Employing External Rich Knowledge for Machine Comprehension

Bingning Wang, Shangmin Guo, Kang Liu, Shizhu He, Jun Zhao
National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
{bingning.wang, shangmin.guo, kliu, shizhu.he, jzhao}@nlpr.ia.ac.cn

Abstract

Recently proposed machine comprehension (MC) application is an effort to deal with natural language understanding problem. However, the small size of machine comprehension labeled data confines the application of deep neural networks architectures that have shown advantage in semantic inference tasks. Previous methods use a lot of NLP tools to extract linguistic features but only gain little improvement over simple baseline. In this paper, we build an attention-based recurrent neural network model, train it with the help of external knowledge which is semantically relevant to machine comprehension, and achieves a new state-of-art result.

1 Introduction

Machine comprehension is one of the primary goals in Aritficial Intellegence (AI) and Natural Language Processing (NLP). However, there is no consensus on the evaluation of the *comprehensive* ability, many researchers believe that we can provide the system with multiple-choice reading comprehension test, in which the objective is gradable and requires the ability of causal reasoning, understanding of the relationship between two facts or even world knowledge. Richardson [2013] introduced MCTest-a dataset of narrative stories with a set of questions. In order to answer these questions, we must understand the context story. The dataset is open-domain and many questions require lexical matching and semantic inference to get the right answer. The stories are relatively easy that are restricted to a 7-year-old can understand, an elementary school student is able to answer all of these questions with high accuracy.

Some recent works [Sachan *et al.*, 2015; Narasimhan and Barzilay, 2015; Wang and McAllester, 2015] have been proposed to build machine comprehension models on MCTest. Due to the small size of the training data, they use a lot of NLP tools to extract sophisticated language features and use these features as similarity scores, but only get little improvement over simple unsupervised lexical method. The improvement by sophisticated NLP features is blurry, even a simple enhanced baseline system could achieve a comparative result [Smith *et al.*, 2015].

The reasons why hard crafted feature engineering systems are not able to gain significant improvement are various. On the one hand, they depend on off-the-shelf NLP tools such as dependency parser or named entity tagger, which may introduce a lot of noises; on the other hand, the existing methods heavily focus on shallow linguistic features that are confined to lexical overlap or syntactic patterns rather than semantic relatedness. However, semantic inference ability is important in machine comprehension tasks, take a question in MCTest for instance:

Q: *What was the dad's favorite food?*
A: *John liked chicken most of all.*

Since the question and answer share no words in common and their syntactic structures are not similar, in order to answer the above question we must know that *favorite* and *like...most of all* are semantically equivalent.

There are many works trying to build a model that have semantic inference ability such as Memory Networks [Weston *et al.*, 2014], a deep learning architecture that stores the context information in a dense tensor which is called *memory*, and the *memory* is retrieved to achieve AI tasks such as question answering [Sukhbaatar *et al.*, 2015; Kumar *et al.*, 2015]. Memory networks-based models achieve good result on newly published synthetic large question answering dataset called bAbi 20 tasks. Despite the success of this deep learning architecture, its structure is so complicated that a lot of labeled data is required for properly training, but in machine comprehension the labeled data are limited, so the deep neural based model may not be applied to this task directly [Kapashi and Shah, 2015].

In this work, in order to build a model on MCTest with more semantic inference ability, we cast the machine comprehension as standard question answering tasks which can be divided into two process, namely answer selection (AS) and answer generation. In the answer selection process, we build an attention-based recurrent neural network (RNN) model which has shown advantage in many semantic inference tasks such as recognizing textual entailments (RTE) and paraphrase detection [Feng *et al.*, 2015; Tan *et al.*, 2015]. In order to train this deep model properly, supervision is not only from the small golden-standard A-B-C-D answers but also from rich external AS knowledge. We customize the external AS resource to meet our task-specified requirement, we train a sophisticated LSTM model on it, and use the output of

