Co-Attention Hierarchical Network: Generating Coherent Long Distractors for Reading Comprehension

Xiaorui Zhou,¹ Senlin Luo,¹ Yunfang Wu²

¹School of Information and Electronics, Beijing Institute of Technology
²MOE Key Lab of Computational Linguistics, School of EECS, Peking University
{zhouxr, luosenlin}@bit.edu.cn, wuyf@pku.edu.cn

Abstract

In reading comprehension, generating sentence-level distractors is a significant task, which requires a deep understanding of the article and question. The traditional entity-centered methods can only generate word-level or phrase-level distractors. Although recently proposed neural-based methods like sequence-to-sequence (Seq2Seq) model show great potential in generating creative text, the previous neural methods for distractor generation ignore two important aspects. First, they didn’t model the interactions between the article and question, making the generated distractors tend to be too general or not relevant to question context. Second, they didn’t emphasize the relationship between the distractor and article, making the generated distractors not semantically relevant to the article and thus fail to form a set of meaningful options. To solve the first problem, we propose a co-attention enhanced hierarchical architecture to better capture the interactions between the article and question, thus guide the decoder to generate more coherent distractors. To alleviate the second problem, we add an additional semantic similarity loss to push the generated distractors not semantically relevant to the article and question. The better we understand ourselves, the easier it becomes to live up to our chances for a rich and fulfilling life.

Introduction

Reading comprehension (RC) is an advanced cognitive activity of human beings, which involves interpretation of the text and making complex inferences (Chen, Bolton, and Manning 2016). The most popular form of assessment for reading comprehension is Multiple Choice Question (MCQ), since MCQs have many advantages including quick evaluation, less testing time, consistent scoring, and automatic evaluation (RAO CH and Saha 2018). Besides the article itself, a MCQ consists of three elements: (i) stem, the question body; (ii) key, the correct answer; (iii) distractors, alternative answers used to distract examinees from the correct answer. The effectiveness of MCQs depends not only on the validity of the question and the correct answer, but also on the quality of distractors (Goodrich 1977). Among all methods for creating good MCQs, finding reasonable distractors is crucial and usually the most time-consuming (Liang et al. 2018).

In real examinations, a question for reading comprehension usually requires summarizing the article, or making inferences about a certain detail in the article. Figure 1 shows an example of MCQ from the RACE (Lai et al. 2017) dataset, which records real English exams for middle and high school Chinese students. Different from the SQuAD (Rajpurkar et al. 2016) dataset which is widely used in RC research, in RACE the answer is a newly-generated se-

Figure 1: An example from our dataset, with the generated distractors of our strongest baseline and our proposed model, we use colors and underlines to indicate the semantic connection between MCQ segments and the article text.
quence with the length of a sentence other than a text span extracted from the original text. Accordingly, the distractor should also be a sequence of words that is fluent and grammatical. More importantly, the distractor should be coherent with the question, and semantically relevant to the article. We call this type of distractor long distractor to distinguish it from the word-level or phrase-level distractor in fill-in-the-blank (Liang et al. 2017) or cloze MCQs. In this paper, we investigate the task of generating coherent long distractors for reading comprehension MCQs.

Traditionally, distractor generation is a component of an automatic MCQ generation system, and seldom has been taken out as a separate task. The process of generating MCQs (RAO CH and Saha 2018) usually consists of: (i) sentence selection: select a sentence that contains a questionable fact as a candidate for generating MCQs. (ii) Key selection: determine which word, n-gram, or phrase in the selected sentence should be blanked out. (iii) Question formation: transform a declarative sentence into the interrogative form. (iv) Distractor generation: generate distractors that are able to confuse the examinees. Approaches for generating distractors may utilize WordNet (Miller 1995) to find synonyms or other related words as distractors, or use an existing domain-specific ontology to find related phrases (Stasaski and Hearst 2017; Araki et al. 2016), or adopt other similarity-based methods like word embedding similarities (Guo et al. 2016; Kumar, Banchs, and D’Haro 2015), and co-occurrence likelihoods (Hill and Simha 2016), etc. As we can see, the traditional MCQ generation is in a pipeline fashion, which requires human-designed features and external knowledge bases. Moreover, all the above mentioned approaches are based on entity relations, which can only generate word-level or phrase-level distractors and are not able to generate long distractors.

