Original Article

ViMRC - VLSP 2021: An empirical study of Vietnamese Machine Reading Comprehension with Unsupervised Context Selector and Adversarial Learning

Tran Hoang Vu*, Nguyen Phúc Minh*

Vinbrain, 458 Minh Khai, Hai Ba Trung, Hanoi, Vietnam

Received 19 December 2021
Revised 18 April 2022; Accepted 5 May 2022

Abstract: Machine Reading Comprehension (MRC) is a great NLP task that requires concentration on making the machine read, scan documents, and extract meaning from the text, just like a human reader. One of the MRC system challenges is not only having to understand the context to extract the answer but also being aware of the trust-worthiness of the given question is possible or not. Thought pre-trained language models (PTMs) have shown their performance on many NLP downstream tasks, but it still has a limitation in the fixed-length input. We propose an unsupervised context selector that shortens the given context but still contains the answers within related contexts. In VLSP2021-ViMRC Challenge [1] dataset, we also empirical several training strategies consisting of unanswerable question sample selection and different adversarial training approaches, which slightly boost the performance 2.5% in EM score and 1% in F1 score.

Keywords: Machine reading comprehension, Adversarial learning, Vietnamese.

1. Introduction

Machine Reading Comprehension (MRC) is a task introduced to test the level at which a machine can understand natural languages by asking the machine to answer questions based on a given context. The early MRC systems were designed on a latent hypothesis that all questions can be answered according to a given context, which is not always true for real-world cases. The current MRC task has required that the model have to classify unanswerable and answerable questions to avoid giving plausible answers. Figure 1 shows an unanswerable example from UIT-ViQuAD 2.0 dataset [1]. PTMs such as ELMo [2], GPT [3], or BERT [4] have been proposed and achieved superior results on MRC tasks by capturing contextual representation features. Most of BERT-family architecture [4] usually face to the limitation of fixed input-length. This make a long input must be partitioned into smaller segments of manageable sizes and leads to the loss of salient cross-segment information, the context...
fragmentation problem. Although researchers [5, 6] proposed new architecture to solve this problem, these previous works are focus on English only. Inspired by selecting salient sentences before extract the span answer [7, 8], we introduce an unsupervised context selector that address the long input context.

We experiment with different adversarial training approaches in MRC.

We evaluate and experiment with the proposed methods on the dataset released by VLSP2021-ViMRC Challenge [1].

2. Background

2.1. MRC for Vietnamese

There are limited studies of understanding a text and answering relevant questions for Vietnamese. Most of the Vietnamese dataset that is close to the MRC task comes from AI challenge or Shared-task workshop[1]. It makes lack of benchmark datasets for Vietnamese to develop robust PTMs or MRC models. Then for the first time, UIT-ViQuAD[15] is the first public academic dataset in MRC for Vietnamese. The author benchmarked the dataset in different embeddings, and multilingual PTMs showed the best performance on this dataset.

2.2. Pre-trained Language Models

PTMs on the large unlabeled corpus has shown impressive performance on many downstream NLP tasks, proving that they can learn universal patterns. There have been several applications for using pre-trained language models that can capture contextual word embeddings, such as ELMo [2], GPT [3], or BERT [4] to transfer the knowledge from pre-training to various downstream tasks.

For the very first time that BERT has been introduced, it significantly outperforms previous SOTA models on eleven NLP tasks in GLUE [16]. In terms of monolingual language models pre-trained for Vietnamese, it has shown significant improvements in Named Entity Recognition, Parsing, and Natural Language Inference tasks. PhoBERT pre-training approach is based on RoBERTa [18] which optimizes the BERT pre-training procedure for more robust performance.

In terms of the context sequence $C = \{c_1, c_2, ..., c_N\}$ and question $Q = \{q_1, q_2, ..., q_M\}$, we experiment with different adversarial training approaches in MRC.

