Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Towards Reading Comprehension for Long Documents

Yuanxing Zhang, Yangbin Zhang, Kaigui Bian and **Xiaoming Li**
Peking University, China
{longo,yangbin_zhang,bkg,lxm}@pku.edu.cn

Abstract

Machine reading comprehension has gained attention from both industry and academia. It is a very challenging task that involves various domains such as language comprehension, knowledge inference, summarization, etc. Previous studies mainly focus on reading comprehension on short paragraphs, and these approaches fail to perform well on the documents. In this paper, we propose a hierarchical match attention model to instruct the machine to extract answers from a specific short span of passages for the long document reading comprehension (LDRC) task. The model takes advantages from hierarchical-LSTM to learn the paragraph-level representation, and implements the match mechanism (i.e., quantifying the relationship between two contexts) to find the most appropriate paragraph that includes the hint of answers. Then the task can be decoupled into reading comprehension task for short paragraph, such that the answer can be produced. Experiments on the modified SQuAD dataset show that our proposed model outperforms existing reading comprehension models by at least 20% regarding exact match (EM), F1 and the proportion of identified paragraphs which are exactly the short paragraphs where the original answers locate.

1 Introduction

Reading comprehension of human languages is one of the most challenging tasks for machines. Collecting clues from passages or contexts, machines are expected to understand the natural language, locate the probable span of the passages, and answer various types of questions [Hermann *et al.*, 2015]. Recently, machine reading comprehension has been successfully deployed in the E-commerce systems such as Alibaba to quickly respond to questions from customers based on the detailed introduction pages.

Studies on reading comprehension have made great progress in the past few years. Attention-based deep neural networks learn to read documents and answer complex questions with minimal prior knowledge of language structure [Hermann *et al.*, 2015]. Besides, the memory network jointly considers inference components as well as a long-term memory component that acts as a dynamic knowledge base [Weston *et al.*, 2014], which helps bridge the gap of requiring co-reference and involving more verbs and nouns. In addition, co-attention mechanisms have shown prominent performance for locating the proper span corresponding to the questions and predicting final answers on machine reading comprehension (MRC) [Cui *et al.*, 2017; Seo *et al.*, 2016; Wang and Jiang, 2017]. Recently, the SLQA+ model and the r-net+ [Wang *et al.*, 2017] model, proposed by Alibaba iDST NLP and Microsoft Research Asia respectively, have exceeded human-level performance according to the SQuAD's exact match (EM) metric [Rajpurkar *et al.*, 2016].

Tasks of reading comprehension are much more difficult in many real-world scenarios. Figure 1 shows an example of the long document reading comprehension (LDRC) task. A decent question and answer (QA) system is expected to provide the precise answers to various types of questions over an entire document, rather than a short paragraph of at most several hundreds of tokens. For instance, the context in SQuAD has at most 800 words in each paragraph, while a single article has 150 paragraphs with 17174 tokens at most. Concatenating all paragraphs to one continuous context would not be a wise solution, as it would be difficult to focus on the targeted information, and prone to being misguided in the attention calculation. Hierarchical structure [Li *et al.*, 2015] has been proposed to build embeddings for various structural levels, such as document level, paragraph level and even sentence level. However, implementing a large number of zero paddings and constructing dynamic graph may degrade the precision and efficiency of the machine reading comprehension system for LDRC tasks.

Intuitively, it can be observed that human written documents can always be split into paragraphs deliberately, where each paragraph expresses relatively independent meaning among the whole document. Therefore, we adopt a hierarchical LSTM model [Li *et al.*, 2015] to learn the representation for each paragraph. We also implement a bi-directional match-LSTM [Wang and Jiang, 2016] to learn a hidden representation of each paragraph in the document for reflecting its relation to the question. The relationship between each paragraph and the question can then be figured out, providing us with the evidence to choose the best paragraph to find the answer. Moreover, the Pointer Net model [Vinyals *et al.*, 2015]

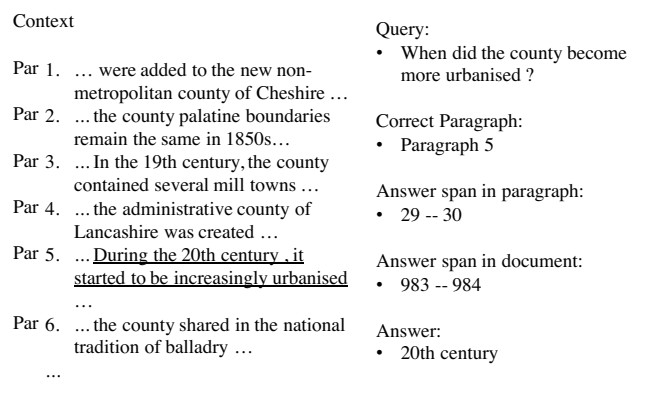


Figure 1: An example of long document reading comprehension task.