Recently, deep neural models like Seq2Seq (Sutskever, Vinyals, and Le 2014) have achieved great success in a lot of Natural Language Processing (NLP) tasks, including machine translation, text summarization, headline generation and story generation. The recent work (Gao et al. 2019) employs a hierarchical encoder-decoder framework and proposes a static attention to notice the sentences related to the question and penalize the sentence related to the answer to generate long distractors on RACE dataset, and the proposed model outperforms several baselines, achieving a BLEU-4 score of 6.47 for the first distractor.

However, there is still much room for improvement in generating distractors. First, all previous proposed neural-based models adopt a simple Seq2Seq structure to build a direct mapping from article to distractor, which fails to model the interactions between the article and question, so the generated distractors tend to be too general that is not relevant to the theme of the article or not consistent with the question context. Figure 1 shows an example of distractors generated by our strongest baseline, which contain key words irrelevant to the article and question. As has been proved previously in the RC task (Seo et al. 2016; Xiong, Zhong, and Socher 2016), capturing the complex interactions between article and question can improve performance in selecting correct answer. Second, previous proposed methods did not emphasize the relevance between the generated distractor and article. As a result, some of generated distractors are semantically far away from the article, thus fail to form a set of meaningful and educative options.

In this paper, we propose a Co-attention Hierarchical Network to generate distractors. The basic framework is a hierarchical encoder-decoder network with dynamic attention, which first obtains word-level hidden representations and then based on them to obtain the sentence-level representations. The dynamic attention combines word-level and sentence-level attention at each decoding time step. Based on this framework, we propose to incorporate co-attention mechanism, i.e. article-to-question and question-to-article, to allow the encoder better capture the rich interactions between the article and question. Further, we introduce a semantic similarity loss between the generated distractor and article into the original loss function, to guide the decoder to generate distractors that are more relevant to the article content.

We conduct extensive experiments on the challenging RACE dataset. Comparing with different approaches, our full model obtains the best results across most metrics (BLEU and ROUGE) for all three distractors. It outperforms the existing best method (Gao et al. 2019), achieving a new state-of-the-art performance of BLEU-4 7.01 for the first distractor. The ablation study validates the effectiveness of our two proposed components. Further human evaluation demonstrates that our model can generate better quality distractors that are more consistent with the article and have stronger distracting ability.

Proposed Framework

Notations and Task Definition

For each sample in our dataset, we have an article that contains $k$ sentences $T = (s_1, s_2, \ldots, s_k)$ and each sentence $s_i = (w_{i,1}, w_{i,2}, \ldots, w_{i,l_i})$ is a word sequence where $l_i$ denotes the length of it. We also have a question $Q = (q_1, q_2, \ldots, q_m)$ for each sample, where $m$ denotes the length of the question. Our task is to generate a wrong option (distractor) $D = (d_1, d_2, \ldots, d_z)$ where $z$ is the distractor sequence length.

Formally, we define the distractor generation (DG) task as generating the best wrong option in reading comprehension, which is the conditional log-likelihood of the predicted distractor $D$, given the article $T$ and question $Q$, such that:

$$
\hat{D} = \arg\max_D \log P(D|T, Q).
$$

Model Overview

In this paper, we propose a co-attention hierarchical network to generate distractors, as shown in Figure 2.

The encoder consists of three layers: 1) Hierarchical Encoding Layer maps input word embeddings to their word-level and sentence-level hidden representations. 2) Co-Attention Layer couples the question and article representations, and produces a set of question-aware feature
vectors for each sentence in the article. 3) **Merging Layer** then merges the sentence-level representations and question-aware feature vectors to get the final sentence representations.

In the decoding phase, the word-level hidden representations and the final sentence representations are referenced at every decoding time step to calculate a hierarchical attention score. We employ a language model to compress the question into a fixed-length vector to initialize the decoder state, to make the distractor grammatically consistent with the question.