We introduce a simple strategy to generate unanswerable examples, called Question-Context Shuffle.
predictions, the model is enabled to output the correct answer span. The answer span $A$ is either a valid span $A = \{a_i, a_2, \ldots, a_j\}$ where $I \leq i \leq j \leq N$ or an empty $A = \{\}$. The input model is the concatenation of C and Q with special tokens [CLS] and [SEP] as [CLS] Q [SEP] C [SEP]. We employ a linear layer with Softmax operation and feed last-layer hidden representation $H \in RLx_d$ as the input to obtain the start/end position probability distributions $ps, pe$ respectively. The training objective of answer span prediction is defined as cross entropy loss for the start and end index position.

$$loss_{start/\text{end}} = \frac{1}{N_k} \sum_k \left[ y_s^k \log(p^k_s) + y_e^k \log(p^k_e) \right]$$

where $N_k$ is the number of examples, $y_s^k$ and $y_e^k$ are respectively ground-truth start and end position of example $k$. We also employ linear layer with Softmax for $h_{CLS} \in H$ and use cross entropy as loss function for classification answerable/unanswerable question.

$$\text{loss}_{\text{CLS}} = -\frac{1}{N_k} \sum_k \sum_u y_u^k \log(p_u^k)$$

where $p_u^k$ is answerable and unanswerable probability distributions. $U$ means the number of classes ($U = 2$ in this work). The overview of our method architecture is illustrated in Figure 2.

2.3. Adversarial Training

Small perturbations to the input images can mislead models to predict wrong labels in the image classification, and the perturbed inputs are called adversarial examples [19]. Then, a simple adversarial training method has been proved can improve the robustness of the model by training on both clean examples and adversarial examples [9]. In NLP tasks, a popular approach to generate perturbations is to perturb word vectors from the embedding layer. In general, adversarial training idea is formulated as follows:

$$y = f_\theta(x)$$
$$y' = f_\theta(x + \text{noise})$$

Figure 2. The overview architecture of our method.

where $\theta$ is our model weight, $x$ is the embedding of the input sequence and noise is simply a tensor that is randomly generated with normal distribution. Motivated by making the MRC model more generalized with diverse inputs, we apply adversarial learning, which is a noise layer for the input. In this work, we utilize R3F [20] that encourages the model to generalize with representation changes during training without hurting performance. The adversarial
training loss $loss_{ADV}$ is calculated by the following:

$$loss_{ADV} = KL(y, y') + KL(y', y')$$

where KL is the KL-Divergence. The final loss function is the summation of mentioned loss with $\lambda_0$, $\lambda_1$ and $\lambda_2$ are learned weight for each task:

$$loss = \lambda_0 loss_{CLS} + \lambda_1 loss_{start/end} + \lambda_2 loss_{ADV}$$

### 2.4. Domain Agnostic (DA)

Adapting models to a new domain without fine-tuning is a challenging problem in deep learning. In this paper, we also experiment with adversarial training called Domain-agnostic. The adversarial training is leveraged for learning domain-invariant representation. Specifically, the MRC model learns to make the discriminator that classifies the joint embedding of context and question into the given T domains. If the discriminator cannot tell the difference between embeddings from different T domains, the MRC model learns domain-invariant feature representation.

The discriminator is trained to minimize the KL divergence between uniform distribution over T classes and discriminator's prediction:

$$loss_{ADV} = -\frac{1}{N} \sum_{t=1}^{T} \sum_{k=1}^{N_k} KL(U(l)||P(l, h_{t}^{k}))$$

where $l$ is domain category, $U(l)$ is the uniform distribution over T classes and $h$ is the hidden representation of both context and question. $N_k$ is number of samples of class $k$ and $N$ is total samples.

### 3. Method

#### 3.1. Unsupervised Context Selector

Due to the input sequence may exceeding the beneficial length of BERT [4] (256 tokens), the losing context results in not only a missing answer context but also harm the model by learning a noisy sample. We introduce an unsupervised context selector that shortens the context but still contains the answer within related contexts. The context selector takes context and question as input then outputs a shorter version of the context while ensuring the answer must be included. We observe that almost all of the questions focus on the entities in the question, so we want to take advantage of these properties to shorten the context.

Since the linguistic style and syntactic of both context and question from the dataset are formal, we decided to use POS-TAGER from underthesea which has been trained on a dataset that has a similar distribution of the former dataset. Given the question, we filter stopwords and use POS-TAGER from underthesea to get POS output. Then we select important phrases based on the following output with tags: 'N','Np','V','Vp' to finalize a phrase set $N$. The context is chunked by sentence segmentation from NLTK [21], each sentence is scored by the occurrence of tokens that are included in the extracted phrases. The sentence $s$ has $t$ syllable-level tokens would be selected if it has a score $score(s) > 2 \epsilon$ as following:

$$score(s) = \max(f(s) + f(s+1); f(s) + f(s-1))$$

$$f(s) = \sum_{t \in N} g(t)$$

where $\epsilon = \sum_{t \in N} score(t) #0 score(t)$, $g(t)$ is the number of co-occurrence of an token $t$ in the given context and question. We also select the previous and next sentence of the selected sentence to make a leading sentence and augment the surrounding context.