is leveraged to find the most potential paragraph, while the softmax outputs can also provide the top-K best paragraphs, similar to how a recommender system works [Resnick and Varian, 1997]. Then, we can apply the state-of-the-art models to focus on these short paragraphs and analyze the most appropriate span of answers. The SQuAD dataset provides up to 150 paragraphs for each article, which can be used for the LDRC task. Experiments on SQuAD show that our proposed model performs better than simply applying single-paragraph MRC models such as Match-LSTM with Pointer [Wang and Jiang, 2017], AoA network [Cui *et al.*, 2017] with Pointer, BiDAF [Seo *et al.*, 2016], as well as structural model, i.e., Hierarchical-LSTM [Li *et al.*, 2015].

Our contributions can be summarized as follows.

- We leverage the characteristic that documents are usually divided to paragraphs, and combine the match-LSTM network and the hierarchical-LSTM network to find the best paragraph for answering queries related to long documents. Then, the LDRC task can be simplified to reading comprehension task on single-paragraph.
- The proposed MH-LSTM model has achieved an exact match score of 58.83% and an F1 score of 67.15% on the modified development set of SQuAD, where it correctly locates 80.91% of the original paragraphs of the answers. It outperforms existing approaches for reading comprehension on single-paragraph.

2 Related Work

Three Types of Reading Comprehension Tasks: The first type comes from multiple choice questions, where the models are asked to choose one answer from several candidates, such as MCT [Richardson *et al.*, 2013]. The second type is cloze style comprehension, either with given candidate tokens such as CBT [Hill *et al.*, 2016] or without any candidate such as CNN/Daily Mail [Hermann *et al.*, 2015]. The third type includes analyzing the context and answering the questions without options. Dataset such as SQuAD [Rajpurkar *et al.*, 2016] is usually taken as the standard for comparing the reading comprehension capabilities of machine and human. In this paper, we study the reading comprehension task for long documents and experiments are conducted over SQuAD.

Single-paragraph Reading Comprehension: Reading comprehension on single-paragraph has gained efforts from researchers. Existing work includes developing various arrangements of layers of neural networks such as recurrent neural network (RNN) [Sutskever *et al.*, 2014], convolution neural network (CNN) [Hu *et al.*, 2014], fully-connected layers [Botha *et al.*, 2017], and designing different attention mechanisms [Hermann *et al.*, 2015] and memory mechanisms [Weston *et al.*, 2014], for providing better answers to the questions. In particular, the co-attention model on both context and query is one of the preeminent solutions to this task. For example, BiDAF [Seo *et al.*, 2016] is a hierarchical multi-stage architecture for modelling the representation of the context paragraph at different levels of granularity. The model constructs the character embedding layer and the word embedding layer as inputs to map each word to a vector space. Then, the attention flow layer is set to couple the query and context vectors and produce query-aware feature vectors for each token in the context. The attention flow layer connects to multi-layer bi-LSTM and finally the model provides answers to the queries.

Hierarchical Structure: Regarding long documents with individual paragraphs, most models for single-paragraph comprehension fail to work well. Treating the whole document as one paragraph would increase the possibility of vanishing or exploding gradient during training and make it tough to calculate accurate attention. The hierarchical structure is proposed to learn the representation of different levels of the documents. Hierarchical-LSTM [Li *et al.*, 2015] is a typical hierarchical structure to learn the representation for paragraphs and documents. Together with attention, the learned embeddings can reserve most of the context and approximately recover the origin sentences.

Natural Language Inference: With the help of the hierarchical representation, the potential span of the answer can be fixed to one or several short paragraphs. Hence, it is necessary to find the relationship between each paragraph and the corresponding query. Natural language inference (NLI) is the task of determining whether one can infer one hypothesis from another premise [MacCartney, 2009]. Several attention models have been verified to be promising in addressing this problem, such as match-LSTM [Wang and Jiang, 2016]. Match-LSTM processes the premise and the hypothesis using two LSTMs. Then the model calculates the attention from premise to hypothesis, and feeds them to another LSTM network. The state of the last cell of this LSTM network can be taken as the evidence for natural language inference.

3 Match-based Hierarchical Attention Model

Finding the appropriate paragraph or several sentences as the evidence is the core of our approach for the LDRC task. Given a query and a corresponding document context, it is necessary to learn the hierarchical representation of different granularity and figure out the inference between each paragraph and the query. The structure of our model is illustrated in Fig. 2, abbreviated as MH-LSTM.

