Which is the Effective Way for Gaokao: Information Retrieval or Neural Networks?

Shangmin Guo[†1], Xiangrong Zeng[†1,2], Shizhu He[1], Kang Liu[1] and Jun Zhao[1]

[1]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China
[2]University of Chinese Academy of Sciences, Beijing, 10049, China
{shangmin.guo, xiangrong.zeng}@nlpr.ia.ac.cn
{shizhu.he, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

As one of the most important test of China, Gaokao is designed to be difficult enough to distinguish the excellent high school students. In this work, we detailed the Gaokao History Multiple Choice Questions(GKHMC) and proposed two different approaches to address them using various resources. One approach is based on entity search technique (IR approach), the other is based on text entailment approach where we specifically employ deep neural networks(NN approach). The result of experiment on our collected real Gaokao questions showed that they are good at different categories of questions, i.e. IR approach performs much better at entity questions(EQs) while NN approach shows its advantage on sentence questions(SQs). Our new method achieves state-of-the-art performance and show that it's indispensable to apply hybrid method when participating in the real-world tests.

1 Introduction

Gaokao, namely the National College Entrance Examination, is the most important examination for Chinese senior high school students. Every college in China, no matter it is Top10 or Top100, would only accept the exam-takers whose Gaokao score is higher than its threshold score. As there are almost 10 million students take the examination every year, Gaokao needs to be difficult enough to distinguish the excellent students. Therefore, it includes various types of questions such as multiple-choice questions, short-answer questions and essays and it covers several different subjects, like Chinese, Math, History and etc. In this work, we focus on Gaokao History Multiple Choice questions which is denoted as GKHMC. Both of the factoid question answering task and reading comprehension task are similar to GKHMC. But, the GKHMC questions have their own characteristics.

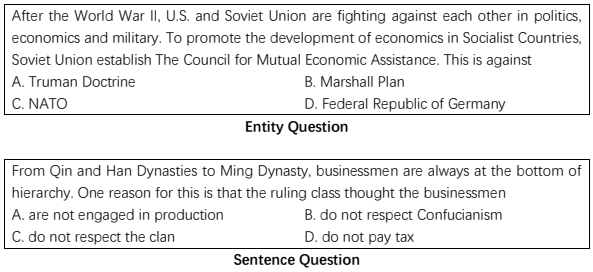

Figure 1: Examples of questions and their types. The upper one is an entity question. The lower one is a sentence question.

A multiple-choice question in GKHMC such as the examples shown in Figure 1 is composed of a question stem and four candidates. Our goal is to figure out the only one correct candidate. But, there are certain obstacles to achieve it. First, several background sentencess and a lead-in sentence conjointly constitutes the question stem, which makes these questions more complicated than former one-sentence-long factoid questions that can be handled by the existing approaches, like (Kolomiyet and Moens, 2011; Kwiatkowski et al., 2013; Berant and Liang, 2014; Yih et al., 2015). Secondly, the background sentences generally contain various clues to figure out the historical events or personages which may be the perdue key to answer the question. These clues may include Tang poem and Song iambic verse, domain-specific expressions, even some mixture of mod-

[†] Both of the two authors contributed equally to this paper.

Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers, pages 111–120, Valencia, Spain, April 3-7, 2017.

Question Type	Candidate Type	Count
EQ	Entities	160
SQ	Sentence	584
ALL	Whatever	744

Table 1: The GKHMC dataset.

ern Chinese and excerpt from ancient books and etc. The dependence of background knowledge makes the models that are designed for reading comprehension such as (Peñas et al., 2013; Richardson et al., 2013) fail. Thirdly, the diversity of candidates' granularity, i.e. candidates can either be entities or sentences, makes it harder to match the candidate and stem. So, the answer selection is disparate from the former approaches whose candidates are usually just entities. Lastly, as the candidates are already given, the answer generation step in former neural network approaches based question answering system is no longer necessary.

