VNU Journal of Science:
Computer Science and Communication Engineering

Journal homepage: http://www.jcsce.vnu.edu.vn/index.php/jcsce

Original Article

ViMRC - VLSP 2021: Improving Retrospective Reader for Vietnamese Machine Reading Comprehension

Le Hai Nam, Nguyen Sy Duc, Chu Quoc Quan*, Ngo Van Vi

VCCorp Corporation, 1 Nguyen Huy Tuong, Thanh Xuan Trung, Thanh Xuan, Hanoi, Vietnam

Received 8 February 2022
Revised 30 March 2022; Accepted 5 May 2022

Abstract: In recent years, there are multiple systems (e.g search engines and dialogue systems) that require machines to be able to read and understand human text to perform several tasks in an application. Machine Reading Comprehension (MRC) has posed a challenge to the Natural Language Processing (NLP) community in teaching machines to understand the meaning of human text in order to answer questions provided. Specifically, in this challenge, the dataset contains questions that can be unanswerable, otherwise the answers can be extracted from the given passages. To deal with this challenge, our work is mainly based on a recent approach, known as Retrospective Reader, to confronting unanswerable questions. Additionally, we focus on enhancing the ability of answer extraction by applying properly attention mechanism and improving the representation ability through semantic information. Besides, we also present an ensemble way to acquire significant improvements in results provided by single models. Our method achieves 1st place on the Vietnamese MRC shared task at the 8th International Workshop on Vietnamese Language and Speech Processing (VLSP) with F1-score of 0.77241 and exact match (EM) of 0.66137 on the private test phase. For research purposes, our source code is available at https://github.com/NamCyan/MRC_VLSP2021.

Keywords: Machine Reading Comprehension, Vietnamese.

1. Introduction

The concept of Machine Reading Comprehension (MRC) comes from the goal of teaching machines to perform as humans in understanding of text. Tasks in this field mainly relate to the form of question answering, in which the system is required to give correct answers to questions related to given passages. MRC tasks are commonly divided into four types: cloze style, multiple-choice, span prediction and free-form answer [1]. Span prediction type has received extensive attention in the research community and this is also the challenge that we focus on in this work. In span prediction, the answer appears in the passage and

* Corresponding author.
E-mail address: quanchuquoc@tech.admicro.vn

https://doi.org/10.25073/2588-1086/vnucsce.346

the purpose of systems is to find start and end position of this answer. Due to the powerful support of Pre-trained Language Models (PrLMs) [2, 3], it is no longer challenging for machines to reach human-level performance on resource-rich languages with high quality built MRC datasets [4, 5]. However, it has not been widely analyzed in resource-poor languages like Vietnamese. Recently, two Vietnamese MRC span extraction datasets have been published are UIT-ViQuAD 1.0 [6] and UIT-ViNewsQA [7]. However, these two datasets are not diverse and contain only answerable questions, thus it is impractical in real-world applications.

Thanks to VLSP 2021 MRC Shared Task organizer [8], for the first time, a challenging Vietnamese MRC dataset was released in the campaign for competition. This dataset requires the models not only to extract the answers from the passages, but also to be able to distinguish those unanswerable questions to avoid giving plausible answers.

To overcome these challenges, our work inherits from the work of Retrospective Reader (Retro-Reader) [9], as this is a highly generalized approach that can teach a machine to mimic the way how humans deal with complex reading comprehension. Therefore, this approach can be applied to many different languages and is suitable for Vietnamese. However, the methods used in Retro-Reader are quite simple and not applied properly. Our major contributions are as follows:

• We introduce a more appropriate way to apply extra attention mechanism and utilize layer aggregation techniques to improve Retro-Reader and apply to Vietnamese MRC.

• We make a combination of our methods to achieve outstanding results.

• Experiments show that our methods can yield substantial improvements to Retro-Reader and attain the highest result in VLSP2021-MRC Shared Task.

In the rest of the paper, the related work is briefly summarized in section 2. Section 3 and 4 present our methods and experiments respectively. Finally, the conclusion is drawn in section 5.