this model as a supplementary label to our internal RNN AS model. In the answer generation stage, as we have already been provided with candidate answers, which can cast the answer generation process to answer ranking process, so we transform the question with its candidate answer into a statement, and then use recognizing textual entailment (RTE)-a well developed technique which is able to capture deep semantic relationship between two sentences-to measure the relationship between the transformed candidate statement and supporting sentence derived from AS process. Again we use external knowledge to guide this process, train a RTE model on the newly published Stanford Natural Language Inference (SNLI) dataset [Bowman *et al.*, 2015] and use this deep learning model to select semantically related sentence pair. However, there may be lexical gap between SNLI and MCTest, so we combine the external RTE model with traditional lexical model by weight: we design a similarity score that measures the lexical overlap between the supporting sentence and the statements, when the overlap is obvious, simple lexical based model could judge the relationship properly and robustly, so we set external RTE weight small; when the overlap is blurry, which means we must count on external RTE more to catch the deep semantic relationship, we set external RTE weight large. Finally, we combine answer selection process and answer generation process in a joint model and achieve a new state-of-the-art result on MCTest.

2 Task description

Machine Comprehension (**MC**) is an extension to the traditional question answering, it consists not only questions but also a document that the question is based on. One typical MC dataset is MCTest [Richardson *et al.*, 2013]. In this dataset, each question is labeled 'one' or 'multiple' to indicate the number of sentences in document that are related to this question. In addition, each question is followed with four candidate answers which may span single or multiple words. The question may be factoid or non-factoid that have many types such as *How, When, What, Why* etc. They divide the dataset into two parts, namely MC160 and MC500 which contains 160 and 500 stories respectively, and each part is divided into training, development and test sets.

3 Method

In this paper, we denote the document as D and document sentences as $\{s_0, ..., s_n\}$, each D has several questions $Q = \{q_0, ...q_i, ..., q_m\}$ and each q_i consists of 4 candidate answer $A_i = \{a_{i0}, .., a_{i3}\}$, in order to answer the question, we must choose the relevant sentences S from D and then combine it with the question to get final answer:

$$p(a|q, D) = p(S|q, D)p(a|q, S) \tag{1}$$

We define the joint probability as a product of two factors, the first one can be modeled as answer selection process: we get the sentences set S that can answer the question, while the second one can be modeled with answer generation process: generate answer from S and q. The regularized likelihood objective to maximize is:

$$\begin{aligned} &L_1(\theta; D_{train}) \\ &= \log \sum_{i=1}^{|D_{train}|} \sum_{j=1}^{|Q|} P(a_{ij}^*|q_{ij}, D_i) - \lambda g(\theta) \end{aligned} \tag{2}$$

where D_{train} is the document set and the extra g is regularization function.

For the above process, all the supervision available is correct answer choice a_{ij}^*, however, it is too weak to train a deep model with numerous parameters, so we add external knowledge as additional supervision to it. Existing AS and RTE labeled data are abundant, which are sufficient to train a complicated model that captures deep linguistic relatedness, and the external knowledge are closely related to our process in view of semantic. However, there may be lexical mismatch between them, we use simple method to customize the external data to fit our machine comprehension.

3.1 Add external Answer Selection Knowledge

We train a long and short term memory (LSTM) RNN on customized external AS data which will be illustrated in Section 4. Suppose we had external model with parameter θ_{AS}, then the AS process can be denoted as:

$$s_{AS}^* = argmax_{s \in D} P(s|q; \theta_{AS}) \tag{3}$$

Given Eq. 3, we can directly use it in our training process, remove the first part of right hand side of Eq. 1:

$$p(S|q, D) = P(s_{AS}^*|q; \theta_{AS}) \tag{4}$$

This means that we use the external AS model directly, initiate it with abundant external AS resources, then re-fit this model in MC process. However, the MCTest is small, so the model alters slightly in MC process and is mostly fit to external resource, we will illustrate this issue in experiment. In this work, we adopt a smaller neural network architecture that uses semantically expressive recurrent neural network (RNN) to model the question and candidate support sentences, and use the output of external LSTM AS model as supplemental supervision to this model.

In MCTest, the length of most sentences and questions are no more than 10 tokens, the gradient exploding or vanishing may not be an issue. So we use the simple vanilla type instead of LSTM or GRU as RNN framework:

$$\begin{aligned} &\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n] \\ &\mathbf{h}_t = \sigma(\mathbf{W}_{ih}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h) \\ &\mathbf{y}_t = \sigma(\mathbf{W}_{ho}\mathbf{h}_t + \mathbf{b}_o) \end{aligned} \tag{5}$$

where $\mathbf{x}_i$ represents the word embedding vector, and $\mathbf{W}_{ih}$, $\mathbf{W}_{hh}$, $\mathbf{W}_{ho}$ are weight matrices and $\mathbf{b}_h$, $\mathbf{b}_o$ are bias vectors, σ is activate function such as $tanh$ or $relu$.