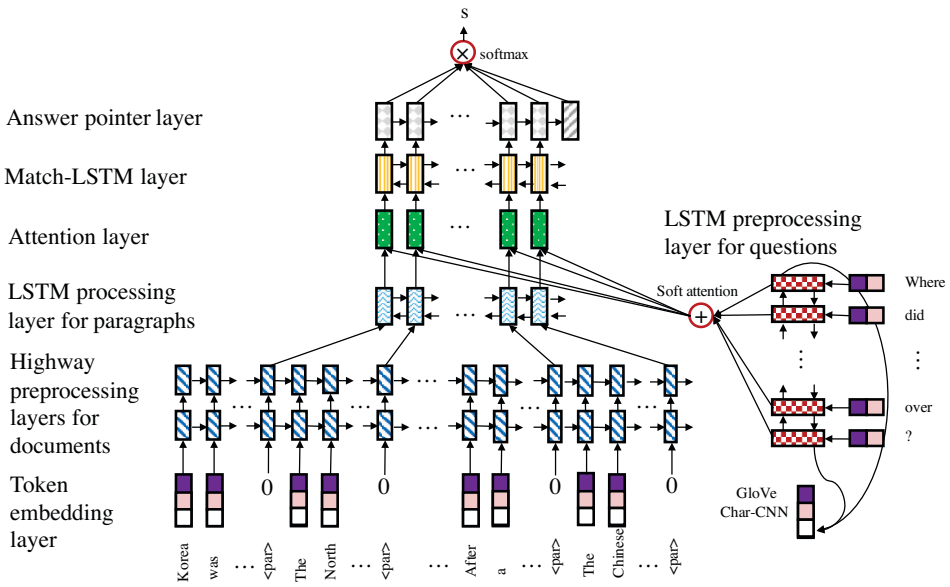


Figure 2: The structure of the MH-LSTM model. The model takes the word-level embeddings and the character-level embeddings as input. A two-layer highway network is used to learn the representations of each word in documents, and a bi-LSTM model to learn the representations of queries. The representations of each paragraph are extracted by mask, which are then fed into a match-LSTM network. The outputs are generated by a softmax pointer.

3.1 Problem Formulation

The problem is formulated as follows. Given a full document $\widetilde{P_*}$ with several paragraphs (which we refer to as a piece of context, and a query $\widetilde{Q_*}$ related to the context), the preprocessing script on the dataset would label the paragraph $\widetilde{a_*}$ where the answers locate. Therefore, the dataset can be represented by a tuple $\{(\widetilde{P_n}, \widetilde{Q_n}, \widetilde{a_n})\}_{n=1}^{N}$, where N is the number of tuples in the dataset.

3.2 Design of the Model

Token Embedding Layer

The token embedding layer maps each token in the document to a high-dimensional vector space. The words in the document are first tokenized into individual tokens. Specifically, we add a paragraph break "<par>" at the end of each paragraph. Then the tokens are transformed to the corresponding embeddings by two mechanisms. The first mechanism comes from word-level embedding. We use pre-trained word vectors, GloVe [Pennington *et al.*, 2014], as the fixed word embedding of each word, where the missing words from GloVe are denoted by "<unk>". The second mechanism computes the character-level embedding. Characters of every word are embedded into vectors and fed into CNN as introduced by Kim [Kim, 2014]. The outputs of the CNN are max-pooled over the entire word as the fixed-size embedding. The word-level embedding and character-level embedding are then concatenated to represent the embedding of each token, which has a dimension of d.

Preprocessing Layer for Query

Let Q denote the embeddings of words in a query from token embedding layer with length $|Q|$, where $Q =< \mathbf{q}_1, \ldots, \mathbf{q}_{|Q|} >\in \mathbb{R}^{d\times|Q|}$. We first obtain representation vectors of each word in the query as well as the whole query sentences by feeding the tokens to a one-layer bi-LSTM, i.e.,

$$\mathbf{H}^Q, \mathbf{q}_{rep} = \overleftrightarrow{\text{LSTM}}(Q), \tag{1}$$

where $\mathbf{H}^Q = [\mathbf{y}_1^Q, \ldots, \mathbf{y}_{|Q|}^Q] \in \mathbb{R}^{h^Q\times|Q|}$ represents the matrix of the outputs of bi-LSTM, where each column represents the concatenation of forward and backward outputs of the corresponding word from the LSTM network. Besides, $\mathbf{q}_{rep}$ concatenates the forward and the backward final states of bi-LSTM, and h^Q is the hidden size of the bi-LSTM layer.