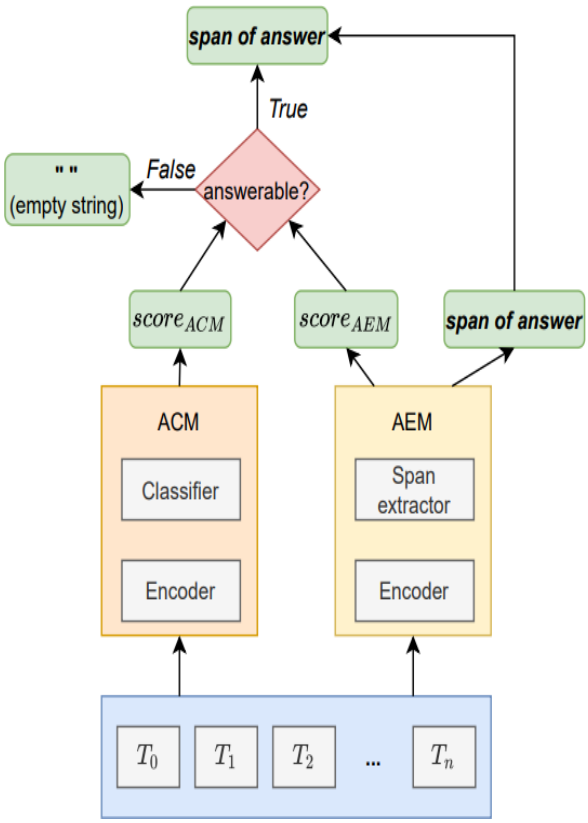

Figure 1. Our approach overview.

2. Related Work

Machine reading comprehension has received widespread attention from researchers in the NLP community. With the emergence of datasets [5] including negative samples (or unanswerable questions), more effort is required to overcome this challenge. Due to remarkable representation capacity, many PrLMs [2, 3] have provided strong performance to this task. Based on this, multiple methods have used PrLMs as the backbone of their strategy in solving MRC. Zhang [10] enriched syntactic relationship for MRC system using information of dependency parsing tree, but this is not appropriate for low-resource languages because of error propagation among tasks. Zhang [9] adopted PrLMs as Encoder and focused on the decoder-side by designing a verifier to handle unanswerable questions. However, these methods have not been investigated on low resource languages

such as Vietnamese. Recently, Nguyen [11] proposed an MRC system for Vietnamese, which is composed of sentence retriever and span extractor components, even so this method has not been exploited in unanswerable cases yet.

To our best knowledge, no work has been done to address the case of unanswerable questions for Vietnamese MRC. In this work, to deal with unanswerable samples, we introduce various techniques to ameliorate the work motivated by Zhang [9] and create a robust system for Vietnamese MRC.

3. Methodology

3.1. General Idea

Problem formalization: Given a passage P and a question Q. The goal is to acquire a predictor R that can provide an answer A with P and Q as inputs. Specifically, in span prediction task, answer A is a span of text in the passage P if the question is answerable and an empty string otherwise.

To handle unanswerable questions, follow the work in Retro-Reader, our approach also uses two main modules:

• A classifier module is used to determine whether a question is answerable when giving a passage, called Answerability Classification Module (ACM).

• Answer Extraction Module (AEM) aims to extract the answer from the passage.

These two modules are similar to sketchy reading module and intensive reading module respectively in Retro-Reader, however, we will introduce some different improvements to the latter in our approach. ACM and AEM modules are trained separately and then process in parallel during the prediction phase. In both modules, we apply BERT-based PrLMs as Encoder to acquire contextual representations of input tokens. The answer prediction is given only by AEM module, while prediction for answerability is provided by both modules depends on their output scores. These scores will be revealed in the two following subsections and the way of combining scores is also presented in Section 3.4. The overview of our approach can be witnessed in Figure 1.