In addition, many recent works have found that sentence representation can be enhanced with attention mechanism [Yin *et al.*, 2015; Rocktäschel *et al.*, 2015]. When building answer sentence representation, not all parts of the sentence are identical, but with weights that are related to the question representation. For example, we had a question: *who is*

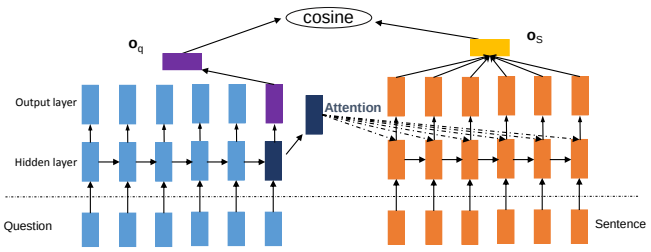

Figure 1: Attention based RNN answer selection model.

Conner's mother? and a candidate answer: *One day Conner went to street with his mother Tinna to buy some goods for the incoming Christmas*, we should pay more attention to: *his mother Tinna*; However, for the question: *Why did Conner go to street*? we should focus on *buy some goods* instead. So in our candidate support sentence representation stage, we add attention information from question to the candidate sentence output representation as follows:

$$\begin{aligned} s_t &\propto \mathbf{h}_t^T \mathbf{W}_{qo} \mathbf{h}_n^q \\ \tilde{\mathbf{h}}_t &= s_t \mathbf{h}_t \end{aligned} \tag{6}$$

where $\mathbf{h}_n^q$ is the last question hidden state and $\mathbf{W}_{qo}$ is attention weight matrix which projects the two representation in a same space. In this way, each word gets different weights in building the answer representation. For the question, we use the last output vector as its representation, for the candidate supporting sentence, we average each time-step output variable $\tilde{\mathbf{y}}_t$ to get the final sentence representation $\mathbf{o}_s$. Finally, we calculate the cosine similarity and use it as question-sentence pair score:

$$SCORE(q, s) = cosine(\mathbf{o}_q, \mathbf{o}_s) \tag{7}$$

Our internal AS model architecture is shown in Figure 1.

To train this model, we first softmax the similarity scores across document sentences, which result in the distribution $P(s|q, D; \theta_{RNN})$. In answer selection, as we focus on the most confident sentence probability, so we use cross-entropy loss function:

$$L_{AS}(q, D) = \sum_{s \in D} P(s|q, D; \theta_{RNN}) \log Q(s|q, D) \tag{8}$$

where $Q(s|q, D)$ is the supporting sentence probability that the external LSTM AS model predicts.

In this way, the external knowledge plays a role in supervise the subpart of our MC training process, and casts the first part as a multi-task learning which results in the following objective to maximize:

$$\begin{aligned} &L_2(\theta_{+AS}; D_{train}) \\ &= \log \sum_{i=1}^{|D_{train}|} \sum_{j=1}^{|Q|} [P(a_{ij}^*|q_{ij}) - \eta L_{AS}(q_{ij}, D_i)] \\ &- \lambda g(\theta_{+AS}) \end{aligned} \tag{9}$$

where the hyper-parameter η denotes the weight we learn AS process from external supervision, when set to zero, our AS model reduced to a totally self-supervised.

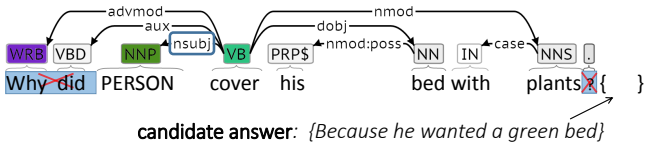

Figure 2: Dependency rules to transform a question and a candidate answer into a statement.

3.2 Add external Answer Generation Knowledge

After AS process, we get the supporting sentence probability, denote the most confident supporting sentence as s, we must combine it with question q_i to get the final answer. Like some previous works [Sachan *et al.*, 2015; Richardson *et al.*, 2013] we cast the problem as RTE. we first transform each question-answer pair into a statement, and then use an external-RTE-enhanced method to measure the relationship between the sentence and the candidate statement.