#### 3.2. Question-Context Shuffle

According to Table 3, there is an imbalance between answerable and unanswerable questions. This makes the model easily predict plausible answers and mistaken the given context and question. We introduce a simple strategy called Question-Context shuffle to generate unanswerable examples from the training set. This approach aims to augment more unanswerable samples by getting a random irrelevance question for each given context.

We divide the generated unanswerable samples into two types are EXAMPLES$_{hard}$ and EXAMPLES$_{easy}$, for each context, the selected unanswerable questions are from different context but the sample title are categorized into
EXAMPLES\textsubscript{hard} while the unanswerable questions from different title are categorized into EXAMPLES\textsubscript{easy}. The title is the main topic of many contexts which is shown in Table 1. The statistic of the dataset after pre-processing is presented in Table 2 in which the total samples of two class has been balanced.

Table 1. Sample structure of UIT-ViQuAD 2.0 dataset. Each example contains a title corresponds to the the category of the context. Each example has lots of context, each context contains multiple (question,answer,is-possible) triplets.

Table 2. Statistic of classes of training dataset after data augmentation

<table>
<thead>
<tr>
<th></th>
<th>answerable</th>
<th>unanswerable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original data</td>
<td>19240</td>
<td>9217</td>
</tr>
<tr>
<td>EXAMPLES\textsubscript{hard}</td>
<td>0</td>
<td>5975</td>
</tr>
<tr>
<td>TOTAL</td>
<td>19240</td>
<td>21167</td>
</tr>
</tbody>
</table>

4. Experimental Results

4.1. Setup

We employ RDRSegmenter [22] from VnCoreNLP [23] to perform word-level and sentence segmentation on UIT-ViQuAD 2.0 dataset (e.g "Những cá thể xung quanh ghi nhớ tốt tóm tắt bằng cách nào?"). Our experimental models were implemented PyTorch [24] and utilize Huggingface’s Transformers [25] for pretrained language models. In our experiments, almost all experiments used the Shuffle-Context Shuffle strategy to make to model aware of more data.

In practice, we have three-phase of training. In the first phase, we make the model generalize with and warm up with the data by setting the $\lambda_0 = 0.2$, $\lambda_1 = 0.6$, and $\lambda_2 = 0.2$. We observed that the loss\textsubscript{ADV} is converged after the first phase, we decided to set $\lambda_2 = 0$ on every next phase. In the second phase, we aim to make the classification loss which only saves the checkpoint with the lowest loss on the dev set. In the third phase, we focus on the start/end index loss, which considers only the best checkpoint based on CE loss of start/end on dev set. We set the $\lambda_0 = 0.9$ and $\lambda_1 = 0.1$ on the second phase and $\lambda_0 = 0.1$ and $\lambda_1 = 0.9$ on the third phase.

Table 3. Data analysis of UIT-ViQuAD 2.0 dataset. 
# stands for numbers of samples. Public stands for Public testset. Private stands for Private testset. The average length unit is calculated in syllable-level.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Public</th>
<th>Private</th>
</tr>
</thead>
<tbody>
<tr>
<td># articles names</td>
<td>138</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td># passages</td>
<td>4101</td>
<td>557</td>
<td>515</td>
</tr>
<tr>
<td># total ques.</td>
<td>28457</td>
<td>3821</td>
<td>3712</td>
</tr>
<tr>
<td># unanswerable ques.</td>
<td>9217</td>
<td>1168</td>
<td>1116</td>
</tr>
<tr>
<td>Avg. context length</td>
<td>178.98</td>
<td>167.60</td>
<td>175.62</td>
</tr>
<tr>
<td>Avg. ques length</td>
<td>14.64</td>
<td>14.24</td>
<td>14.43</td>
</tr>
</tbody>
</table>

4.2. Dataset

In VLSP2021-ViMRC Challenge [1], the dataset is organized into 3 sets are train/public test/private test has 138/19/19 number of articles respectively. The analysis of the dataset is shown in Table 3. Since there is no dev set, we decided to categorize the articles in the training dataset into two main sets based on answerable and unanswerable questions, making the split dataset balanced in categories and no leaked articles. Then we randomly split these two sets with a ratio of 9/1 before uniting them into a train/dev set based on the mentioned ratio.