3.2. Answerability Classification Module

Encoder: The question Q and passage P are tokenized into lists of tokens. Then, these tokens are concatenated as the input to the PrLMs (special <s> and </s> tokens are respectively used to determine the start of the input and to separate tokens between Q and P). The output of the Encoder is contextual representation vectors $H = \{h_0, h_1, ..., h_n\}$. Classifier: The Classifier is a fully connected layer. The first token (<s>) representation $h0 \in H$ is pooled then passed as input to the classifier to obtain logits yˆcls including answerable ($logit_{ans}$) and unanswerable ($logit_{na}$) elements. Cross entropy is used as the module's training objective.

The score used to determine answerability is calculated as follow:

$$Score_{ACM} = logit_{na} - logit_{ans} \quad (1)$$

3.3. Answer Extraction Module

It is necessary to have a strong answer extractor to achieve high overall performance. Therefore, we focus on some techniques to improve AEM module. In this section, we illustrate several methods used in our approach to ameliorate AEM performance.

Attention Mechanism: Zhang [9] also investigated two alternative attention mechanisms as an additional layer for question-aware, which are Matching attention and Cross attention. In order to do that, they splitted the contextual representation H into representations of question (H^Q) and passage (H^P).

• Matching attention (MA): HQ is used as the attention to H:

$$M = SoftMax(H(WH^Q + b)^T)$$
$$H' = MH^Q, \quad (2)$$

where W and b are learnable parameters.

• Cross attention (CA): In CA, H, H^Q, H+Q act as Q, K, V respectively like in multi-head self attention mechanism.

The vector H' is then passed through a fully connected layer to obtain logits yˆans composed of answer start and end logits. The original submission of Retro-Reader does not apply any

extra attention mechanisms, while they introduce a joint loss function (span extraction loss and answerability classification loss) to train the module. However, our experiments on Vietnamese dataset (Table 4) show that this can reduce the ability of AEM module to extract answers and thereby decrease overall performance. Moreover, the question is the key to be aware in both attention mechanisms suggested by Zhang [9]. Previously, Cui [12] investigated attention mechanism in BERT on the MRC task and showed that attention from the passage has the most impact on performance, while the result when reducing the attention from the question only dropped slightly. From those perspectives, we only use span extraction loss for AEM and also experiment with HP as the key for attention mechanism-based improvement methods.

Layer Aggregation: Semantic information is clearly a crucial resource for helping models gain deeper understanding of the text to solve MRC tasks. Jawahar [13] previously showed that higher layers in BERT bring rich semantic features of linguistic information and intuitively different layers can have their unique capabilities to encode semantic information. Therefore, we aim to combine multiple top layers of the model to enrich semantic features for the input representations. Inspired by Karimi [14] - a method for Aspect based sentiment analysis task, we apply their P-SUM and H-SUM layer aggregation methods as an extra layer to enhance representation ability. The architectures of P-SUM and H-SUM are depicted in Figure 2. There is a slight difference between our usage of these aggregation methods and the original way. For each layer used, Karimi [14] computed the loss function from the output of that layer and then averaged all the loss values to obtain the final loss. Meanwhile in our work, we first average the outputs of the layers used, then calculate the final loss. Since it is more convenient for our implementation and improvement while preserving the semantic information enhancement effect of the original method.

In AEM, the model is oriented to predict the first token (<s>) as the answer for unanswerable examples. Thus, AEM can also support to determine whether a question is answerable by providing a score according to answer start (logits) and end (logite) logits. This score is calculated as follows:

$$score_a = \max(logit_{s,k} + logit_{e,l}),\ 0 < k \leq l \leq n,$$
$$score_{null} = logit_{s,0} + logit_{e,0},$$
$$score_{AEM} = score_{null} - score_a, \quad (3)$$

where n is the number of tokens of the input.

Table 1. Information about UIT-ViQuAD 2.0

	Train	**Public test**	**Private test**	**All**
Number of articles	138	19	19	176
Number of passages	4,101	557	557	5,173
Number of total questions	28,457	3,821	3,712	35,990
Number of unanswerable questions	9,217	1,168	1,116	11,501

Table 2. Statistics of splitted training and developmentsets

	Train	Dev
Number of articles	125	13
Number of total questions	26,259	2,198
Number of unanswerable questions	8,505	712

3.4. Ensemble Method

Since our system contains two separate modules playing different roles, we need to combine them into a joint model to avoid error propagation if working independently. Besides, each improvement technique corresponds to a single model which comprehends specific valuable information, so combining them can also accomplish significant improvement in performance. In Retro-Reader, this is referred to a module called Rear Verification, however we find that in general it is an ensemble way to combine two modules. Additionally, the original Retro-Reader only use single model per each module, thus we will

present the extensive combination way in the application of multiple models per module.