There are many ways to transform the question and answer to a statement [Cucerzan and Agichtein, 2005; Wang *et al.*, 2007; Heilman and Smith, 2010], similar with Wang [2015], we use a rule based system which convert question by its type. First of all, we use StanfordCoreNLP[1] to get the constituency tree and named entities of the question, if there exists a NNP with child nodes DT+NN in constituency parsing tree, or a named entity with type *PERSON*, we transform these words to a special symbol 'PERSON'. After this transformation, we use some rules to convert each question based on the POS of a constituents or dependency relation between two words. For example, if the question type is *why*, and the POS of the root in dependency tree is VB, and the root is located between the question word *why* and the named entity 'PERSON', we delete all the words before the 'PERSON' and add 'because'+answer after the question. One illustration for this transformation is shown in Figure 2.

After transforming the question q and answer candidate a into a statement s_q, we combine this statement with previous selected sentences and use RTE techniques to evaluate the entailment probability between them. The external RTE model will be detailed in Section 4. Denote the parameter learnt from external RTE resource is θ_{RTE}, as the external RTE model pays more attention to semantic similarity: the premise and hypothesis may have no words in common, or their linguistic representation are different, we use it as a supplement to traditional linguistic feature based model $P(s_q|s; \theta_1)$: when we can not inference this entailment from simple linguistic feature, we resort to external RTE to judge the entailment probability, which result in a combination of lexical based model and external RTE model $P(s_q|s; \theta_{RTE})$ as follows:

$$P(a|s,D) = [\beta P(s_q|s; \theta_1) + (1 - \beta) P(s_q|s; \theta_{RTE})] \tag{10}$$

The parameter β can adjust the weight between this two parts: when set to 1, it is equal to original linguistic model; when set to zero, it is totally a RTE model learnt from external knowledge. We can alter this parameter based on the lexical or

[1] http://stanfordnlp.github.io/CoreNLP/

surface structure similarity between the source and target sentence:

$$\beta = similarity(s_q^-, s) \tag{11}$$

where s_q^- denotes the transformed statement which replaces the answer with a common word 'ANSWER', when the supporting sentence and hypothesis statement have less surface overlap, which means only using the shallow linguistic feature may not reflect the real relationship between them, we must depend more on external intelligence to guide the inference process; when the overlap is obvious, only relying on the linguistic features to discriminate the difference is sufficient and reliable, so we would set β smaller accordingly. In this work, we use two set of similarity score as follows:

- **Constituency match**: In constituency tree, subtree are denoted as triplet: a parent node and its two child nodes. We add the number of triplet that **I**: the POS of three nodes are matching. **II**: the head words of parent nodes matching.
- **Dependency match**: In dependency tree, a dependency is denoted as (u,v,arc(u,v)) where arc(u,v) denote dependency relation. We add two terms similarity: **I**:u_1=u_2,v_1=v_2 and $arc(u_1, v_1)$=$arc(u_2, v_2)$.**II**: whether the root of two dependency tree matches.

For the linguistic model $P(s_q|s;\theta_1)$, we use bag-of-words match, dependency root match, constituency sub-tree match, dependency path match, name entity match and digit match as similarity feature, we also add 1-n gram match where n is the length of shorter sentence.

After answer selection and answer generation process, we combine this two part together and get the final model:

$$\begin{aligned} &P(a|q,D) = P(s|q,D;\theta_{RNN}) \\ &* [\beta P(s_q|s;\theta_1) + (1-\beta)P(s_q|s;\theta_{RTE})] \end{aligned} \tag{12}$$

and maximizing the likelihood in training data, we get:

$$\begin{aligned} &L_3(\theta_{+AS+RTE}; D_{train}) \\ &= \log \sum_{i=1}^{|D_{train}|} \sum_{j=1}^{|Q|} [P(a_{ij}^*|q_{ij}) + \eta L_{AS}(q_{ij}, D_i)] \\ &- \lambda g(\theta_{+AS+RTE}) \end{aligned} \tag{13}$$

where $P(a_{ij}^*|q_{ij})$ is equal to Eq. 12.