Moreover, we add a semantic similarity loss into the standard loss function, to enable the generated distractor more related to the article content.

**Encoding Article and Question**

**Hierarchical Article Encoder.** For each sentence \( s_i = (w_{i,1}, w_{i,2}, \ldots, w_{i,t}) \) in the article, we use a bidirectional LSTM (Hochreiter and Schmidhuber 1997) (denoted as LSTM\(_{\text{enc}}\)) with hidden size \( r \) to encode this sequence into hidden representations

\[
h_{i,t}^{w} = \text{LSTM}_{\text{enc}}^{w}(h_{i,t-1}^{w}, w_{i,t}). \tag{2}
\]

The vector output at the ending time step of this sequence is used as the embedding to represent the entire sentence:

\[
e_i = h_{\text{end}_w}^{w} = h_{i,t}. \tag{3}
\]

In order to build representation \( e_T \) for the current article \( T \), another layer of LSTM (denoted as LSTM\(_{\text{enc}}\)) with hidden size \( r \) is placed on top of all sentence embeddings, computing representations sequentially for each time step:

\[
h_{t}^{s} = \text{LSTM}_{\text{enc}}^{s}(h_{t-1}^{s}, e_t). \tag{4}
\]

Similarly, we use the final time step sentence-level representation \( h_{\text{end}_s}^{s} \) to represent the entire document:

\[
e_T = h_{\text{end}_s}^{s} = h_{k}. \tag{5}
\]

Let \( H^* \in \mathbb{R}^{r \times k} \) to denote the sentence-level representations of the article, where \( H_{i,t}^* = h_{i,t}^s \).

By utilizing this hierarchical structure, we decompose a long document into a two-level connection of relatively short sequences, avoiding directly encoding the whole document through a single LSTM.

**Question Encoder.** We use a bidirectional LSTM to encode the question sequence \((q_1, q_2, \ldots, q_m)\) into hidden representation. In our implementation, we share the same LSTM with the word-level LSTM used in article encoding, so

\[
h_{q}^{q} = \text{LSTM}_{\text{enc}}^{q}(h_{q-1}^{q}, q_t). \tag{6}
\]

We use \( U^* \in \mathbb{R}^{r \times m} \) to denote all word-level representations of the question, where \( U_{i,t}^* = h_{i,t}^q \).

**Co-Attention between Article and Question**

To model the complex interactions between the article and question, we adopt a co-attention mechanism (Seo et al. 2016), which is computed in two directions: from article to question as well as from question to article. Specifically, these two types of attention are calculated between sentence-level representations of the article \( H^* \) and the word-level representations of the question \( U^* \).

First, we let \( H^* \) and \( U^* \) go through a \textit{dimension transformation} layer to shrink the dimension of these two representations,

\[
H = w_d H^* + b_d \in \mathbb{R}^{r/4 \times k}, \tag{7}
\]

\[
U = w_d U^* + b_d \in \mathbb{R}^{r/4 \times m}. \tag{8}
\]

\( w_d \) and \( b_d \) are trainable parameters, here we must assume that the hidden size \( r \) is divisible by 4.

Next, both the two directions of co-attention, which will be discussed below, are derived from a shared similarity matrix, \( S \in \mathbb{R}^{k \times m} \), between the transformed sentence representations \( H \) and the transformed question representations \( U \), where \( S_{ij} \) indicates the similarity between \( i \)-th article sentence and \( j \)-th question word. The similarity matrix is computed by

\[
S_{ij} = \phi(H_{i,t}, U_{j,t}) \in \mathbb{R}, \tag{9}
\]
where \( \phi \) is a trainable scalar function that encodes the similarity between its two input vectors, \( \mathbf{H}_i \) is \( i \)-th column vector of \( \mathbf{H} \), and \( \mathbf{U}_j \) is \( j \)-th column vector of \( \mathbf{U} \). We set:

\[
\phi(h, u) = w^\phi_h [h; u; h \circ u], \tag{10}
\]

where \( w^\phi_h \in \mathbb{R}^{3r/4} \) is a trainable weight vector, \( \circ \) is element-wise multiplication, \([;]\) is vector concatenation across row, and implicit multiplication is matrix multiplication.