Preprocessing Layer for Context

Let P' denote the embeddings of each word in the document from token embedding layer with length $|P'|$, where $P' =< \mathbf{p}'_1, \ldots, \mathbf{p}'_{|P'|} >\in \mathbb{R}^{d\times|P'|}$. To discover the relationship between each word in the document and the query, we concatenate $\mathbf{q}_{rep}$ to the embeddings of each word in the document, i.e., $P =< \mathbf{p}_1, \ldots, \mathbf{p}_{|P|} >\in \mathbb{R}^{(d+h^Q)\times|P|}$, where $\mathbf{p}_t = \mathbf{p}'_t||\mathbf{q}_{rep}, t \in [1, |P|]$ and "||" represents the concatenation. As the length of a document is usually far longer than a paragraph, it would be much more difficult for training [Glorot and Bengio, 2010]. Here, we follow the approach proposed by Zilly [Zilly *et al.*, 2017] and leverage a two-layer recurrent highway network (RHN) for the preprocessing of context. Specifically, let $\mathbf{W}^P_{H,T,C} \in \mathbb{R}^{h^P\times(d+h^Q)}$ represent the weight matrices of the nonlinear transformation. Let $\mathbf{R}^P_{H_l,T_l,C_l} \in \mathbb{R}^{h^P\times h^P}$ and $\mathbf{b}^P_{H_l,T_l,C_l} \in \mathbb{R}^{h^P\times h^P}$ denote the weights and biases of the transformation, and carry gates at layer $l \in \{1, 2\}$. Then the calculation of the recurrence

process could be expressed by

$$\mathbf{s}^P_{l,t} = \mathbf{h}^P_{l,t} \cdot \mathbf{t}^P_{l,t} + \mathbf{s}^P_{l-1,t} \cdot \mathbf{c}^P_{l,t}, \tag{2}$$

$$\mathbf{h}^P_{l,t} = \tanh(\mathbf{W}^P_H \mathbf{p}_t \mathbb{I}_{\{l=1\}} + \mathbf{R}^P_{H_l} \mathbf{s}^P_{l-1,t} + \mathbf{b}^P_{H_l}), \tag{3}$$

$$\mathbf{t}^P_{l,t} = \sigma(\mathbf{W}^P_T \mathbf{p}_t \mathbb{I}_{\{l=1\}} + \mathbf{R}^P_{T_l} \mathbf{s}^P_{l-1,t} + \mathbf{b}^P_{T_l}), \tag{4}$$

$$\mathbf{c}^P_{l,t} = \sigma(\mathbf{W}^P_C \mathbf{p}_t \mathbb{I}_{\{l=1\}} + \mathbf{R}^P_{C_l} \mathbf{s}^P_{l-1,t} + \mathbf{b}^P_{C_l}), \tag{5}$$

where $\mathbb{I}_{\{\cdot\}}$ is the indicator function, and $\sigma(\cdot)$ represents the sigmoid function. The initial states $\mathbf{s}^P_{0,t}$ are set to $\mathbf{p}_{t-1}$. We collect the outputs of layer 2, i.e. $\mathbf{s}^P_{2,t}$, and take them as the representations of each token in the context, i.e.,

$$\mathbf{H}^P = [\mathbf{s}^P_{2,1}, \ldots, \mathbf{s}^P_{2,|P|}] \in \mathbb{R}^{h^P \times |P|}. \tag{6}$$

Hierarchical Mask Layer

As we mainly focus on the representation of paragraph-level, we should extract the evident vectors that contain sufficient knowledge of the paragraphs. Hence, we set up a mask layer, which only reserves the outputs from the RHN whose input corresponds to the paragraph break "<par>". The vectors are then fed into a bi-LSTM network to obtain the representations of each paragraph, i.e.,

$$\mathbf{H}^M = \overleftrightarrow{\text{LSTM}}(\text{mask}(\mathbf{H}^P)) = [\mathbf{y}^M_1, \ldots, \mathbf{y}^M_{|M|}], \tag{7}$$

where $|M|$ is the length of the reserved representations after being masked, and it is exactly the number of paragraphs in the document.

Attention Layer

This layer is designed to interpret the degree to which the network attends to every paragraph in the document. Inspired by the word-by-word attention mechanism, we concentrate on the attention from each paragraph to every token in the query. Specifically, the attention of the t-th paragraph on the i-th token in the query in the forward direction can be calculated by:

$$\overrightarrow{\mathbf{G}}_{t,i} = \tanh(\overrightarrow{\mathbf{W}^{AQ}} \mathbf{y}^Q_i + \overrightarrow{\mathbf{W}^{AM}} \mathbf{y}^M_t + \overrightarrow{\mathbf{b}^A}), \tag{8}$$

$$\overrightarrow{\alpha}_t = \text{softmax}(\overrightarrow{\mathbf{w}}^\intercal \overrightarrow{\mathbf{G}}_{t,*}), \tag{9}$$

where $\overrightarrow{\alpha}_t$ is the normalized attention from paragraph t to every token in the query.