4.3. Hyperparameters

In all experiment settings, we use Adam optimizer [26] with a learning rate of 1e-5 without warm-up steps, batch size of 32. In the inference stage, we set the threshold $\delta$ is 0.4 to determine if the question is answerable or not. We set the maximum sequence length for context and questions to be 230 and 50 for each sample. All experiments are launched with a
maximum of 10 epochs and a single A100-40GB GPU device.

4.4. Metric

We evaluated our models by Exact Match (EM) and F1 score for each question-answer pair. The EM score measures the percentage of predictions that match ground-truth answers in character level. The F1 metric aims to care equally about precision and recall of the number of shared words between the prediction and the truth. The precision is the ratio of the number of shared words to the total number of words in the prediction, while recall is the ratio of the number of shared words to the total number of words in the ground truth. The higher the F1 and EM scores are, the closer ground truth and predicted context is.

4.5. Results

4.5.1. Main Result

We use two main PTMs as backbone are: PhoBERT that supports a maximum of 256 tokens, and XLM-Roberta that provides a maximum input length is 512 tokens. We observed that monolingual models (e.g. mBERT [17]) perform better than multilingual models (e.g mBERT [4], XLM-Roberta [27]). Moreover, training monolinguals on a word-level dataset improves performance significantly due to improved quality of words and reduced length of context and question pairs. Using methods Context Selector and Adversarial Training also slightly improve performance. Result experiment is shown on Table 4.

In terms of the private test set, our method has exceeded the baseline +9.76 in F1 score and +7.12 in Exact Match score. However, our method still shows limitations compare to top-3 teams and we would discuss them in Section 4.6. The result of the top-3 teams and our result in the private test is illustrated in Table 5.

Table 4. Results on the UIT-ViQuAD 2.0 public test set. (R3F, DA) refers to adversarial training methods. (CS) refers to Context Selector. ⋆ refers to word-level. w/o QAS refers to without Question-Context shuffle

<table>
<thead>
<tr>
<th>Method</th>
<th>Dev</th>
<th>Public</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EM</td>
<td>F1</td>
</tr>
<tr>
<td>mBERT (baseline)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PhoBERT&lt;sub&gt;base&lt;/sub&gt;</td>
<td>43.56</td>
<td>57.24</td>
</tr>
<tr>
<td>XLM-R w/o QAS</td>
<td>29.35</td>
<td>51.61</td>
</tr>
<tr>
<td>PhoBERT&lt;sub&gt;base&lt;/sub&gt;</td>
<td>45.23</td>
<td>61.18</td>
</tr>
<tr>
<td>PhoBERT&lt;sub&gt;large&lt;/sub&gt;</td>
<td>54.27</td>
<td>69.37</td>
</tr>
<tr>
<td>PhoBERT&lt;sup&gt;*&lt;/sup&gt;&lt;sub&gt;large&lt;/sub&gt;</td>
<td>59.12</td>
<td>74.29</td>
</tr>
<tr>
<td>PhoBERT&lt;sup&gt;*&lt;/sup&gt;&lt;sub&gt;large+DA&lt;/sub&gt;</td>
<td>59.89</td>
<td>75.19</td>
</tr>
<tr>
<td>PhoBERT&lt;sup&gt;*&lt;/sup&gt;&lt;sub&gt;large+R3F&lt;/sub&gt;</td>
<td>59.93</td>
<td>75.35</td>
</tr>
<tr>
<td>PhoBERT&lt;sup&gt;*&lt;/sup&gt;&lt;sub&gt;large+R3F+CS&lt;/sub&gt;</td>
<td><strong>60.05</strong></td>
<td><strong>75.39</strong></td>
</tr>
</tbody>
</table>

4.5.2. Context Selector

We also evaluate our unsupervised Context Selector on the train set, which is shown in Table 6. The probability that the shortened context contains an answer shows competitive results compared to the raw input. In terms of the average context length, the Context Selector helps the model to receive salient sentences only by reducing from 324,32 tokens to 169,2 tokens. The result shows that the Context Selector has

Table 5. Results on UIT-ViQuAD 2.0 private testset. ⋆ refer word-level

<table>
<thead>
<tr>
<th>Team</th>
<th>F1</th>
<th>EM</th>
</tr>
</thead>
<tbody>
<tr>
<td>vs-tus</td>
<td><strong>77.24</strong></td>
<td>66.14</td>
</tr>
<tr>
<td>ebisu_uin</td>
<td>77.22</td>
<td><strong>67.43</strong></td>
</tr>
<tr>
<td>F-NLP</td>
<td>76.46</td>
<td>64.66</td>
</tr>
<tr>
<td>mBERT (baseline)</td>
<td>60.34</td>
<td>49.35</td>
</tr>
<tr>
<td>PhoBERT&lt;sup&gt;*&lt;/sup&gt;&lt;sub&gt;large+R3F+CS&lt;/sub&gt;</td>
<td>70.10</td>
<td>56.47</td>
</tr>
</tbody>
</table>
crucially reduced the context length while retaining the answer in the filtered context.