Then, the similarity matrix is normalized for each column to produce the attention weights \( \mathbf{S}^Q \in \mathbb{R}^{m \times k} \) across the question words for each sentence in the article, and normalized for each row to produce the attention weights \( \mathbf{S}^T \in \mathbb{R}^{m \times k} \) across the article sentences for each word in the question:

\[
\mathbf{S}^Q_{ij} = \text{softmax}(S_{ij}), \forall j; \tag{11}
\]

\[
\mathbf{S}^T_i = \text{softmax}(S_i), \forall i. \tag{12}
\]

**Article-to-question Attention.** Article-to-question attention (A2Q) signifies which question words are most relevant to each article sentence. So for \( j \)-th sentence in the article, we sum over all question representations \( \mathbf{U}_j \in \mathbb{R}^{r/4}, \forall i \) according to their normalized attention weights with that sentence \( \mathbf{S}^Q_{ij} \in \mathbb{R} \). Subsequently, each attended question vector is computed by \( \tilde{\mathbf{U}}_{ij} = \sum_i S^Q_{ij} \mathbf{U}_i \). Hence \( \tilde{\mathbf{U}} \) is a \( r/4 \)-by-\( k \) matrix containing the attended question vectors for the entire article sentences.

**Question-to-article Attention.** Question-to-article (Q2A) attention signifies which article sentences have the closest similarity to each of the question words and are hence critical for locating information that is most relevant for answering that question. We obtain the attended sentence vector on the article words by

\[
\tilde{\mathbf{H}} = \mathbf{H} (\mathbf{S}^T)^\top \mathbf{S}^Q \in \mathbb{R}^{r/4 \times k}, \tag{13}
\]

where \( \mathbf{H} (\mathbf{S}^T)^\top \in \mathbb{R}^{r/4 \times m} \) is the weighted sum of sentence representations for each question word, and \( \mathbf{S}^Q \) is used to map the matrix to length \( k \).

Finally, the sentence representations and the attention vectors are combined together to yield \( \mathbf{G} \), where each column vector can be considered as the question-aware representation of each article sentence. \( \mathbf{G} \) is defined by

\[
\mathbf{G}_{i,j} = \psi(\mathbf{H}_{i,t}, \tilde{\mathbf{U}}_{i,t}, \tilde{\mathbf{H}}_{j,t}) \in \mathbb{R}^{d_G}, \tag{14}
\]

where \( \mathbf{G}_{i,j} \) is the \( t \)-th column vector (corresponding to \( t \)-th article sentence), \( \psi \) is a trainable vector function that fuses its (three) input vectors, and \( d_G \) is the output dimension of the \( \psi \) function. While the \( \psi \) function can be an arbitrary trainable neural network, in our experiments we adopt a simple concatenation as following:

\[
\psi(h, \tilde{u}, \tilde{h}) = [h; \tilde{u}; h \circ \tilde{u}; h \circ \tilde{h}] \in \mathbb{R}^{r \times k}. \tag{15}
\]

**Merging Sentence Representation**

We then go through a gated connection layer to merge the sentence contextual representations and the question-aware representations

\[
g = \sigma(\mathbf{G}) \in \mathbb{R}^{r \times k}, \tag{16}
\]

\[
\mathbf{Z} = g \circ \mathbf{H} + (1 - g) \circ \mathbf{G}. \tag{17}
\]

Where \( \sigma \) is Sigmoid function, and \( \mathbf{Z} \) is the final representation of sentence-level contextual information and the article question co-attention, thus is more likely to capture the article content that is more related to the given question.

**Hierarchical Attention Decoder**

We use another uni-directional LSTM as the decoder to generate distractor.

**Question Initialization.** Unlike standard Seq2Seq generation task like machine translation, in which both source and target sequence are complete sentences, our dataset contain near half of the questions that are not complete sentences (as shown in Figure 1 and Table 1). In order to handle this problem, we follow previous work to use a question-based initializer (Gao et al. 2019).