For simplicity, we first assume that there is only one model per module like Retro-Reader. The scores provided by ACM (Eq. 1) and AEM (Eq. 3) modules are weighted sum to attain verification score and threshold based answerable verification (TAV) [2,3,9] is employed to specify answerability.

$score_{ver} = \beta_0\ score_{ACM} + \beta_1\ score_{AEM}$ (4)

Following the source code provided by Zhang [9], we adopt a simple grid search method to find the weight vector β = {β0, β1}. TAV requires a threshold δ to decide whether a question is answerable and this threshold δ is specified based on the development set. The model predicts the answer span given by AEM if the scorever is above the threshold δ and null string otherwise. In the case of combining multiple models, each module contains a number of models and each model provides a score si. Final verification score (scorever) used for answerability prediction is also the weighted sum of all scores in s, with $s = \{si|i = \overline{1,m}\}$ and weight vector $\beta = \{\beta_j|\ j = \overline{1,m}\}$ where m is the number of total models and Σβj = 1. In addition, answer prediction scores of models in AEM are also combined to get the highest score to give the final prediction if the question is answerable. Similar to verification score, a weight vector α = {α0, α1, ..., αl} (where l is the numbers of models in AEM) and grid search are used in this combination.

4. Experiments and Results

4.1. Dataset

The VLSP organizers released a dataset called UIT-ViQuAD 2.0 into three phases during the competition. Details for this dataset given by the organizers are illustrated in Table 1. Since development set is required to choose the best model and to specify heuristic parameter δ, we split original training set into a new training set and a development set. Table 2 reveals information about our training and development sets.

In addition, we find that some answers are wrong annotated in the published training set. There are 153 examples where the answer position is annotated 1 character off from the exact position, so we correct these mistakes in our data processing step. We also analyze some characteristics of the sentence length according to words in the dataset to select the appropriate parameters for experiment. This information is revealed in Figure 3.

Table 3. Examples of word segmentation using three powerful libraries for Vietnamese:

original text	nàng thùy mị.
vncorenlp	nàng thuỳ_mị .
pyvi	nàng thùy_mị .
underthesea	nàng thùy_mị .

4.2. Experiment Setup

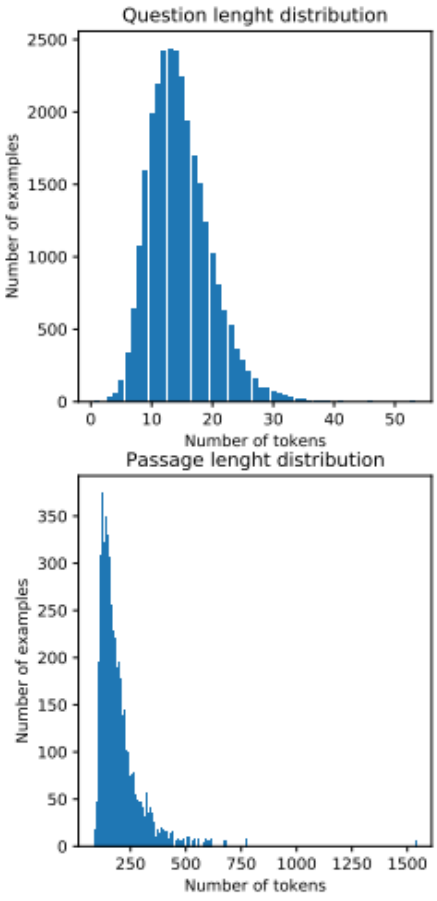

Figure 3. Question and Passage length distribution.