As mentioned above and shown in Figure 1, in accordance with candidates' granularity, the GKHMC questions can be divided into two types: entity questions(EQs) and sentence questions(SQs). Entity questions are those whose candidates are all entities, no matter they are people, dynasties, warfares or something else. And, sentence questions are those whose candidates are all sentences. We observe that such two types of questions have their own specific characteristics. Most of background sentences in EQs are description of the right candidate, so it may be particularly suitable to apply information retrieval like approach to handle them. Meanwhile, as the background sentences and lead-in sentences in SQs are more like the entailing text, these questions aren't appropriate to be addressed by lexically searching and matching. Therefore, it seems that it's more resonable to resolve SQs by using textual reasoning techniques.

In this paper, we wonder about which kind of approach is more effective for GKHMC. Furthermore, whether we should select specific method to work out different types of questions. In terms of various characteristics of GKHMC questions, we introduce two independent approaches to address them. One is based on entity search technique (IR approach) and the other is based on a text entailment approach where we specifically employ deep neural networks (NN approach). In IR approach, we use the key entities and relationships extracted from questions to form a query, then inquire this query in all the text resources to get the most relevant candidate. In NN approach, we take the question text and every candidate to form four statements respectively, then judge how possible every statement is right so that we can figure out which is most likely to be the correct answer.

To test the two approaches' performance, we collected and classified the multiple-choice questions in Gaokao test papers from 2011 to 2015 all over the country, and they are released. From the result, we find that the performance of two approaches are significantly discrepant at each kind of questions. That is, IR approach shows noticeable advantages on EQs, while NN approach performs much better on SQs. This will be further discussed in Section 4.4.

In this paper, our contributions are as follows:

- We gave a detailed description of the Gaokao History Multiple Choice Questions task and showed its importance and difficulty.
- We released a dataset[1] for this task. The dataset is manually collected and classified. All questions in the dataset are real Gaokao quesitons from 2011 to 2015.
- We introduced two different approaches for this task. Each approach achieved a promising results. We also compared this two approaches and found that they are complementary, i.e. they are good at different types of questions.
- We introduced permanent provisional memory network(PPMN) to model the joint background knowledge and sentences in question stem, and it beats existing memory networks on SQs.

2 Dataset

As described in the Introduction, we collected the historical multiple-choice questions from Gaokao all over the country in rencent five years. However, quite a lot contain graphs or tables which require the techniques beyond natural language processing(NLP). So, we filter out this part of questions and manually classified the left into two parts: EQs and SQs. The number of different kinds of questions are listed in Table 1. The examples of

[1] https://github.com/IACASNLPIR/GKHMC/tree/master/data

different types of questions translated into English are shown in Figure 1.

It is worth mentioning that there is a special type of questions on test papers named sequential questions. The candidates of this kind of questions are just some ordered numbers. Every number stands for a certain content which is given in question stem. We simply replace every sequential number in candidates with their corresponding contents. Then, we can classify these questions as EQs or SQs according to the type of contents.

We also collected a wide diversity of resources including Baidu Encyclopedia, textbooks and practice questions as our external knowledge when inquiring the generated query. Baidu Encyclopedia which is also known as Baidu Baike, is something like Wikipedia, but the content of it is written in Chinese. We denote this resource as BAIKE. The textbooks resource contains three compulsory history textbooks published by People's Education Press. We denote them as BOOK. And we gathered about 50,000 practice questions and their answers, and this is denoted as TIKU.

3 Approaches[2]

3.1 IR Approach

The GKHMC questions require figuring out the most relevant candidate to the question stem from the four given candidates. Our IR approach is inspired by this observation. The diagram of IR approach is illustrated in Figure 2.

The pipeline of IR approach is: (1) use the classifier to automatically classify the question and select the weights according to the classification result; (2) calculate the relevance scores for every candidate(we introduce three different methods with seven score functions to calculate the relevance scores) and combine them together with specific weights; (3) choose the candidate with highest score as right answer. Despite the simplicity of it, IR approach achieves a promising result in experiment.

3.1.1 Naive Bayes Classifier

We build a naive bayes classifier to classify questions. Using length of candidates, entity number of candidates and verb number of candidates as features, every question is classified as EQ or SQ. When building the classifier, we do 10-folder cross validation on the GKHMC dataset and the results are 90.00% precision and 84.38% recall in EQs and 95.79% precision and 97.43% recalls in SQs.

