joint (joint prediction as Eqn. (8)). The comparison of these methods are reported in Table 2. From the results, we can see that MemNN-H1 without temporal encoding and MemNN-NT without multi-hop reasoning make the worst performances, which clearly demonstrate that multiple hops for reasoning module and temporal encoding on sentence-level memory play a very important role on our tasks. Despite that MemNN represents surprising results on this task, our HMN-Word and HMN-joint

Dialogue history		Reasoning			Sampling	Attention
Time	Text	Hop 1	Hop 2	Hop 3	4-max pooling	Attention on word-level memory
1.	Hello, this is air ticket booking agent center. What can I do for you?	4.5e-2	3.7e-4	1.5e-31		0.0 0.0 0.0 0.0 0.0 0.0 ↓ ↓ ↓ ↓ ↓ ↓
...		--	--	--	→	899917359 , my passport number .
8.	899917359, my passport number.	5.7e-2	2.7e-3	1.7e-29		4.3e-7 0.0 ↓ ↓
9.	The phone number, Miss?	4.1e-2	1.3e-3	7.2e-31	→	0016173976838 .
10.	0016173976838.	7.2e-2	2.3e-3	3.7e-28		9.2e-6 1.1e-5 1.0e-7 0.0 1.2e-6 0.0 0.0 ↓ ↓ ↓ ↓ ↓ ↓ ↓
...		--	--	--	→	Miss , the travel time is ?
13.	Miss, the travel time is?	1.7e-1	3.0e-1	6.9e-27		0.0 0.0 0.0 0.0 8.5e-7 1.0 ↓ ↓ ↓ ↓ ↓ ↓
14.	I'd like to go on 10/13/2018	1.6e-1	6.9e-1	1.0	→	I'd like to go on 10/13/2018
...		--	--	--		

Question: "When does the client depart?"   Correct Answer: "10/13/2018"
Prediction (Sent): "598771901"   Prediction (Word): "10/13/2018"   Prediction (Joint): "10/13/2018"

Figure 2: One example prediction by our HMN for answer selection. The hop columns in reasoning module show the probabilities of each hop, and the sampling module selects the 4-max relevant sentences and feeds these selected sentences to the attention module on word-level memory. In the experiments, all the sentences in the dialogue history are indexed in reverse order to reflect their relative distance.

further improve the answer selection performance on all the datasets. Compared with the results of HMN-Sent and HMN-Word, the results also show that the joint prediction can make a better performance.

4.3 How to Select the Correct Answers on Unknown Words

Here, we try to answer two questions: (1) How does the reasoning module focus on the related sentences and predict the rare and unknown words on sentence-level memory? (2) How does the attention module focus on the correct answers and distinguish multiple rare and unknown words on word-level memory? We give a visual example of our HMN over one dialogue history for answer selection in Figure 2 to get a better understanding of the effect of each module. The example is one dialogue history from air-ticket booking domain which contains 16 sentences associated with their temporal indices.

From Figure 2, we can see that in the first hop, the reasoning module mainly focuses on the sentences 13 and 14, which have most semantic relevance to the question "When does the client depart?". As the effect of temporal encoding as Eqn. (4), the reasoning allocates more weight to the most related sentence 14 in the following hops. Another interesting result as shown in Table 2 is that MemNN and HMN-Sent represent surprising performance to predict the rare and unknown words on sentence-level memory. An explanation maybe that the way in which these methods use a simple way, rather than sophisticated LSTM or GRU, to encoding the sentence into memory as Eqn. (1). This simple sentence encoding strategy can remain the raw embedding representation of words, and MemNN and HMN-Sent utilize the transpose of the raw embedding matrix as the decoding weights to conduct answer match as Eqn. (5) in the raw embedding space. Nonetheless, sentence-level encoding may introduce other semantic information which may lead to predict an error answer, as the prediction (Sent) in the example.

After k-max pooling for sampling the most relevant sentences, a sophisticated attention mechanism is applied on the selected word-level memory. Two possible reasons make the attention successfully focus on the correct answers: One is that the probe vector $u_R^{(S)}$ as Eqn. (6), used in attention mechanism to interact with word-level memory, is generated from the reasoning module and has semantic relevance to the
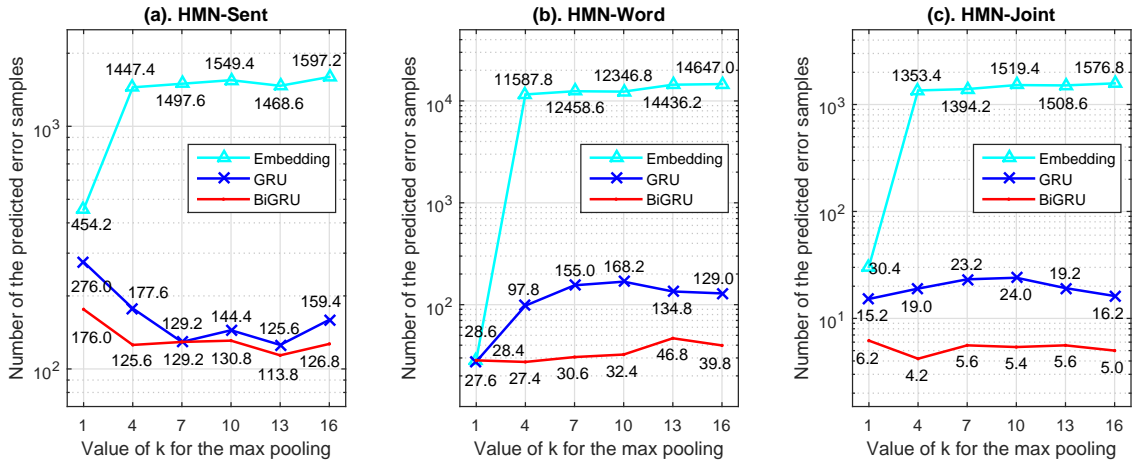
Figure 3: Answer selection evaluations of HMN-Sent, HMN-Word and HMN-Joint with different word-level memory encoding methods (BiGRU, GRU and Embedding) using various values of k on *Total*.

