Toward Human-Friendly ASR Systems: Recovering Capitalization and Punctuation for Vietnamese Text

Thi Thu HIEN NGUYEN^{†a)}, *Member*, Thai BINH NGUYEN^{††b)}, Ngoc PHUONG PHAM^{††c)},
Quoc TRUONG DO^{††d)}, Tu LUC LE^{†††e)}, *Nonmembers*, and Chi MAI LUONG^{††††f)}, *Member*

SUMMARY Speech recognition is a technique that recognizes words and sentences in audio form and converts them into text sentences. Currently, with the advancement of deep learning technologies, speech recognition has achieved very satisfactory results close to human abilities. However, there are still limitations in identification results such as lack of punctuation, capitalization, and standardized numerical data. Vietnamese also contains local words, homonyms, etc, which make it difficult to read and understand the identification results for users as well as to perform the next tasks in Natural Language Processing (NLP). In this paper, we propose to combine the transformer decoder with conditional random field (CRF) to restore punctuation and capitalization for the Vietnamese automatic speech recognition (ASR) output. By chunking input sentences and merging output sequences, it is possible to handle longer strings with greater accuracy. Experiments show that the method proposed in the Vietnamese post-speech recognition dataset delivers the best results.

key words: capitalization, punctuation, automatic speech recognition

1. Introduction

Research in Vietnamese speech synthesis and speech recognition was first introduced more than twenty years ago. Since 2018, there has been a competition in Vietnamese Language and Speech Processing (VLSP) for researchers on Vietnamese speech processing. The results from the competition show that the accuracy of systems has improved.

In standard automatic speech recognition (ASR) systems, outputs typically consist of lower-case word sequences where the important linguistic and punctuation information are not available. In particular, there is a difference in the display of numbers, year, currency, proper nouns denoting foreign people, places, organization names. Therefore, improving the output of ASR is necessary. It is simpler for people to read and comprehend a text with correct punctu-

ation and capitalization.

In fact, where ASR results are used in natural language processing models, punctuation and capitalization become important information to enhance further the accuracy or identification of machine translation (MT), name entity recognition (NER), etc. To the best of our knowledge, this task for Automated Speech Recognition in Vietnamese has not been explored before. An example enriching ASR with recovering punctuation and capitalization is shown below:

gần đây việt nam đang tích cực triển khai giáo dục stem trong chương trình giáo dục phổ thông tuy nhiên thực tế cho thấy ở những vùng kinh tế xã hội khó khăn vùng núi vùng có nhiều đồng bào dân tộc thiểu số lại có thêm các vấn đề cần bàn thảo như chương trình đào tạo giáo viên là người dân tộc thiểu số như thế nào nội dung giáo dục stem có nên thống nhất giữa vùng khó khăn và vùng đồng bằng hay không

Gần đây, Việt Nam đang tích cực triển khai Giáo dục STEM trong Chương trình Giáo dục Phổ thông. Tuy nhiên, thực tế cho thấy ở những vùng kinh tế khó khăn, vùng núi, vùng có nhiều đồng bào dân tộc thiểu số lại có thêm các vấn đề cần bàn thảo như: chương trình đào tạo giáo viên là người dân tộc thiểu số như thế nào? nội dung Giáo dục STEM có nên thống nhất giữa vùng khó khăn và vùng đồng bằng hay không?

For example, in addition to the regulation of proper nouns (Việt Nam), capitalizes at the starting of a paragraph, and after punctuation (Gần đây, Tuy nhiên), there are regulations in Vietnam about the name of agencies, organizations, programs, and projects (Chương trình Giáo dục Phổ thông, Giáo dục STEM).

Studies often follow several directions, such as restoring punctuation, capitalizing separately, or combining parallel processing of punctuation and capitalization tasks. Regarding the only task of restoring capitalization, Lita et al. [1] insisted that capitalization consists of restoring the first word of a sentence and proper noun (names of persons, organizations, locations, etc.), and also called tRuEcasIng. The proposed method focused on n-gram LMs, which was derived from a corpus with case information. Chelb et al. [2] provided a general technique for modifying maximum entropy probability models. Another approach also was introduced as Conditional Random Fields (CRFs) for capital-

Manuscript received November 11, 2020.