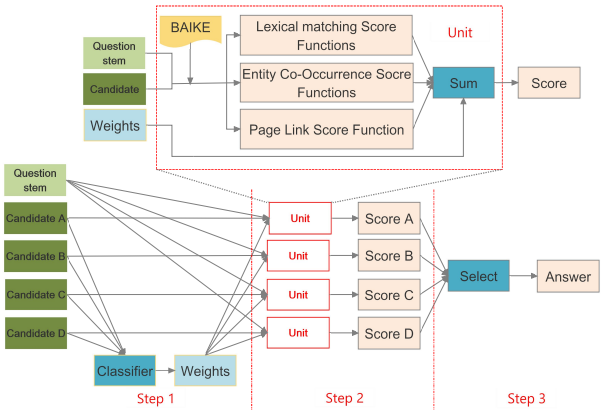

Figure 2: Pipeline of IR approach.

3.1.2 Score Functions

To calculate the relevance between question stem and candidates, we introduce three different methods with seven score functions, which are summarized in Table 2.

Lexical Matching Score: Since the correct candidate usually directly related to question stem, it's reasonable to assume that the facts in question stem may appear in documents related to them, together with the correct candidate. Here we introduce our lexical matching score functions, taking BAIKE as our external resource. The four queries are formed by each candidate and question stem separately. Then we retrieval every query and sum up the scores of the top three returned documents as the lexical matching score. We use $score_{top_i}$ to denote the score of the top i-th returned documents. $score_{top_i}$ is calculated by Lucene's TFIDFSimilarity function[3]. The lexical matching score $Score_{lexical}(candidate_k)$ is calculated as

$$Score_{lexical}(candidate_k) = \sum_{i=1}^{3}(score_{top_i}). \quad (1)$$

We build indices for BAIKE with different grains. The index built for every BAIKE document is denoted as BAIKE Document Index(BDI). The index built for every paragraph in BAIKE is denoted as BAIKE Paragraph Index(BPI). And, the index built for every sentence in BAIKE is called BAIKE Sentence Index(BSI).

[2] The codes of this project can be obtained at https://github.com/IACASNLPIR/GKHMC

[3] https://lucene.apache.org/core/

We denote the lexical matcing score function using BDI, BPI and BSI as $Score_{BDI}$, $Score_{BPI}$ and $Score_{BSI}$ respectively.

Entity Co-Occurrence Score: We also consider the relevance of entities in co-occurrence aspect. If two entities often appearing together, we assume that they are revelent. We use normalized google distance(Cilibrasi and Vitanyi, 2007) to calculate the entity co-occurrence score $Score_{co}(candidate_k)$.

$$NGD(e_i, e_j) = \frac{Max(e_i, e_j) - \log f(e_i, e_j)}{\log N - Min(e_i, e_j)} \tag{2}$$

$$Max(e_i, e_j) = \max\{\log f(e_i), \log f(e_j)\} \tag{3}$$

$$Min(e_i, e_j) = \min\{\log f(e_i), \log f(e_j)\} \tag{4}$$

$$Score_{co}(candidate_k) = -\log(NGD(e_i, e_j)) \tag{5}$$

where

$$e_i \in E_{stem}, e_j \in E_{candidate_k}.$$

In which, e_i is entity; $f(e_i)$ is the number of parts which contain entity e_i; $f(e_i, e_j)$ is the number of parts which contain both entity e_i and e_j; E_{stem} and $E_{candidate_k}$ denotes the entities in question stem and candidate.

The entity co-occurrence could be in document, paragraph or sentence, and they are donated as $Score_{BDC}$, $Score_{BPC}$ and $Score_{BSC}$ respectively.

Page Link Score: Inspired from PageRank algorithm(Page et al., 1999), we assume that entities have links to each other are relevant. Here we introduce the page link score function. We use $Link(e_i, e_j)$ to denote the number of links between entities e_i and e_j. The link score $Score_{link}(candidate_k)$ could be calculated as:

$$Score_{link}(candidate_k) = \max(Link(e_i, e_j)) \tag{6}$$

where

$$e_i \in E_{stem}, e_j \in E_{candidate_k}.$$

We only count the number of links between BAIKE documents, and it is denoted as $Score_{BDL}$

Function	Description
$Score_{BDI}$	$Score_{lexical}$ using BDI
$Score_{BPI}$	$Score_{lexical}$ using BPI
$Score_{BSI}$	$Score_{lexical}$ using BSI
$Score_{BDC}$	document level $Score_{co}$
$Score_{BPC}$	paragraph level $Score_{co}$
$Score_{BSC}$	sentence level $Score_{co}$
$Score_{BDL}$	document link score function

Table 2: Summarization of score functions.