4.6. Error Analysis

We also examined the errors of our method in the dev dataset that decrease the evaluation score significantly. The major errors are:

Span error: We found that about 40% of errors are span errors. More specifically, the start and end index from the model prediction usually is shifted from the correct ground truth. We hypothesis that this span error may come from the annotator’s bias. It is difficult for the model to be aware of samples with ambiguous answer text. Table 7 shows a few span error examples that we have analyzed in VLSP2021-ViMRC Challenge.

Table 6. Results of Context Selector on T-ViQuAD 2.0 train set. * refers samples that has context length > 256 syllable tokens

<table>
<thead>
<tr>
<th>Input</th>
<th>prob contains ans</th>
<th>avg length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw input</td>
<td>0.6</td>
<td>178.96</td>
</tr>
<tr>
<td>Raw input*</td>
<td>0.6</td>
<td>324.32</td>
</tr>
<tr>
<td>Context Selector</td>
<td>0.92</td>
<td>110.27</td>
</tr>
<tr>
<td>Context Selector*</td>
<td>0.90</td>
<td>169.2</td>
</tr>
</tbody>
</table>

Misclassify answerable/unanswerable: About 35% of errors are failures of misclassifying answerable and unanswerable questions. According to our experiment on dev set, the best threshold δ to classify either the answerable question or not is 0.4. It means that our model does not generalize for the classification of the question when encountering out-of-domain questions.

Context Selector: Since we use the context selector to shorten the context length for each input sequence, the performance of the whole pipeline still depends on the context selector output result. We observe that the context selector dealt with straightforward questions well (e.g., "Tên của vua Nam_Hán là gì?"). However, it has two main drawbacks not exploiting the training data and depending on manual rules. This makes the context selector unable to acknowledge the entities in the dataset domain and has a limited ability to handle multi-hop questions. Moreover, the surrounding context of the answer may not be sufficient or related to the filtered context, which may hurt the model on prediction.

Table 7. Examples of error analysis in VLSP-2021 MRC. Label refers to Growth-truth of the question. Pred refers to predictions of the model with given question

<table>
<thead>
<tr>
<th>Question:</th>
<th>Labels:</th>
<th>Pred:</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Lịch sử của Ba Tư được ghi chép vào năm nào?&quot; (In what year is the history of Persia recorded?)</td>
<td>&quot;không năm&quot; TCN (cira 3200 BC)</td>
<td>&quot;năm 3200 TCN&quot; (3200 BC)</td>
</tr>
<tr>
<td>&quot;Hai năm nay vẫn đang gây ra nhiều tranh cãi&quot; (Currently, a controversy about which issue?)</td>
<td>&quot;nguồn gốc của tên gọi của thực thể - Iran và Persia&quot; (origin of entity names - Iran and Persia)</td>
<td>&quot;nguồn gốc của tên gọi của thực thể - Iran và Persia (Ba Tư)&quot; (origin of entity names - Iran and Persia (Persian))</td>
</tr>
<tr>
<td>&quot;Mẫu thuan giữa Iran và Mỹ ngày càng leo thang ở vấn đề nào?&quot; (On what issue is the conflict between Iran and the US increasingly escalating?)</td>
<td>&quot;chương trình hạt nhân của Iran&quot; (Iran’s nuclear program)</td>
<td>&quot;Vấn đề chương trình hạt nhân của Iran&quot; (Iran’s nuclear program problem)</td>
</tr>
</tbody>
</table>

5. Conclusion

We introduce applied Context Selector to overcome the large context problem, which is a major limitation of PTMs. We introduce Question-Passage shuffle to solve imbalanced data by generating unanswerable examples. In addition, we investigated the effect of some adversarial training methods on the VLSP2021-ViMRC Challenge dataset. We also show error analysis which helps future studies in MRC or interested research utilize our method. Our experiments demonstrate that adversarial training methods improve the MRC model over the pre-trained model 1%.

References