We use a uni-directional LSTM to encode the question sequence \((q_1, q_2, \ldots, q_m)\) into hidden representations, and denote the hidden state of the final step as \( h_{q_{init}}^w \). Then \( h_{q_{init}}^w \) is used as the initial state the decoder. Moreover, instead of using begin-of-sentence symbol, we use the last token in the question \( q_m \) as the initial input of the decoder.

**Hierarchical Attention.** We employ hierarchical attention (Ling and Rush 2017) to attend the article with different granularity. At each decoding time-step, we parallelly calculate both the sentence-level attention weight \( \beta \) and word-level attention \( \alpha \) by

\[
\beta_i = \mathbf{Z}_i^\top \mathbf{W}_{d_i} h_{i,t}, \quad \alpha_{i,j} = h_{i,j}^w \top \mathbf{W}_{d_j} h_{j,t}, \tag{18}
\]

\[
\gamma_{i,j} = \frac{\alpha_{i,j} \beta_i}{\sum_{i,j} \alpha_{i,j} \beta_i}. \tag{19}
\]

Where \( \mathbf{W}_{d_i} \) and \( \mathbf{W}_{d_j} \) are trainable parameters. The sentence-level attention determines how much each sentence should contribute to the generation at the current time-step, while the word-level attention determines how to distribute the attention over words in each sentence.

Then the context vector \( \mathbf{c}_t \) is derived as a combination of all word-level representations reweighted by the combined attention \( \gamma \):

\[
\mathbf{c}_t = \sum_{i,j} \gamma_{i,j} h_{i,j}^w. \tag{20}
\]

And the attentional vector is calculated as:

\[
\tilde{h}_{i,t}^d = \tanh(W_{\tilde{h}} [h_{i,t}^d, \mathbf{c}_t]). \tag{21}
\]

Finally, the predicted probability distribution over the vocabulary \( V \) at the current step is computed as:

\[
P_V = \text{softmax}(W_V \tilde{h}_{i,t}^d + b_V), \tag{22}
\]

where \( W_{\tilde{h}}, W_V \) and \( b_V \) are learnable parameters.
Semantic Similarity Loss
We assume that in order to form a set of meaningful and educative options, each distractor should be semantically relevant to the article. So we incorporate an additional semantic similarity loss into the original loss function.

In order to calculate the semantic similarity score, we should first obtain the semantic representation vectors of the generated distractor. Previous work has proved that a simple subtraction between LSTM hidden states can represent segment sequence effectively (Wang and Chang 2016). So the distractor representation $e_D$ is computed by:

$$e_D = s_M - e_T,$$

(23)

where $s_M$ denotes the decoder last hidden state, and $e_T$ is the sentence-level encoder’s last hidden state, which is also the representation of the article.

Then we compute the cosine similarity to measure the semantic relevance between distractor and article, which is represented with a dot product and magnitude:

$$\cos(e_D, e_T) = \frac{e_D \cdot e_T}{\|e_D\| \cdot \|e_T\|}.$$

(24)

Our training objective is to maximize the similarity score so that the generated distractor have high semantic relevance with the article. Previous work in text summarization has proved that this cosine similarity loss can improve the semantic relevance of the source text and the generated summary (Ma et al. 2017).

Finally, the model is trained to minimize the total loss:

$$L = -\sum_{d \in V} \log P(d|T, Q; \Theta) - \lambda \cos(e_D, e_T),$$

where $\lambda$ is a hyperparameter to balance two loss functions.

Experimental Settings
Dataset
We conduct extensive experiments on the RACE dataset, which was collected from the English exams for middle and high school Chinese students. It contains 27,933 articles with 97,687 questions, which are designed by human experts for education purpose, making RACE an ideal dataset for training model to generate questions and distractors.

Instead of using original RACE dataset which contains samples that are not suitable for sequence generation, we use the dataset processed by previous work (Gao et al. 2019) under the following two conditions: 1) Filter out the distractors that are semantic irrelevant to the article context. 2) Remove the questions which require to fill in the options at the beginning or in the middle of the questions. Table 1 shows the statistics of our dataset.

Baselines and Evaluation Metrics
We compare our model with the following baselines.