3.1.3 Training Weights

Since we have seven score functions, we need combine them together with different weights.

For a given question, we calculate the score of every candidate as follows:

$$score_{candidate_k} = \sum_{i=i}^{7}(w_i * f_i(candidate_k)) \tag{7}$$

where $k \in \{1, 2, 3, 4\}$, f_i is one of the seven score functions and w_i is the corresponding weight. Then we normalize the scores of all candidates:

$$score_k = \frac{score_{candidate_k}}{\sum_{i=1}^{4}(score_{candidate_i})} \tag{8}$$

We suppose that the true answer of a question is the n-th candidate, where $n \in \{1, 2, 3, 4\}$. The loss of it is

$$loss_{quesiton} = -log(1 - score_n) \tag{9}$$

Now we can calculate the total loss of the dataset with M questions:

$$loss = \sum_{i}^{M}(loss_{question_i}) \tag{10}$$

All operations are derivable so that we can use gradient descent algorithm to train the weights.

3.2 NN Approach

As deep neural networks are widely used in natural language processing tasks and has gained great success, it's naturally to come up with building deep neural networks to handle GKHMC task. So, we built several deep neural networks in different structures. And, we used both TIKU and BOOK to train these models, in order to teach models not only how to answer the questions but also the historical knowledge.

To handle the joint inference between background knowledge and question stems in GKHMC

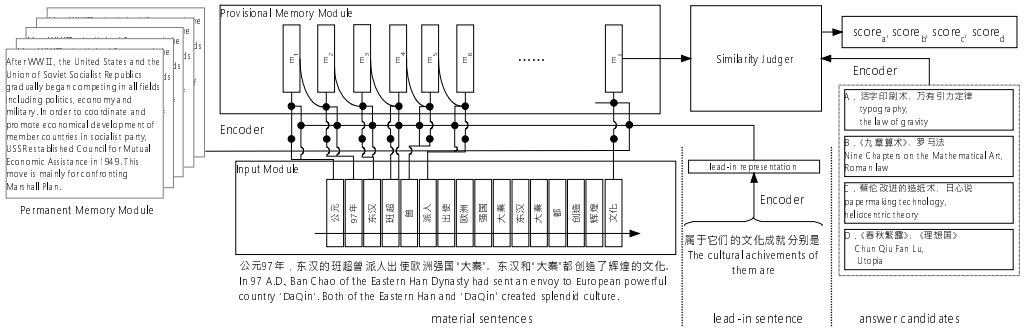

Figure 3: Diagram of PPMN. The questions hasn't been translated into English.

questions, we introduce permanent-provisional memory network(PPMN). As illuminated in Figure 3, our PPMN is composed by the following components:

1. **Permanent Memory Module** that plays the same role as a knowledge base and stores the original text from history textbooks or other relevant resource.
2. **Provisional Memory Module** that generates some contents based on the current word in background sentences, permanent knowledge and the lead-in sentence.
3. **Input Module** that reads the words sequentially in background sentences and maps them into high-dimensional vector space.
4. **Similarity Judger** that scores the similarity between the output of provisional memory and the vector representations of answer candidates.
5. **Sentence Encoder** that encodes lead-in sentence, sentences in permanent memory and answer candidates.

Permanent Memory Module: We denote the sentences encoded by sentence encoder in this module as $\{k_1, k_2, ..., k_K\}$, where K is the scale of permanent memory. The permanent memory is a constant matrix composed by the concatenation of representation vectors of these sentences, namely $[k_1; k_2; k_3; ...; k_K]$. Considering the time complexity of training PPMN, we only take the syllabus of all history courses including 198 sentences, i.e. $K = 198$, as the permanent memory. If necessary, all of the history text books can be taken into the permanent memory.