In our experiment, we use two powerful pretrained models that support Vietnamese are PhoBERT [15] and XLM-RoBERTa (XLM-R) [16]. For improvement methods, we follow public source codes provided in [9, 14] for our implementation. In ACM, suggested by Nguyen [15] we segment words using VnCoreNLP library for PhoBERT model. However, word segmentation can change the original text and mess up the position of the characters (see Table 3), leading to negative effect in extracting answers, we only use whitespace tokenization for PrLMs in AEM.

For the fine-tuning in our tasks, we set the initial learning rate in {2e −5, 3e −5} and batch size in {16, 32}. Besides, we select max input length in {256, 400, 512}, max question length in {32, 64} and max answer length to 200. For layer aggregation methods, the number of top layers to combine is set to 4 in both experiments of P-SUM and H-SUM.

Table 4. Individual results of ACM and AEM modules using simple single models. JL and SL stand for joint loss and single loss, respectively, to train AEM module. "2-X" means PrLM X is used as the Encoder in both ACM and AEM. HSUM is the method presented in Section 3.3

ACM	
Models	**Accuracy (%)**
PhoBERT$_{Large}$	82.393
PhoBERT$_{Large_HSUM}$	82.985
XLM-R$_{Large}$	**85.805**

AEM		
	F1 (%)	**EM (%)**
XLM-R$_{Large_JL}$	75.740	62.102
XLM-R$_{Large_SL}$	**76.212**	**62.690**
PhoBERT$_{Large_SL}$	67.668	53.822

ACM + AEM (joint)		
	F1 (%)	**EM (%)**
2-XLM-R$_{Large_JL}$	76.478	62.920
2-XLM-R$_{Large_SL}$	**76.925**	**63.103**

4.3. Results

F1-score and Exact match (EM) are used to evaluate Vietnamese MRC task in the competition. Details are as follows:

Exact match (EM): For each question-answer pair, EM = 1 if predicted answer exactly match the gold standard answer, otherwise EM = 0.

F1-score: The F1-score is calculated based on the number of matched tokens between the predicted and gold standard answers.

$$\text{F1-score} = \frac{2 * Precision * Recall}{Precision + Recall}$$

With

$$Precision = \frac{num\_tokens_{MT}}{num\_tokens_{PA}},$$

$$Recall = \frac{num\_tokens_{MT}}{num\_tokens_{GA}}$$

where num_tokens_{MT} is the number of matched tokens. num_tokens_{PA} and num_tokens_{GA} are the total number of tokens in the predicted answer and the total number of tokens in the gold standard answer respectively.

We first train some simple models to investigate initial results using PhoBERT and XLM-R for ACM and AEM. Table 4 illustrates the discrete results provided by ACM and AEM using PhoBERT and XLM-R and the results of using joint model to show the impact of single loss and joint loss on overall performance of the system. Note that the results presented in Table 4 of ACM and AEM are evaluated separately on the tasks that they undertake (answerability classification and answer extraction respectively). As we can see, XLM-R outperforms PhoBERT in both answer extraction and answerability classification tasks, although PhoBERT is one of the most powerful PrLMs for Vietnamese and has been used widely in many works. This can be explained as the maximum input length for PhoBERT is only 256 tokens, which is quite short (compared with length features shown in Figure 3), hence truncation is applied quite often and sometimes the answer context might be eliminated. Meanwhile, the maximum input length of XLM-R can be up to 512 tokens and can handle most cases. Moreover, as mentioned in Section 4.2, word segmentation is not applied in AEM to avoid altering original text (see Table 3), thus the capability of PhoBERT is restricted, as this model was trained on a huge corpus processed

with word segmentation. However, XLM-R is not affected by this, as it only uses whitespace tokenization in the preprocessing step. Therefore, PhoBERT becomes less efficient for these tasks compared to XLM-R. From those perspectives, we only adopt XLM-R as the backbone in AEM to apply improvement techniques in the following experiments. Furthermore, we also observe that models trained with joint loss function [9] perform worse than which using single loss function in AEM and lead to low overall performance (sub-tables 2 and 3 of Table 4). Thereby, single loss function is adopted to train AEM's models in our work.