Match-LSTM Layer

We are expected to focus on the textual entailment by treating the query as a premise and the paragraphs as hypotheses. The match-LSTM layer is used for providing the evidence for the inference [Wang and Jiang, 2017], which is actually a one-layer LSTM network. Let $\overrightarrow{\mathbf{H}^O} = [\overrightarrow{\mathbf{y}^O}_1, \ldots, \overrightarrow{\mathbf{y}^O}_{|M|}]$ denote the hidden outputs of the match-LSTM, satisfying:

$$\overrightarrow{\mathbf{y}^O}_t = \overrightarrow{\text{LSTM}}(\overrightarrow{\mathbf{z}}_t, \overrightarrow{\mathbf{y}^O}_{t-1}), \tag{10}$$

where

$$\overrightarrow{\mathbf{z}}_t = \begin{bmatrix} \mathbf{y}^M_t \\ \mathbf{H}^Q \overrightarrow{\alpha}^\intercal_t \end{bmatrix}. \tag{11}$$

In order to improve the attention between each paragraph and query, we take the outputs from the match-LSTM into account for Eqn.(8), i.e., modified to

$$\overrightarrow{\mathbf{G}}_{t,i} = \tanh(\overrightarrow{\mathbf{W}^{AQ}} \mathbf{y}^Q_i + \overrightarrow{\mathbf{W}^{AM}} \mathbf{y}^M_t + \overrightarrow{\mathbf{W}^{AO}} \overrightarrow{\mathbf{y}^O}_t + \overrightarrow{\mathbf{b}^A}). \tag{12}$$

Similarly, we build a match-LSTM in the backward direction, denoted as $\overleftarrow{\mathbf{H}^O} = [\overleftarrow{\mathbf{y}^O}_1, \ldots, \overleftarrow{\mathbf{y}^O}_{|M|}]$, to obtain the precise representation based on the surrounding contextual paragraphs. Finally, the outputs of the bi-directional match-LSTM network would be the concatenation of the outputs of the forward match-LSTM and the outputs of the backward match-LSTM, i.e., $\mathbf{H}^O = [\overrightarrow{\mathbf{y}^O}_1 || \overleftarrow{\mathbf{y}^O}_1, \ldots, \overrightarrow{\mathbf{y}^O}_{|M|} || \overleftarrow{\mathbf{y}^O}_{|M|}]$.

Output Pointer Layer

The final outputs should indicate the probability of a paragraph to be the most appropriate one for answering the query. To address this problem, we leverage the novel attention mechanism to model these probabilities, i.e., the outputs of the match-LSTM layer connect to a sequence-to-sequence model [Sutskever *et al.*, 2014]:

$$[\mathbf{y}^e_1, \ldots, \mathbf{y}^e_{|M|}], \mathbf{e}_{rep} = \text{ENCODER}(\mathbf{H}^O), \tag{13}$$

$$\mathbf{y}^d = \text{DECODER}(\mathbf{e}_{rep}), \tag{14}$$

where $\mathbf{y}^e_*$ denotes the hidden outputs of the encoder, and $\mathbf{y}^d$ represents the hidden output of the decoder. Finally, we focus on the output distribution over each paragraph, i.e., for the t-th paragraph,

$$u_t = \mathbf{v}^\intercal \tanh(\mathbf{W}^e \mathbf{y}^e_t + \mathbf{W}^d \mathbf{y}^d). \tag{15}$$

We then take softmax normalization $\mathbf{s}$ on the distribution to represent the probability of each paragraph to be the best candidate:

$$\mathbf{s} = \text{softmax}(u_*). \tag{16}$$

Answer to the Query

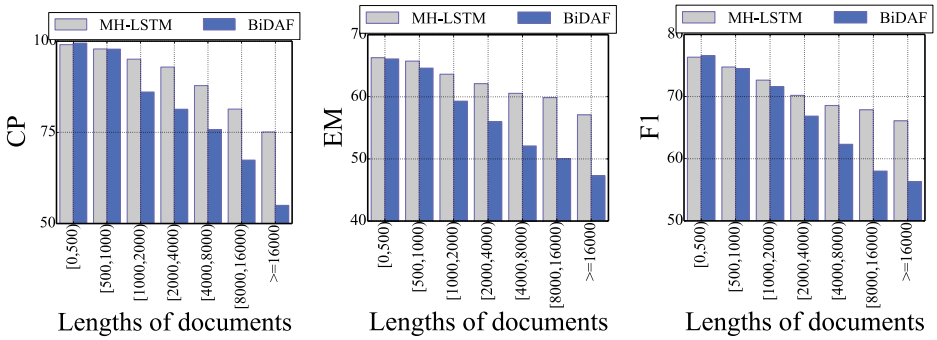
The most appropriate paragraph would be targeted by $\mathbf{s}$. Then we could leverage the reading comprehension model for single-paragraph on the selected paragraph to generate the answer. In this paper, we use BiDAF that has been verified to be effective on the original SQuAD dataset.