Provisional Memory Module: It first inquires the current word of background sentences in the permanent memory, then use an attention vector generated by current word and lead-in sentence as well as the following words to decide how to adjust itself. The update equations are as follow:

$$h_t = GRU(w_t, h_{t-1}) \tag{11}$$

$$p = softmax(pW^p h_t) \tag{12}$$

$$M_t = \sum_{i=1}^{K} p_i k_i \tag{13}$$

$$\begin{aligned} x = [&h_t, M_t, l, h_t \circ l, M_t \circ l, h_t \circ M_t, \\ &|h_t - l|, |M_t - l|, |h_t - M_t|] \end{aligned} \tag{14}$$

$$g = \sigma(W^g tanh(W^t x + b^t) + b^g) \tag{15}$$

$$m_t = g \circ M_t + (1 - g) \circ m_{t-1} \tag{16}$$

In the above equations, w_t denotes the t-th word in the background sentences, GRU is defined in equation (19-22), h_{t-1} and h_t are the hidden representation of w_{t-1} and w_t respectively, l stands for the lead-in sentence encoded by the sentence encoder, $\circ$ is element-wise multiplication and m_t is the computational result of current step. The final output of this module is the last provisional memory vector m_n where n is the length of background sentences.

Input Module: This module takes the same weight matrices in sentence encoder and calculates the hidden states of every word sequentially. All the words in background sentences are first mapped into the hidden states in this module and then can be taken as input by other modules. The calculation of hidden states are the same as equation(19-22).

Similarity Judger: This module takes the concatenation of the output from provisional memory and representation of answer candidate as input and use a classifier based on logistic regression to score it. The judging procedure is defined as follow:

$$\hat{p} = \sigma(W^l[m_K; a] + b^l) \quad (17)$$

$$score = softmax(\hat{p}) \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (18)$$

where W^l is a matrix that can map the concatenation vector $[m_K; a]$ into a vector $\hat{p}$ of length **2** and a stands for the answer candidate encoded by sentence encoder.

Sentence Encoder: We experimented several recurrent neural networks with different structures as the sentence encoder. Both of Long-Short Term Momery (LSTM)(Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU)(Cho et al., 2014) perform much better than the standard $tanh$ RNN. However, considering that the computation of LSTM is more complicated and time-consuming, we choose GRU as the sentence encoder. The calculation of GRU denoted as $h_t = GRU(w_t, h_{t-1})$ is as follow:

$$z = \sigma(W^z w_t + U^z w_t + b^z) \quad (19)$$

$$r = \sigma(W^r w_t + U^z w_t + b^r) \quad (20)$$

$$s = tanh(W^s w_t + U^s(r \circ h_{t-1}) + b^s) \quad (21)$$

$$h_t = (1 - z) \circ s + z \circ h_{t-1} \quad (22)$$

In the above equations, w_t is extracted from a word embedding matrix W_e initialized by word2vec(Mikolov et al., 2013) through an id number that indicates which word it is.

Loss Function: Intuitively, as we want to encourage the score as same to the true score (0 or 1) as possible, a negative log-likelihood loss function is introduced:

$$L = -log(\hat{p}y) \quad (23)$$

where y would be $[0\ 1]^\top$ if a is the right answer or $[1\ 0]^\top$ otherwise.

Optimization Algorithm: We use the AdaDelta introduced by (Zeiler, 2012) to minimize the loss L, and use back propagation through time to optimize the calculation results of intermediate results.

	Accuracy
EQ-W_{EQ}	49.38%
SQ-W_{SQ}	28.60%

Table 3: Accuracy of SQs and EQs with their corresponding best weights.

4 Experiment

4.1 Experiments of IR Approach

To find the best weights for EQ and SQ, We use TIKU as the training dataset. Using gradient descent to optimize parameters, we get the best weights for EQs and SQs separately, that is, W_{EQ} is the weight best for EQs and W_{SQ} is the weight best for SQs. We test the weights on EQs and SQs of GKHMC with their corresponding weights, and result is shown in Table 3. As we can see, with these weights, we achieve promising result.

We use GKHMC as the dataset to test the performance of IR approach with naive bayes classifier. The precision of EQs and SQs are 48.75%, 28.42% respectively. It's clear that the accuracy of both EQs and SQs decreased with automatic classification. But still, IR approach achieves much better results on EQs than SQs.