4 Customize and Train External Knowledge

4.1 External Answer Selection Model

Traditional AS resources are focused on factoid questions, and the question types are no more than *who, when, where, which* etc. Recently, Some new datasets have been released for not only factoid questions but also non-factoid QA questions such as QASent [Wang *et al.*, 2007], InsuranceQA [Feng *et al.*, 2015] and WIKIQA [Yang *et al.*, 2015]. However, InsuranceQA are domain-specified which focus on insurance, and in QAsent the question and answer share one or more non-stopwords. But MCTest are open domain and the answers may have no words in common with supporting sentence. So we choose WIKIQA, a large annotated AS corpus based on Wikipedia, which is closely related to MCTest narrative style and contains not only factoid questions but also non-factoid questions, and the size of this dataset are relatively large compared with previous datasets (more than 20K sentences).

To customize the WIKIQA, we remove all questions that have no right answer and truncate sentence length to 40 tokens. In addition, there are many proper nouns which may mislead the attention model to focus on entity in WIKIQA, so we replace all named entities in question or answer with their types (i.e. PERSON, ORGANIZATION, LOCATION). We train an attention-based model similar with [Tan *et al.*, 2015]. As the sentence in WIKIQA is relatively long so we use LSTM to capture long distance information. For the question part, we use max-pooling over all the hidden states to obtain the final question representation $\mathbf{o}_q$. For the answer part, after calculating the hidden LSTM state, we add attention information from question as follows:

$$\begin{aligned} \mathbf{m}_{a,q}(t) &= tanh(\mathbf{W}_{am}\mathbf{h}_a(t) + \mathbf{W}_{qm}\mathbf{o}_q) \\ s_{a,q}(t) &\propto exp(\mathbf{w}_{ms}^T \mathbf{m}_{a,q}(t)) \\ \tilde{\mathbf{h}}_a(t) &= \mathbf{h}_a(t) s_{a,q}(t) \end{aligned} \tag{14}$$

where $\mathbf{W}_{am}, \mathbf{W}_{qm}$ and $\mathbf{w}_{ms}$ are attention parameters. Then we average question-attended answer hidden representation $\tilde{\mathbf{h}}_a(t)$ to get the final answer representation $\mathbf{o}_a$. Finally, we adopt the Geometric mean of Euclidean and Sigmoid Dot (GESD) proposed in [Feng *et al.*, 2015] to measure the similarity between the two representations:

$$GESD(x,y) = \frac{1}{1+||x-y||} . \frac{1}{1+exp(-\gamma(xy^T+c))} \tag{15}$$

GESD shows advantage over cosine similarity in experiment. Finally, we adopt a max-margin hinge loss as objective:

$$L = max\{0, M - GESD(\mathbf{o}_q, \mathbf{o}_{a_+}) + GESD(\mathbf{o}_q, \mathbf{o}_{a_-})\} \tag{16}$$

where a_+ is right answer and a_- is wrong answer candidate, M is the predefined margin value.

4.2 External RTE Model

Given a pair of sentence, RTE model could help us judge whether there exits *ENTAILMENT, NEUTRAL* or *CONTRADICTION* relationship between them. Recently proposed Stanford Natural Language Inference (SNLI) dataset[Bowman *et al.*, 2015] is annotated by human annotators, and magnitudes larger than previous data (more than 500k sentence pairs). So we use this dataset as external RTE resources.

As in SNLI most of the sentences are depiction of a picture, so its representation are sometimes like: *A/The ... woman/girl/boy...*, but in MCTest the named entities are mostly proper nouns, so we transform each sentence as follows: if there exists *DT+NN* in dependency tree and the dependent type is *det*, we replace all words between *DT* and *NN* with a special word 'ENTITY'. The RTE model we adopt is similar with the AS model described in previous section, with exception that the premise and hypothesis representation are projected to same space as follows:

$$\mathbf{o}_r = softmax(\mathbf{W}_o^T[\mathbf{W}_p^T\mathbf{o}_p, \mathbf{W}_h^T\mathbf{o}_h]) \tag{17}$$

where the output $\mathbf{o}_r$ is the distribution over three classes. We use cross-entropy as loss function.