Training

We define the training loss (to be minimized) as the sum of the negative log probabilities that the softmax outputs can correctly identify the short paragraphs where the original answers locate, averaged over all examples:

$$L(\theta) = -\frac{1}{N} \sum_{n=1}^{N} \log p(\widetilde{a_n} | \widetilde{P_n}, \widetilde{Q_n}) = -\frac{1}{N} \sum_{n=1}^{N} \log \mathbf{s}_{\widetilde{a_n} | \widetilde{P_n}, \widetilde{Q_n}}, \tag{17}$$

where θ represents the trainable variants in the model as mentioned in this section, and $\mathbf{s}_{\widetilde{a_n} | \widetilde{P_n}, \widetilde{Q_n}}$ represents the $\widetilde{a_n}$-th softmax value from the output layer, given $\widetilde{P_n}, \widetilde{Q_n}$ as inputs.



(a) CP (b) EM (c) F1

Figure 3: The performance of MH-LSTM with various lengths of documents, compared with BiDAF for single-paragraph.

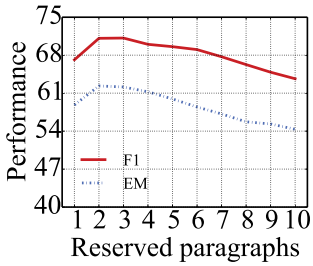


Figure 4: The performance of MH-LSTM with various numbers of reserved paragraphs.

4 Experiments

4.1 Dataset

We use the SQuAD v1.1 dataset to conduct the experiments. Passages in SQuAD come from 536 articles in Wikipedia covering a wide range of topics. Each article contains up to 150 paragraphs, and each paragraph contains up to 785 words. Each paragraph has around 5 queries with the corresponding answers, and the answer to each query has a prelabeled position in the paragraph. Totally, there are 107,785 queries.

We first merge all paragraphs of every article to substitute the original context. Specific paragraph breaks are added as mentioned in section 3, and the answer spans are also updated. Meanwhile, we label the original paragraph, which would be used in the LDRC task. After merging the paragraphs, there would be at most 17174 tokens in each passage. We split the training set into a training set (81,398 tuples) and a validation set (4,285 tuples). The development set of SQuAD is treated as the test set in our experiments.

4.2 Experiment Settings

We use the pre-trained GloVe [Pennington *et al.*, 2014] parameters with 300-dimension embeddings to initialize the model. The word embeddings would not be changed during training. Besides, the hidden size of every recurrent network involved in our model is fixed as 150 in single direction. The labelled paragraphs where the original answers occur are transformed to one-hot vectors. The learning rate is dynamically updated according to Adam optimization [Kingma and Ba, 2014]. Besides, in order to avoid overfitting or out of memory, the gradient descent is updated by minibatch of 32 instances. We do not observe performance gains using dropout on this task.

We mainly focus on three metrics in the experiments: (1) the percentage of exact match (EM) with the ground truth answers, (2) the word-level F1 score when comparing the tokens in the predicted answers with the tokens in the ground truth answers, and (3) the proportion of identified paragraphs which are exactly the short paragraphs where the original answers locate (CP). The experiments run on two GTX1080Ti GPUs.

4.3 Results

We establish two categories of models, i.e., non-hierarchical models and hierarchical models, to examine their performances on the LDRC task. Both categories of models take the word-level and character-level embeddings as inputs.

For the non-hierarchical models, we set up one baseline model and select three reputable models from the SQuAD leader board. The one-layer vanilla LSTM that takes the tokens from an entire document as inputs is used as the baseline model, and then the outputs connect to a Pointer network to find the beginning and the end of the span of the answer. We examine the performance of match-LSTM with answer pointer [Wang and Jiang, 2017] on the LDRC dataset. We also modified the AoA model [Cui *et al.*, 2017], i.e., replacing the dot product in the last step of attention-over-attention process by a fully-connected feedforward network that takes the concatenation of attentions as inputs and obtains 150 output neurons. Therefore, we would get one representative vector for each paragraph as its representation. Similarly, a Pointer network is leveraged to seek the answer. Meanwhile, we directly implement BiDAF [Seo *et al.*, 2016] on this LDRC task.

For the hierarchical models, we first compare the performance of our model to the hierarchical-LSTM [Li *et al.*, 2015]. Specifically, we calculate the paragraph-level representation rather than the sentence-level representation. A bi-LSTM is implemented, and the hidden state of the last cell is taken as the representation of the query. We use dot product to depict the relationship between each paragraph and the query. The paragraph with the highest probability are then

Model Type	Model	EM	F1	CP
Non-Hierarchical	LSTM+Pointer	11.99	15.17	18.04
	Match-LSTM+Pointer	36.74	44.00	59.28
	AoA+FC+Pointer	30.82	37.17	50.63
	BiDAF	49.71	58.35	67.27
Hierarchical	Hierarchical-LSTM+BiDAF	52.60	61.45	71.33
	MH-LSTM+BiDAF	58.83	67.15	80.91

Table 1: Results of different models on LDRC task.