Table 5. F1 and EM scores of a range of MRC methods on UIT-ViQuAD 2.0. MA and CA stand for Matching Attention and Cross Attention respectively. QuA is question-aware and PaA is passage-aware. We also consider the results of top 3 participants (ours, ebisu_uit and F-NLP) on the private test phase of the competition in our comparison. The results provided by the organizers [8] are marked with "†"

MRC methods	Development set		Public test set		Private test set	
	F1 (%)	**EM** (%)	**F1** (%)	**EM** (%)	**F1** (%)	**EM** (%)
BASELINE †	-	-	63.031	53.546	60.338	49.353
XLM-R	76.212	62.690	-	-	-	-
XLM-R$_{MA_QuA}$	77.609	63.830	-	-	-	-
XLM-R$_{CA_QuA}$	77.805	64.467	-	-	-	-
Our improvement methods						
XLM-R$_{MA_PaA}$	78.576	64.330	-	-	-	-
XLM-R$_{CA_PaA}$	78.807	65.195	-	-	-	-
XLM-R$_{PSUM}$	77.880	63.920	-	-	-	-
XLM-R$_{HSUM}$	78.027	63.694	-	-	-	-
Other teams						
ebisu_uit †	-	-	**82.622**	**73.689**	77.222	**67.430**
F-NLP †	-	-	80.578	70.622	76.456	64.655
Our ensemble models						
Ensemble$_5$	81.162	67.561	81.013	71.316	77.132	65.867
Ensemble$_8$	**81.536**	**68.380**	-	-	**77.241**	66.137

Since only Large-type PrLMs are used, we omit large notation for brevity. The results of our methods are presented in Table 5. It is straightforward to see that our methods on single model significantly improve the performance of AEM module compared to the baselines by 1 to 2% on the F1 score. Furthermore, ensemble models (joint models of ACM and AEM and each module contains multiple models) even achieve a remarkable increase of around 3 to 4 % on development set. Two ensemble models are used in our submission during the test phases of the competition. Ensemblen is the notation for the combination of n single models. In our cases, we experiment with Ensemble5 and Ensemble8, details are described as follows:

- Ensemble5:
  - **ACM:** XLM-R
  - **AEM:** XLM-R$_{HSUM}$, XLM-R$_{MA_PaA}$, XLM-R$_{CA_PaA}$ XLM-R$_{CA_QuA}$
- Ensemble8:
  - **ACM:** XLM-R, PhoBERT$_{HSUM}$
  - **AEM:** XLM-R$_{HSUM}$, XLM-R$_{PSUM}$, XLM-R$_{MA_PaA}$,XLM-R$_{MA_QuA}$, XLM-R$_{CA_PaA}$, XLM-R$_{CA_QuA}$

The difference between Ensemble5 and Ensemble8 is that the latter contains one additional model in ACM and two additional models in AEM, since we believe the more number of models is used the better the performance is. Due to time limit, we could not submit the result of Ensemble8 in time on the public test phase. For the final submission (predictions of Ensemble8) on private test, we use a simple trick to make the result more robust, called ensemble of ensemble (EOE). After searching, top 3 results according to F1-score are acquired to give the final answers.

The final answer of a question is considered based on the majority predictions and is the answer of the best result if the predictions are all different. Our final result using EOE achieves F1-score of 0.77241 on the private test and rank 1st place in the competition.

5. Conclusion

In this paper, we have presented several methods to improve Retrospective Reader system and apply to Vietnamese MRC task. Our works focused on improving the answer extraction module by applying matching and cross attention suggested by the Retrospective Reader in a more consistent way. In addition, we also used the information of many layers from PrLM to enhance the semantic information for the representation ability. The experimental results showed that our methods are effective and achieves superior performance on UIT-ViQuAD 2.0 dataset. In the future, we plan to adopt more PrLMs such as BARTpho, Electra and Albert to our method. The dataset also gives the information of plausible answers for the unanswerable questions, this information has not been exploited in this work. Therefore, we also want to utilize this valuable information to improve the performance of ACM in our future works.