System	MC160			MC500		
	One	Multiple	All	One	Multiple	All
Sliding Window	64.73	56.64	60.41	58.21	56.17	57.09
Sliding Window+Word Distance	76.78	62.50	67.50	64.00	57.46	60.43
Sliding Window+Word Distance+RTE	76.78	62.50	69.16	68.01	59.45	63.33
[Kapashi and Shah, 2015]	-	-	36.0	-	-	34.2
[Narasimhan and Barzilay, 2015]	82.36	65.23	73.23	68.38	59.90	63.75
[Wang and McAllester, 2015]	84.22	67.85	75.27	72.05	67.94	69.94
[Smith *et al.*, 2015]	78.79	**70.31**	75.77	69.12	63.34	65.96
[Sachan *et al.*, 2015]	-	-	-	67.65	**67.99**	67.83
without External Knowledge ($\beta = 1, \eta = 0$)	40.39	37.94	39.08	38.40	33.13	31.33
without External AS knowledge ($\eta = 0$)	41.07	40.63	40.83	49.63	28.05	32.83
without External RTE knowledge ($\beta = 1$)	74.11	64.06	68.75	57.72	50.91	53.00
Final Model	**88.39**	64.84	**75.83**	**79.04**	63.51	**70.96**

Table 1: Result on MCTest test data

	Answer Selection		RTE
	MAP	MRR	Accuracy
State of the Art	0.6921	**0.7108**	**0.835**
Our method	**0.6936**	0.7094	0.829

Table 2: The external model results, for the answer selection in WIKIQA, the state-of-the-art result is achieved by an attention-based convolutional neural network [Yin *et al.*, 2015]. For the RTE, Rocktäschel [2015] train an LSTM attention model to get the state-of-the-art result.

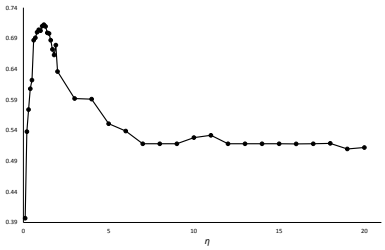

Figure 3: The result of different η value in MC500

5 Experiment

5.1 External Knowledge

For the AS setup, we filter out the questions in WIKIQA that have no right answer and use the removed sentence as negative example. We use the off-the-shelf 100 dimensional word embedding from word2vec[2], LSTM hidden state is activated by $tanh$ and hidden vector length is 178, it has been proved by Pascanu [2013] that the vanishing and exploding gradient issues in RNN depends on the largest singular value, so we initiate all hidden-to-hidden weight matrix in LSTM by fixing its largest singular value to 1. For regularization, we add L_2 penalty with a coefficient of 10^{-5} . Dropout [Srivastava *et al.*, 2014] is further applied to both weights and embeddings. All hidden layers are dropped out by 30%, and embeddings 40%. The max-margin M is set to 0.12 by developmental set behavior. We evaluate AS based on MAP (mean average precision) and MRR (mean reciprocal rank).

For the RTE model, as the dataset consist of more than 570K pairs, so we set word vector size to 300 and hidden variable length to 161, we adopt L_2 and dropout regularization same as AS model, our model is optimized by SGD. In addition, we truncate all the sentence pairs to max length of 50 tokens. We evaluate the system in terms of accuracy. The result of our external model compared with state-of-the-art results is shown in Table 2.

5.2 Machine Comprehension

We evaluate our system on MC160 and MC500. For attention based RNN, we set the hidden variable size to 45 and pre-trained word2vec embedding size to 50, we fix word embedding during training. We share the sentence and question RNN parameters for better behavior and faster convergency [Tan *et al.*, 2015], we use $relu$ as RNN inner state activation function, we use RMSprop to update parameter. We do not filter stopwords during AS process, for the similarity score β in answer generation, we normalize it to have a distribution from 0 to 1. Adding L_1 or L_2 regulation did not show significant improvement so we set them to zero.

For the question with multiple support sentences, we select top 2 sentences from our AS model because we find that most 'Multiple' questions are answerable by up to 2 sentences. In addition, MCTest contains a lot of negative questions which requires the model to choose the most unlikely candidate, in these situation we totally depend on external RTE(set β to zero) and choose the candidate that has most *contradiction* probability. We set η to 1.282 which is tuned on development data. In addition, as noun phrase is a significant feature in narrative, especially in AS procedure, so we refer all pronoun to its anaphora.