- **Seq2Seq**: The standard sequence-to-sequence structure with global attention mechanism (Luong, Pham, and Manning 2015). The encoder take the whole article sequence as input, and use a single LSTM to encoder this sequence.

<table>
<thead>
<tr>
<th># samples</th>
<th>Train</th>
<th>Valid</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>% incomplete-sentence questions</td>
<td>47.48</td>
<td>46.48</td>
<td>47.57</td>
</tr>
<tr>
<td>Avg. article length</td>
<td>347.21</td>
<td>344.78</td>
<td>347.66</td>
</tr>
<tr>
<td>Avg. question length</td>
<td>9.91</td>
<td>9.97</td>
<td>9.93</td>
</tr>
<tr>
<td>Avg. distractor length</td>
<td>8.50</td>
<td>8.50</td>
<td>8.54</td>
</tr>
</tbody>
</table>

Table 1: The statistics of our dataset.

- **HRED**: Vanilla hierarchical structure as described before. It consists of A hierarchical encoder that is able to encode document-level input text in a way that preserve semantic coherence (Li, Luong, and Jurafsky 2015), and a decoder that utilize hierarchical attention. This structure has been proved effective in text summarization (Ling and Rush 2017) and headline generation (Tan, Wan, and Xiao 2017).

- **HRED+copy (HCP)**: Incorporating pointer-generator-network (See, Liu, and Manning 2017) with HERD to enable the decoder to directly copy words from the source text. Pointer-generator-network made a big improvement in text generation tasks like text summarization, so we consider HCP to be a strong baseline.

- **HRED+static_attn (HSA)** (Gao et al. 2019): A static attention that penalize the correlation between the answer and generated distractors was proposed to modulate the hierarchical attention in HERD. This model achieved state-of-the-art performance previously on this task.

Following the previous work (Gao et al. 2019), We adopt BLEU (Papineni et al. 2002) and ROUGE (Lin 2004) to evaluate the performance of our models.

Implementation Details
Our training set contains 100,116 distinct words and we keep the most frequent 50k tokens as vocabulary. Those tokens outside the vocabulary are replaced by the UNK symbol. The hidden unit size of all LSTMs is set to 600. The word-level encoder and sentence-level encoder are bidirectional LSTMs with their number of layer to be 2 and 1 respectively. The question initialization encoder and the decoder are 2 layers unidirectional LSTMs. We use the GloVe.840B.300d word embeddings and make them trainable. For optimization in training, we use stochastic gradient descent (SGD) as the optimizer and set the gradient norm upper bound to 5.0. We set minibatch size to 10 and the initial learning rate to 1.0 with a decay rate of 0.8. For hyperparameter $\lambda$ of semantic similarity loss, we tested in a large range and set it to 0.0001.

During inference, we adopt beam search and set beam size to 10. We keep the 10 best candidate distractors in decoding likelihood, and utilize a Jaccard distance of 0.5 to select three diverse distractors. Therefore, the Jaccard distance between distractor $D_2$ and $D_1$ is larger than 0.5, and the Jaccard distance between distractor $D_3$ and both $D_1$ and $D_2$ is also greater than 0.5.
Table 2: Automatic evaluation results of different models. * symbol indicates the results are taken from the original paper of (Gao et al. 2019). The best performing result for each metric is highlighted in boldface.

Table 3: Ablation study of our model. "+SSL" means adding the semantic similarity loss, and "+Co-Attn" means adding the co-attention network. "-Merging" means getting rid of the merging layer. Here we only list results of the first distractor.

Results and Analysis

Main Results

The experimental results are shown in Table 2. We also list the results of Seq2Seq and HSA from the previous work (Gao et al. 2019) for reference. Our model achieves the best results across most metrics for all three distractors. As for the first distractor that is most important in our task, our model obtains a new state-of-the-art performance of 7.01 at BLEU-4 metric, which outperforms the existing best result (Gao et al. 2019) by 0.54 points.