Description	Context & QA pair
The MH-LSTM finds the correct paragraph, while single BiDAF fails to find the appropriate paragraph.	**Relevant:** *Par 1* (T, M): … the Yuan dynasty bore the Mandate of Heaven, following the Song dynasty and preceding the Ming dynasty… *Par 2* (B): … Before the end of the Yuan dynasty, 14 leaders of the Sakya sect had held the post of Imperial Preceptor, … **Query:** What dynasty came before the Yuan? **Answer:** *BiDAF*: Sakya sect *MH-LSTM+BiDAF*: Song dynasty
Both models fail to find the original paragraph, but they produce the correct answer based on information from other paragraphs.	**Relevant:** *Par 1* (T): … Oxygen is the most abundant element by mass in the Earth's crust as part of oxide compounds such as silicon dioxide… *Par 2* (M): … The rest of the Earth's crust is also made of oxygen compounds … *Par 3* (B): … Oxygen constitutes 49.2% of the Earth's crust by mass and is the major component of the world's oceans … **Query:** What element makes up almost half of the earth's crust by mass? **Answer:** *BiDAF*: Oxygen *MH-LSTM+BiDAF*: oxygen
Although MH-LSTM finds the correct paragraph, it produces the wrong answer. However, single BiDAF finds answer in another paragraph.	**Relevant:** *Par 1* (B): …The textbook examples are cydippids with egg-shaped bodies and a pair of retractable tentacles… *Par 2* (T, M): … Cydippid ctenophores have bodies that are more or less rounded, sometimes nearly spherical and other times more cylindrical or egg-shaped… Some species of cydippids have bodies that are flattened to various extents… **Query:** Cydippid are typically what shape? **Answer:** *BiDAF*: egg-shaped *MH-LSTM*: flattened
Neither of the two models find the correct paragraph, nor do they produce the correct answer.	**Relevant:** *Par 1* (T): … to make way for the Eldon Square Shopping Centre, including all but one side of the original Eldon Square itself… *Par 2* (B): … The largest of these is the Eldon Square Shopping Centre, one of the largest city centre shopping complexes in the UK… *Par 3* (M): … Haymarket bus station and Eldon Square bus station. **Query:** What is in Eldon Square? **Answer:** *BiDAF*: UK *MH-LSTM*: bus station

Figure 5: Four typical instances in the LDRC task. The "T" and the green dashed rectangles represent the true answer; The "M" and the red rectangles represent the answer from the MH-LSTM with BiDAF; The "B" and the blue dotted rectangles represent the answer from BiDAF for single-paragraph.

treated as the context for BiDAF to extract the most relevant answer.

Our results are summarized in Table 1. Due to the fact that there are some tokens from the queries which only appear once in the document, it is straightforward to find the corresponding spans of answers to the queries that contain these categories of tokens. However, the long document increases the difficulty for LSTM to learn the answer to queries. The performance of BiDAF model for single-paragraph decreases by about 26%, where the EM decreases from 66% to 50%, and F1 decreases from 76% to 58%. We find that the hierarchical models have greater probability to find the correct paragraph than the non-hierarchical models. The hierarchical-LSTM could help increase the probability of choosing the correct paragraph to 71%, while our MH-LSTM could further improve the CP to nearly 81%. The EM reaches 59% by our proposed model, indicating that the MH-LSTM is effective for the LDRC task.

4.4 Further Analysis

Performance with Various Lengths of Document

We examine the performance of our MH-LSTM on documents with various lengths. As there are only 48 articles with 48 different lengths in the development set of SQuAD, we construct "sub-articles" for experiments by enumeration. Specifically, we randomly enumerate several sets of paragraphs (which must include the correct paragraph) from the same article, treated as the contexts for queries. Therefore, we have a new LDRC dataset with more context-query-answer tuples. The sub-articles are divided into 7 groups by their lengths: less than 500 tokens, 500–1000, 1000–2000, 2000–4000, 4000–8000, 8000–16000 tokens, and more than 16000 tokens. We set the number of tuples of sub-articles in each group to 1000. We repeat the random enumeration 20 times and calculate the average value for each performance metric. The performance metrics, i.e., CP, EM and F1, on each group of data are plotted in Fig. 3. We observe that performance of our model decreases slightly, compared with that of directly using the BiDAF model for single-paragraph.