Baselines: We take 8 baselines for comparison. The first three baselines are proposed by MCTest dataset provider [Richardson *et al.*, 2013], (1) Sliding window uses a window over document to get bag of words similarity between question+hypothesized answer and document. (2) Word Distance simply subtracted from the sliding-window score to get the final result. (3) The third baseline uses off-the-shelf RTE system to get the relationship between the whole document and transformed statement. (4) Kapashi [2015] uses memory network to train a model on MCTest (5) Narasimhan[2015] builds a discourse parser to model the relationship between two selected sentences. (6) Wang[2015] uses a lot of features

[2]https://code.google.com/archive/p/word2vec/

such as frames arguments matching and syntax matching as similarity scores. (7) Smith [2015] enhances the sliding windows baseline by setting the windows size from 2 to 30 and average all these scores. (8) Sachan[2015] model the alignment between document sentences and statement as hidden variable, then learns a structural SVM on it to minimize the loss on training data.

We also report the accuracy for a self-supervised model without external knowledge, a model without external AS knowledge (i.e. $\eta = 0$), and a model without external RTE knowledge(i.e. $\beta = 1$). The result is shown in Table 1.

6 Analysis

To evaluate the improvement by importing external AS knowledge, we change η from 0 to 20, which means we rely more on external knowledge to train AS model, the result is shown in Figure 3 .

As can be seen from the figure, when set η to zero, our RNN AS model behavior on test set is very poor, which means our deep RNN model overfitting seriously, however, when set it too large, the result becomes poorer because our RNN is subject to external knowledge supervision most of all, and the lexical mismatch may lead our RNN model departing from the final MC target, since the WIKIQA sentences are longer compared to our sentence, the model is prone to choose longer sentence. When set to a proper value, the supervision from subsequent MC gold-answer and external AS knowledge could complement with each other.

In addition, we also try to not use lexical similarity for β but rather a predefined fixed value. When set it to 1, the answer generation process is reduced to lexical matching. However, when the support sentence and transformed statement have no words in common, it is really hard to get true answer, an example is shown below:

Sentence: *oh, the rocks and water and tiny castle were all there alright, but the pretty blue fish with the long shiny tail was nowhere to be seen.*
Statement: *Mr. Fish was missing from the fish bowl.*

The lexical matching method focuses on whole sentence, so it can not infer that it is the fish missing from the bowl. However, when the overlap is very obvious, rely on external RTE may cause problem, for example:

Sentence:(1) *Billy was a big bully* (2)*Billy was like a king on the school yard, a king without a queen.*
Statement:*Billy was like a king*

For the two sentences AS model chooses, the lexical overlap model could select the second one with high confidence, but the external RTE model selects the first sentence more relevant. This failure may attribute to external RTE model misunderstands the idiomatic phrase 'be like a' in sentence (2) and treats it as verb on the complicated context.

Our model performs especially good at single fact question, however, as our model did not measure sentence relationship when dealing with multiple fact question, so the performance on multiple sentence questions does not very well. In addition, when the question requires understanding the narrative flow of the story, such as "*What was the first character mentioned in the story*", our system can not solve it correctly as we deal with each sentence-statement pair independently.

7 Related work

Since the MCTest dataset [Richardson *et al.*, 2013] been proposed, there are many works focusing on machine comprehension tasks as has been described in Section 5. Similar with Narasimhan [2015], we combined the two MC sub-process in a joint model, but we did not use hard crafted linguistic features but rather a deep neural model supervised by MC target and additional resources. Wang [2015] uses a lot of hand craft features to measure similarity, but this method are so sophisticated which is hard to re-implement. Smith [2015] use an simple enhanced baseline model and achieves a comparative result, they claim the improvement that NLP features imported to MC are limited. Sachan [2015] also uses RTE to measure the relationship between candidate statement and document sentence, but they did not select the support sentence and model each sentence with same weight, this is inappropriate as most of the document sentences are unrelated with the question. Kapashi [2015] tried to build a deep learning architecture(i.e. Memory Networks) in MCTest, but the parameter space is so huge and the labeled data are so small, they claimed the model overfitting seriously no matter what regulation strategy they use. As far as we know, our model is the first deep learning architecture in MCTest which behaves better than simple unspervised baseline.