As shown in Table 2, there is a large performance gap between Seq2Seq and HRED, which reveals that the hierarchical structure is indeed effective for keeping semantic information of long sequential input. It also can be found that incorporating copy mechanism does not improve performance, we think one reason is that the copy mechanism mainly handle out-of-vocabulary (OOV) words problem, and our dataset is used for education purpose, there is only an OOV words ratio of 0.51% among all distractor’s word occurrences.

Ablation Study

We compare the results of our full model with its ablated variants to analyze the relative contributions of each component. The results are shown in Table 3. It indicates that both our proposed co-attention architecture and the semantic similarity loss improve the performance over the basic HRED model obviously, and combining them together helps our full model achieve state-of-the-art performance.

We also verified the necessity of the gated connection layer. By getting rid of merging layer and using question-aware representation $G$ as our sentence-level representation, the model performance actually decrease. This shows it’s necessary to keep the original sentence-level representation $H$.

Human Evaluation

We also conduct a human evaluation to evaluate the generated distractors of our different models. We design three metrics to evaluate the quality of generated distractors. For all the metrics, we ask the annotator to score the distractors with three gears, the scores are then projected to 0 - 10. We employ three annotators to evaluate the distractors generated by our three most competitive models over the first 100 samples of the test set.

- **Fluency**: This metric evaluates whether the distractor follows regular English grammar and whether the distractor accords with human logic and common sense.

Table 4: Results of human evaluation.
**Coherence:** This metric looks for key phrases in the distractors and measures whether these key phrases are relevant to the article and the question.

**Distracting Ability:** This metric evaluates how likely a distractor candidate will be chosen as distractor in real examinations. This metric is designed to detect whether a distractor tries to mislead the examinees to an irrelevant topic, or in other words, distract by misleading.

The results are presented in Table 4. It is amazing to find that vanilla hierarchical structure model HERD actually yields quite competitive results. Our model get the highest scores in Fluency and Distracting Ability metrics, and a nearly best score in Coherence metric. This shows that our model is able to generate more coherent and educative distractors. HSA gets a low score in Distracting Ability metric, we hypothesis that this is because static attention penalizes the correlation between the generated distractors and the correct answer, so the generated distractors tend to be semantically far away from the correct answer and less relevant to the article.

**Case Study**

In Figure 1, We show an example of distractors generated by our strongest baseline (HSA) and our model. This article is about shyness and the way to overcome it. Distractors generated by HSA contain words like wealth, provides, and seek help, which are not relevant to the content of this article. So if examinees do not understand this article very well, these irrelevant distractors may confuse them more, leading them to wrongly choose these irrelevant distractors. Previous work has proved that distractors generated by HSA are most successful in confusing the annotators (Gao et al. 2019), we hypothesis that part of this confusing ability comes from their misleading property, thus these distractors do not serve a good education purpose. We also present some general cases in Figure 3, the first article is about dining manners. Distractor generated by HSA shares some common words with the article, but it’s semantically irrelevant to the article. And our model (CHN) generates more coherent distractor than HERD. The second article is about different customs of celebrating birthday in different countries. Distractor generated by HSA is too general to be meaningful, and the article didn’t mention or talk about birthday invitations, so the distractor generated by HERD is not a good one. While distractor generated by our model only changed the country name in the original article sentence, examinees need to carefully check the article content to make a judgment. Therefore, it is a coherent and educative distractor.

**Conclusion**

In this paper, we present a Co-Attention Hierarchical Network to generate coherent long distractors for reading comprehension multiple choice questions. A co-attention enhanced hierarchical architecture is exploited to model the complex interactions between the article and question, guiding the decoder to generate more coherent and consistent distractors. Then a semantic similarity loss is incorporated into the original loss to push the generated distractors to be more relevant to the article content. Our model outperforms several strong baselines including the existing best model. The ablation study verifies the robustness of our model, and human evaluation shows our model is able to generate more coherent and educative distractors. For the future work, some interesting directions include exploring more complex co-attention structure and utilizing the information provided by the correct answer.

**Acknowledgments**

We thank Wenjie Zhou for his valuable comments and suggestions. This work is supported by the National Natural Science Foundation of China (61773026) and the Key Project of Natural Science Foundation of China (61936012).

**References**