Performance with Various Numbers of Paragraphs

We also examine the performance of our MH-LSTM on documents with various numbers of potential paragraphs. We seek to analyze the contribution of the number of reserved paragraphs for comprehension. Intuitively, the more paragraphs we reserve, the greater probability we would have to reach the correct paragraph. However, due to the fact that longer documents would decrease the performance of comprehension model for single-paragraph, it is necessary to restrict the length of the reserved passages. We evaluate the performance of our model when we reserve several paragraphs with the highest values of the output neurons rather than reserve only one paragraph, as shown in Fig. 4. Both EM and F1 increase until two or three reserved paragraphs, and then they begin to drop as the number of paragraphs increase. Up to 3% increase of EM and F1 could be obtained when reserving one or two more paragraphs.

4.5 Case Study

We found certain instances that can support the importance of LDRC and the strength of our proposed model. We illustrate four typical instances in Fig. 5.

The first instance indicates the condition where our proposed MH-LSTM helps find the correct paragraph and then the single passage MRC model produces the expected answer. Without locating paragraph, the BiDAF model would fail to find the correct answer. However, in the second instance when both models fail to find the pre-labeled paragraph, the two models could still answer correctly due to the supplementary information in other paragraphs of the document. Similarly, the whole document would help single passage MRC model to solve some queries although the model may choose a "wrong" paragraph. For these queries, the model should have produced wrong answers in the pre-labeled paragraphs, such as the third instance. Meanwhile, the forth instance implies that the two models may fail to find the correct paragraph, or produce the expected answer, which requires future efforts to address this problem.

5 Conclusion

Reading comprehension is a challenging task for machines. Recently, existing work has focused on the reading comprehension task for single-paragraph or short passages. How-

ever, these approaches fail to work well on the long documents. In this paper, we study the reading comprehension task for long documents to find paragraph-level representation and natural language inference. We design a model that takes advantages from hierarchical-LSTM and match-LSTM to find the most appropriate paragraph for the answers. The SQuAD dataset is modified for experiments, where the original development set is treated as the test set. Results show that the proposed MH-LSTM can outperform some existing known comprehension models for single-paragraph by at least 20% regarding exact match (EM), F1 and the proportion of identified paragraphs which are exactly the short paragraphs where the original answers locate (CP).

Acknowledgments

This work is partially supported by the National Key Research and Development Program No. 2017YFB0803302, the National 973 Grant No. 2014CB340405, and the National Natural Science Foundation of China under Grant Nos. 61572051 and 61632017.

References

[Botha *et al.*, 2017] Jan A. Botha, Emily Pitler, Ji Ma, Anton Bakalov, Alex Salcianu, David Weiss, Ryan McDonald, and Slav Petrov. Natural language processing with small feed-forward networks. In *Proceedings of the 2017 EMNLP*, pages 2879–2885, Copenhagen, Denmark, September 2017. ACL.

[Cui *et al.*, 2017] Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th ACL (Volume 1: Long Papers)*, pages 593–602, Vancouver, Canada, July 2017. ACL.

[Glorot and Bengio, 2010] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th AISTATS*, volume 9, pages 249–256, Sardinia, Italy, 13–15 May 2010. PMLR.

[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 *NIPS*, pages 1693–1701, 2015.

[Hill *et al.*, 2016] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. *ICLR*, 2016.

[Hu *et al.*, 2014] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *NIPS*, pages 2042–2050, 2014.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 EMNLP*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[Kingma and Ba, 2014] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2014.

[Li *et al.*, 2015] Jiwei Li, Thang Luong, and Dan Jurafsky. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of the 53rd ACL and the 7th IJCNLP (Volume 1: Long Papers)*, pages 1106–1115, Beijing, China, July 2015. ACL.

[MacCartney, 2009] Bill MacCartney. *Natural language inference*. Stanford University, 2009.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 EMNLP*, pages 1532–1543, 2014.

[Rajpurkar *et al.*, 2016] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 EMNLP*, pages 2383–2392, Austin, Texas, November 2016. ACL.

[Resnick and Varian, 1997] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[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 3, page 4, 2013.

[Seo *et al.*, 2016] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *International Conference on Learning Representations*, 2016.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Vinyals *et al.*, 2015] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *NIPS*, pages 2692–2700, 2015.

[Wang and Jiang, 2016] Shuohang Wang and Jing Jiang. Learning natural language inference with lstm. In *Proceedings of the 2016 NAACL: Human Language Technologies*, pages 1442–1451, San Diego, California, June 2016. ACL.

[Wang and Jiang, 2017] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. *ICLR*, 2017.

[Wang *et al.*, 2017] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th ACL (Volume 1: Long Papers)*, pages 189–198, Vancouver, Canada, July 2017. ACL.

[Weston *et al.*, 2014] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *International Conference on Learning Representations*, 2014.

[Zilly *et al.*, 2017] Julian Georg Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. Recurrent highway networks. In *Proceedings of the 34th ICML*, volume 70, pages 4189–4198, Sydney, Australia, 06–11 Aug 2017. PMLR.