Weston [2015] proposed a QA dataset called bAbi, which is divided into 20 tasks such as counting and time manipulation, but this dataset are synthetic and the vocabulary size is no more than 50. In addition, the inference requirement is simple and one may build a rule-based system to solve it correctly. Hermann [2015] also proposed a dataset for text comprehension and build an attention-based model on it, however, the questions in this dataset are all declarative with a slot for the right answer, and all answers are limited to entities, which may not be a good evaluation for comprehension ability such as causal reasoning.

Berant [2014] also proposed a reading comprehension dataset and build an event structure model on it, but the context story are confined to biological process so the event style is limited, and their model requires annotation of the event in the story which may not applied to open-domain MC.

8 Conclusion

In this paper, for the subpart of MC process, we build an attention-based RNN model for AS process and add external RTE model to answer generation process. To build a deep learning model from limited data, we train the model with supervision from external knowledge, customize the external resources and add it to MC process properly. The experiment result shows that our model achieves especially well in single support fact question. Error analysis suggests that modeling the relationship between sentences in AS can yield improvement on this task. In addition, the counting problem and common sense problem are really hard to tackle which requires deeper linguistic analysis. In the future, we plan to build a RTE model that could model multiple sentences together for inference tasks.

Acknowledgments

The work was supported by the Natural Science Foundation of China (No.61533018), the National High Technology Development 863 Program of China (No.2015AA015405) and the National Natural Science Foundation of China (No.61272332). And this research work was also supported by Google through focused research awards program.

References

[Berant *et al.*, 2014] Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D Manning. Modeling biological processes for reading comprehension. In *EMNLP*, 2014.

[Bowman *et al.*, 2015] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[Cucerzan and Agichtein, 2005] Silviu Cucerzan and Eugene Agichtein. Factoid question answering over unstructured and structured web content. In *TREC*, volume 72, page 90, 2005.

[Feng *et al.*, 2015] Minwei Feng, Bing Xiang, Michael R Glass, Lidan Wang, and Bowen Zhou. Applying deep learning to answer selection: A study and an open task. *arXiv preprint arXiv:1508.01585*, 2015.

[Heilman and Smith, 2010] Michael Heilman and Noah A Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019. Association for Computational Linguistics, 2010.

[Hermann *et al.*, 2015] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692, 2015.

[Kapashi and Shah, 2015] Darshan Kapashi and Pararth Shah. Answering reading comprehension using memory networks. 2015.

[Kumar *et al.*, 2015] Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. *arXiv preprint arXiv:1506.07285*, 2015.

[Narasimhan and Barzilay, 2015] Karthik Narasimhan and Regina Barzilay. Machine comprehension with discourse relations. In *53rd Annual Meeting of the Association for Computational Linguistics*, 2015.

[Pascanu *et al.*, 2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1310–1318, 2013.

[Richardson *et al.*, 2013] Matthew Richardson, Christopher JC Burges, and Erin Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, volume 1, page 2, 2013.

[Rocktäschel *et al.*, 2015] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*, 2015.

[Sachan *et al.*, 2015] Mrinmaya Sachan, Avinava Dubey, Eric P Xing, and Matthew Richardson. Learning answerentailing structures for machine comprehension. In *Proceedings of ACL*, 2015.

[Smith *et al.*, 2015] Ellery Smith, Nicola Greco, Matko Bosnjak, and Andreas Vlachos. A strong lexical matching method for the machine comprehension test. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, September 2015.

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

[Sukhbaatar *et al.*, 2015] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, pages 2431–2439, 2015.

[Tan *et al.*, 2015] Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.

[Wang and McAllester, 2015] Hai Wang and Mohit Bansal Kevin Gimpel David McAllester. Machine comprehension with syntax, frames, and semantics. *Volume 2: Short Papers*, page 700, 2015.

[Wang *et al.*, 2007] Mengqiu Wang, Noah A Smith, and Teruko Mitamura. What is the jeopardy model? a quasi-synchronous grammar for qa. In *EMNLP-CoNLL*, volume 7, pages 22–32, 2007.

[Weston *et al.*, 2014] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[Weston *et al.*, 2015] Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. Towards ai-complete question answering: a set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

[Yang *et al.*, 2015] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Citeseer, 2015.

[Yin *et al.*, 2015] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193*, 2015.