4.2 Results of NN Approach

We take some other neural network models with memory capability as our baseline models including the standard $tanh$ recurrent neural network(RNN), long-short term memory network(LSTM)(Hochreiter and Schmidhuber, 1997), gated recurrent unite(GRU)(Cho et al., 2014), end-to-end memory network(MemNN)(Sukhbaatar et al., 2015) and dynamic memory network(DMN)(Kumar et al., 2016). As for our PPMN, we summarize the syllabus of all history textbooks for senior school students to cover as much knowledge points as possible and we get 198 sentences which are taken into the permanent memory module. For all the above models, we used rmsprop(Hinton et al., 2012) with 0.001 as the learning rate to train them, the size of hidden units as well as the size of memory were both set to 400 and the size of batches were set to 1000. Also, we used dropout(Srivastava et al., 2014) to prevent the models from overfitting and the probability of it was set to 0.5. We test all these models and the results are shown in Table 4.

From the result, we observe that our PPMN

Model	EQs	SQs	All
RNN	36.25%	29.74%	31.18%
LSTM	40.63%	40.41%	40.46%
GRU	40.63%	40.24%	40.32%
MemNN	43.75%	36.13%	37.77%
DMN	44.38%	45.38%	45.16%
PPMN	**45.63%**	**45.72%**	**45.70%**
Random	25.00%	25.00%	25.00%

Table 4: Results of all neural network models.

gains best performance on all kinds of GKHMC questions and all memory-capable neural network models beat RNN. It's interesting that MemNN performs much worse than other memory-capable models on SQs whereas it shows promising capability on EQs.

4.3 Combine IR Approach and NN Approach

It can be easily observed from the above experiments that IR approach and NN approach are some kind of complementary, namely they performs better to each other on different categories of questions. So, we combine the two approaches together via a weights matrix $W^c \in \mathbb{R}^{2\times 2}$ as follows:

$$score_{EQ} = W^c_{1\cdot} \begin{bmatrix} score_{IR} \\ score_{NN} \end{bmatrix} \tag{24}$$

$$score_{SQ} = W^c_{2\cdot} \begin{bmatrix} score_{IR} \\ score_{NN} \end{bmatrix} \tag{25}$$

where the $W^c_{i\cdot}$ means the i-th row of W^c and $score_{IR}, score_{NN}$ are the scores calculated by IR and NN approaches respectively. Here, the categories of questions are given by the naive bayes classifier. The performance of combined model and its comparison to the two individual approaches are illustrated in Figure 4.

4.4 Discussion

From the global aspect, it can be easily observed that IR approach are more proficient on EQs(49.38% vs 40.63%), whereas NN approach expand superior to it on SQs(28.60% vs 40.24%). And the hybrid method composed by two approaches get the best performance(42.60%).

As for the IR approach itself, the performance on EQs is much better than on SQs. This may because that IR approach is based on the relevance between candidates and question stem. In EQs, the information given by the question stem is usually the description of the key entity which only disappeared in the right candidate. So it's easy for the correct candidate to achieve a higher relevance score than others. And, that's why IR approach achieves promising result on EQs. Whereas, in SQs, the key entity doesn't appear in any candidate. And, it needs to be inferred out from question stem. No matter in aspect of lexical matching, entity co-occurrence or page link, the relevance between question stem and correct candidate may be as low as other candidates. Therefor, it's not surprised that IR approach is not sufficient to figure out the right choice on SQs. After adding the classifier in IR approach, we notice the decrease of accuracy on both EQs and SQs. This is because of the misclassification on the questions, which demonstrates that the weights W_{EQ}, W_{SQ} are particularly efficient on EQs, SQs.

The experiment of NN approach declared that our PPMN does show its advantages on GKHMC questions. During the training, the performance of RNN model is labile, i.e. the precision are still variational when loss is convergent. In contrast, other model's performance is more stable. Hence, we consider that the memory mechanism helps model to "remember" the knowledge that appeared in the training data. Compared with the "inside"[4] memory of LSTM and GRU, the specially designed memory component in MemNN, DMN and PPMN are more powerful to find out the relationships between the question stem and answer candidates in GKHMC questions. However, the limited performance of MemNN on SQs indicates that the sequences of words in GKHMC questions are especially important for questions containing no distinct entities. Last but not least, the best performance of PPMN may due highly on the novel permanent memory module which can helps finding the implicit relationships with the stored background knowledge.