target answers. Another reason is that attention mechanism performs inhibitory effect on our task which can successfully filter the almost useless words, such as "time", "go" and "on" in the example. Besides the above reasons, we also investigate the influence of different word-level memory encoding methods, such as BiGRU ($\overrightarrow{GRU}(\mathbf{C}^R\bar{w}_t) + \overleftarrow{GRU}(\mathbf{C}^R\bar{w}_t)$), GRU ($GRU(\mathbf{C}^R\bar{w}_t)$) and Embedding ($\mathbf{C}^R\bar{w}_t$), by varying the values of k for max pooling, and the comparison of answer selection performance are present in Figure 3. The results show the expected effect that encoding the context information into word-level memory via BiGRU or GRU can help the attention module distinguish multiple rare and unknown words when we enlarge the value of k and introduce more unknown words to the attention mechanism.

From Figure 3, we further investigate the influence of k to the answer selection performance. We can see that the performances almost unchanged by using HMN-Joint with BiGRU when we vary the value of k. Considering that the more sentences k-max pooling samples from sentence-level memory, the more computational complexity the attention mechanism as Eqn. (6) costs on word-level memory, 4-max pooling used in our experiments is a good trade-off.

5   Related Works

Recently, lots of deep learning methods with explicit memory and attention mechanism have shown promising performance in Question Answering (QA) tasks. For example, Yu et al. (2015) applied Neural Machine Translation (NMT) (Bahdanau et al., 2015) with sophisticated attention mechanism and Neural Turing Machine (NTM) (Graves et al., 2014) with distributed external memory to solve QA tasks, and Sukhbaatar et al. (2015) designed end-to-end memory networks and introduced multi-hop reasoning component to solve various types of QA task. These representation learning based methods do not rely on any linguistic tools and can be applied to different languages or domains (Feng et al., 2015). However, most works of these deep learning based methods rarely focus on solving answer selection on unknown word problem. Recently, the unknown word problem has attracted more researchers' attention. Hermann et al. (2015) used NLP tools to recognize all the entity and establish co-references to replace all the rare entities by placeholders and trained an attention based model with softmax to predict the placeholder id. Li et al. (2016) replaced the rare words in a test sentence with similarity in-vocabulary words to solve machine translation task, where the representation of the rare words still can be learned from a large mono-lingual corpus. Gulcehre et al. (2016) utilized and extended the attention-based pointing mechanism (Vinyals et al., 2015) to point the unknown words for machine translation and text summarization. However, the sophisticated attention mechanism is applied on the all word-level representations which may result in high computational complexity, and lots of the fine-grained noise words should be filtered out by reasoning the relevant facts to the query in a high-level semantic space.

6 Conclusion

In this paper, we introduce hierarchical memory networks to solve answer selection problem on unknown words. We first encode the sentences into a sentence-level memory with temporal encoding. Then reasoning module conducts multi-hop interaction on the memory to retrieve the related sentences, and k-max pooling samples the k most related sentences. For word-level memory, BiGRU is utilized to encode the words and introduce context into the memory, then a sophisticated attention mechanism is applied on the selected word-level memory to focus the fine-grained words. We conduct answer selection experiments on four synthetic domain dialogue datasets which contain lots of unseen answers. The experimental results show that our hierarchical memory networks can achieve a satisfying performance.

Acknowledgements

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Léon Bottou. 1991. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8).

Sarath Chandar, Sungjin Ahn, Hugo Larochelle, Pascal Vincent, Gerald Tesauro, and Yoshua Bengio. 2016. Hierarchical memory networks. *arXiv preprint arXiv:1605.07427*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research (JMLR)*, 12(Aug):2493–2537.

Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 813–820. IEEE.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1693–1701.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Xiaoqing Li, Jiajun Zhang, and Chengqing Zong. 2016. Towards zero unknown in neural machine translation. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2852–2858. AAAI Press.

Rui Lin, Shujie Liu, Muyun Yang, Mu Li, Ming Zhou, and Sheng Li. 2015. Hierarchical recurrent neural network for document modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 899–907.

Mónica Marrero, Julián Urbano, Sonia Sánchez-Cuadrado, Jorge Morato, and Juan Miguel Gómez-Berbís. 2013. Named entity recognition: fallacies, challenges and opportunities. *Computer Standards & Interfaces*, 35(5):482–489.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1577–1586.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2440–2448.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3104–3112.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2692–2700.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Yang Yu, Wei Zhang, Chung-Wei Hang, and Bowen Zhou. 2015. Empirical study on deep learning models for question answering. *arXiv preprint arXiv:1510.07526*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING)*, pages 2335–2344.