References

[1] C. Zeng, S. Li, Q. Li, J. Hu, J. Hu, A Survey on Machine Reading Comprehension—Tasks, Evaluation Metrics and Benchmark Datasets, Applied Sciences, Vol. 10, No. 21, 2020, pp. 7640.

[2] J. Devlin, M. Chang, K. Lee, K. Toutanova, Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding, In: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of Tte North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Naacl-Hlt 2019, 2019, pp. 4171–4186.

[3] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, Albert: A Lite Bert for Self-Supervised Learning of Language Representations, In: 8th International Conference on Learning Representations, Iclr 2020, 2020.

[4] P. Rajpurkar, J. Zhang, K. Lopyrev, P. Liang, Squad: 100, 000+ Questions for Machine Comprehension of Text, In: J. Su, X. Carreras, K. Duh (Eds.), Proceedings of The 2016 Conference On Empirical Methods In Natural Language Processing, Emnlp 2016, 2016, pp. 2383–2392.

[5] P. Rajpurkar, R. Jia, P. Liang, Know What You Don't Know: Unanswerable Questions for Squad, In: I. Gurevych, Y. Miyao (Eds.), Proceedings of the 56th Annual Meeting of The Association for Computational Linguistics, Acl 2018 2018, pp. 784–789.

[6] K. V. Nguyen, V. Nguyen, A. H. Nguyen, N. L. Nguyen, A Vietnamese Dataset for Evaluating Machine Reading Comprehension, In: D. Scott, N. Bel, C. Zong (Eds.), Proceedings of The 28th International Conference on Computational Linguistics, Coling 2020, 2020, pp. 2595–2605.

[7] K. V. Nguyen, T. V. Huynh, D. V. Nguyen, G. T. Nguyen, N. L. T. Nguyen, New Vietnamese Corpus for Machine Reading Comprehension of Health News, Arxiv:2006.11138.

[8] N. V. Kiet, T. Q. Son, N. T. Luan, H. V. Tin, L. T. Son, N. L. T. Ngan, Vlsp 2021 - Vimrc Challenge: Vietnamese Machine Reading Comprehension, VLSP 2021.

[9] Z. Zhang, J. Yang, H. Zhao, Retrospective Reader for Machine Reading Comprehension, In: Thirty-Fifth Aaai Conference on Artificial Intelligence, Aaai 2021, 2021, pp. 14506–14514.

[10] Z. Zhang, Y. Wu, J. Zhou, S. Duan, H. Zhao, R. Wang, Sg-Net: Syntax-Guided Machine Reading Comprehension, In: The Thirty-Fourth Aaai

Conference on Artificial Intelligence, Aaai 2020, 2020, pp. 9636–9643.

[11] K. V. Nguyen, N. D. Nguyen, P. N. Do, G. Nguyen, N. L. Nguyen, Vireader: A Wikipedia-Based Vietnamese Reading Comprehension System Using Transfer Learning, J. Intell. Fuzzy Syst., Vol. 41, No. 1, 2021, pp, 1993–2011.

[12] Y. Cui, W. N. Zhang, W. Che, T. Liu, Z. Chen, Understanding Attention in Machine Reading Comprehension, Arxiv:2108.11574.

[13] G. Jawahar, B. Sagot, D. Seddah, What Does Bert Learn About The Structure of Language?, Proceedings Of The 57th Conference Of The Association For Computational Linguistics, Acl 2019, 2019, pp. 3651–3657.

[14] Karimi, L. Rossi, A. Prati, Improving Bert Performance for Aspect-Based Sentiment Analysis, Arxiv Preprint Arxiv:2010.11731.

[15] D. Q. Nguyen, A. T. Nguyen, Phobert: Pre-Trained Language Models for Vietnamese, In: T. Cohn, Y. He, Y. Liu (Eds.), Findings of the Association for Computational Linguistics: Emnlp 2020, Vol. Emnlp 2020 Of Findings of Acl, 2020, pp. 1037–1042.

[16] Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised Cross-Lingual Representation Learning at Scale, In: D. Jurafsky, J. Chai, N. Schluter, J. R. Tetreault (Eds.), The 58th Annual Meeting of The Association for Computational Linguistics, 2020, pp. 8440–8451.