The state-of-the-art performance of hybrid method indicates that combination of IR approach and NN approach is the best strategy to address the GKHMC questions. As illustrated in Figure 4, the combined method shows its enormous advantage on EQs. This may because both character and word embedding are more sufficient to cover the lexical meaning. And, some of EQs may be more suitable to be handled as SQs. Compared to the NN approach separately, the hybrid way does

[4] We consider that the memory of LSTM and GRU are kind of stored inside the weight matrices.

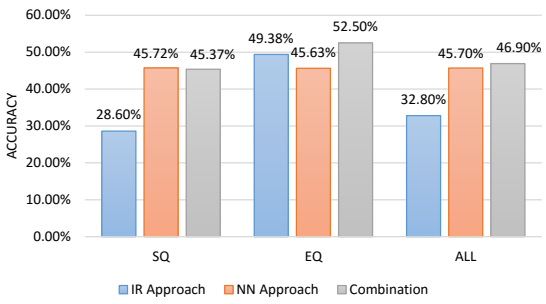

Figure 4: Result of different methods.

a little poorly on SQs, which may caused by the loss of classification.

5 Related Work

Answering real world questions in various subjects already gained attention from the beginning of this century. The ambitious Project Halo (Friedland et al., 2004) was proposed to create a "digital" Aristotle that can encompass most of the worlds's scientific knowledge and be capable of addressing complex problems with novel answers. In this project, (Angele et al., 2003) employed handcrafted rule to answer chemistry questions, (Gunning et al., 2010) took the physics and biology into account. Another important trial is solving the mathematical questions. (Mukherjee and Garain, 2008) attempted to answer them via transforming the natural language description into formal queries with hand-crafted rules, whereas recent works (Hosseini et al., 2014) started to employing learning techniques. However, none of these methods are suitable for history questions which requires large background knowledge, the same to the Aristo Challenge(Clark, 2015) focused on Elementary Grade Tests which is for 6-11 year olds.

The Todai Robot Project(Fujita et al., 2014)aims to build a system that can pass the University of Tokyo's entrance examination. As parts of this project, (Kanayama et al., 2012) mainly focus on addressing the yes-no questions via determining the correctness of the original proposition, and (Miyao et al., 2012) mainly focus on recognizing textual entailment between a description in Wikipedia and each option of question. But, these two methods are separated for different kinds of questions and none of them introduced neural network approach.

It's inevitable to compare the GKHMC with the factoid questions. (Berant and Liang, 2014) takes the question as a kind of semantic parsing which can not handle the specific expressions with lots of background knowledge. Although (Yih et al., 2015) employed knowledge base, but still failed on multiple sentences questions which is beyond the scope of semantic parsing. However, the diversity of candidates in GKHMC makes these models fail to match the question with the right candidate. Another nonnegligible task is machine comprehension, also called reading comprehension. Although in several different datasets introduced by (Smith et al., 2008; Richardson et al., 2013; Weston et al., 2015), questions are open-domain and candidates may be entities or sentences, understanding these questions don't require as much background knowledge as in GKHMC and these models cannot handle the joint inference between the background knowledge and words in questions.

We are not the first to take up the Gaokao challenge, but former information retrieval approach doesn't fit to part of the questions in GKHMC and resources in their system are limited. In contrast, we introduced two different approaches to this task, compared their performance on different types of questions, combined them and gained a state-of-the-art result.

6 Conclusion and Future Work

In this work, we detailed the multiple choice questions in subject History of Gaokao, present two different approaches to address them and compared these approaches' performance on all categories of questions. We find that the IR approach are more sufficient on EQs cause the words in these questions are usually the description of right answer, whereas the NN approach performs much better on SQs, and this may because neural network models can find out the semantic relationship between questions and candidates. When combining them together, we get the state-of-the-art performance on GKHMC, better than any individual approach. This points out that combining different approaches may be a better method to deal with the real-world questions.

In future work, we will explore whether key-value memory network proposed by (Miller et al., 2016) can help improve the performance of PPMN, what content in textbook or encyclopedia should be taken into the permanent memory, how to mathematically organize the permanent mem-

ory to make it can be reasoned on as well as whether transforming the knowledge described in natural language into formal representation is beneficial. As a long-term goal, it's necessary to introduce discourse analysis, semantic parsing to help the model truly understand the material sentences, questions and candidates.

Acknowledgments

We thank for the anonymous reviewers for helpful comments. This work was supported by the National High Technology Development 863 Program of China (No.2015AA015405) and the Natural Science Foundation of China (No.61533018). And this research work was also supported by Google through focused research awards program.

References

Jürgen Angele, Eddie Mönch, Henrik Oppermann, Steffen Staab, and Dirk Wenke. 2003. Ontology-based query and answering in chemistry: Ontonova project Halo. In *The Semantic Web-ISWC 2003*, pages 913–928. Springer, Sanibel Island, Florida, USA.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland, USA. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Rudi Cilibrasi and Paul Vitanyi. 2007. The google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):370–383.

Peter Clark. 2015. Elementary school science and math tests as a driver for AI: Take the Aristo challenge! In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI-15)*, pages 4019–4021, Austin, Texas, USA.

Noah Friedland, Paul Allen, Gavin Matthews, Michael Witbrock, David Baxter, Jon Curtis, Blake Shepard, Pierluigi Miraglia, Jurgen Angele, Steffen Staab, et al. 2004. Project Halo: Towards a digital Aristotle. *AI magazine*, 25(4):29.

Akira Fujita, Akihiro Kameda, Ai Kawazoe, and Yusuke Miyao. 2014. Overview of Todai robot project and evaluation framework of its nlp-based problem solving. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

David Gunning, Vinay Chaudhri, Peter Clark, Ken Barker, Shaw-Yi Chaw, Mark Greaves, Benjamin Grosof, Alice Leung, David McDonald, Sunil Mishra, et al. 2010. Project Halo update-progress toward digital Aristotle. *AI Magazine*, 31(3):33–58.

Geoffrey Hinton, Nirsh Srivastava, and Kevin Swersky. 2012. Lecture 6a overview of mini–batch gradient descent. *Coursera Lecture slides https://class.coursera.org/neuralnets-2012-001/lecture*.

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

Javad Mohammad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing(EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.

Hiroshi Kanayama, Yusuke Miyao, and John Prager. 2012. Answering yes/no questions via question inversion. In *Proceedings of COLING 2012*, pages 1377–1392, Mumbai, India. The COLING 2012 Organizing Committee.

Oleksandrs Kolomiyet and Marie-Francine Moens. 2011. A survey on question answering technology from an information retrieval perspective. *Information Science*, 181(24):5412–5434.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. pages 1378–1387, New York City, New York, USA. ACM.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1545–1556, Seattle, Washington, USA. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computing Research Repository*, abs/1301.3781.

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

Texas, USA. Association for Computational Linguistics.

Yusuke Miyao, Hideki Shima, Hiroshi Kanayama, and Teruko Mitamura. 2012. Evaluating textual entailment recognition for university entrance examinations. *ACM Transactions on Asian Language Information Processing (TALIP)*, 11(4):13.

Anirban Mukherjee and Utpal Garain. 2008. A review of methods for automatic understanding of natural language mathematical problems. *Artificial Intelligence Review*, 29(2):93–122.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: bringing order to the web. Technical Report 1, Stanford InfoLab.

Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, and Roser Morante. 2013. Qa4mre 2011-2013: Overview of question answering for machine reading evaluation. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 303–320. Springer.

Matthew Richardson, J.C. Christopher Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Seattle, Washington, USA. Association for Computational Linguistics.

Noah A. Smith, Michael Heilman, and Rebecca Hwa. 2008. Question generation as a competitive undergraduate course project. In *Proceedings of the NSF Workshop on the Question Generation Shared Task and Evaluation Challenge*, Arlington, Massachusetts, USA.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, Montréal, Canada.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *Computing Research Repository*, abs/1502.05698.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. 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)*, pages 1321–1331, Beijing, China. Association for Computational Linguistics.

Matthew D. Zeiler. 2012. Adadelta: An adaptive learning rate method. *Computing Research Repository*, abs/1212.5701.