Manuscript revised March 16, 2021.

Manuscript publicized May 25, 2021.

[†]The author is with Thai Nguyen University of Education, Thai Nguyen, Viet Nam.

^{††}The authors are with Vietnam Artificial Intelligence System, Ha Noi, Viet Nam.

^{†††}The author is with Office of Hanoi People's Committee, Ha Noi, Viet Nam.

^{††††}The author is with University of Science and Technology of Hanoi, Ha Noi, Viet Nam.

a) E-mail: hienntt.math@tue.edu.vn

b) E-mail: binhnguyen@vais.vn

c) E-mail: phuongpn@tue.edu.vn

d) E-mail: truongdo@vais.vn

e) E-mail: luc.letu@gmail.com

f) E-mail: lcmai@ioit.ac.vn

DOI: 10.1587/transinf.2020BDP0005

ization [3]. In this model, the authors suggested a probabilistic bilingual capitalization model for the capitalization of machine translation outputs and demonstrated better capitalization than the LM model. In the last few years, another proposed method named Recurrent Neural Networks (RNNs) at character level [4], which showed a competitive comparison to CRFs. This method solved effectively the out-of-vocabulary (OOV) words, however, it illustrated the difficulty when handling the long sentences.

Recently, many studies of automatic punctuation restore in transcripts of ASR have been introduced. Punctuation is essential not only for subsequent application performance but also for text readability in ASR transcripts. There are also many approaches to the Punctuation task such as Maximum Entropy [5], Long Short Term Memory [6], Dynamic Conditional Random Fields [7], [8], DNN-CRF (Deep Neural Network - Conditional Random Fields) [9], Bidirectional Recurrent Neural Network [10], Recurrent Neural Networks [11], [12]. Especially, Tundik et al. [13] has proved the outstanding performance of RNN compared to the MaxEnt method [13]. Recently, Tündik et al. [14] also has proposed improving the word sequence-based system using a Convolutional Neural Network (CNN) and described the substantial improvement in the accuracy of punctuations.

It is difficult to perform individually the tasks for optimizing document recovery, therefore, the approaches now try to integrate the handling of punctuation and capitalization recovery in ASR models such as the statistical model [15], n-gram language model [16], [17], Maximum Entropy [18]. The deep learning models have proved a powerful method for punctuation and restore capitalization. Varavs et al. [19] used the Transformer model in Latvian and English to perform these tasks simultaneously. In recent years, transformer models have demonstrated the effect of all types of natural language processing [20]–[23]. Binh Nguyen et al. have experimented with the Transformer model to recover the punctuation and capitalization for English [24] and Vietnamese ASR transcripts as well [25]. The results described the difficulty in tasks solved order, i.e. the punctuation or capitalization should be considered firstly.

In this paper, we propose a method for Vietnamese Automatic Speech Recognition to restore simultaneously the punctuation and capitalization tasks. In particular, we focus

on restoring three essential punctuation marks: period (.), comma (,), and question mark (?). In the first section. We will describe an algorithm for Overlapped-Chunk Split and Merging. The proposed model using a combination of Decoder Transformer and CRF for recovering Capitalization and Punctuation will be introduced in the second section. Results and discussion will be introduced statistically in the third section and followed by the conclusion part.

2. Proposed Method

2.1 Capitalization and Punctuation Model

In the previous study, in [25], when recovering the Capitalization task, we used the Transformer model that is type of sequence to sequence model. In this article, we propose a model for recovering ASR Vietnamese's capitalization and punctuation by one using Transformer Decoder combined with CRF. The structure of the model proposed is indicated in Fig. 1. The model has 3 layers: Embedding, Transformer Decoder and CRF, introduce in the following sections.

2.1.1 Embedding Layer

An embedding is a sufficiently low-dimensional space that was transform from high-dimensional vectors. Machine learning models can work easier on large inputs such as words representation with sparse vectors. An embedding preferably absorbs such input semantics by placing semantically connected inputs closely together in the embedding. An embedding can be learned and reusable using templates. We use word level to feed into the embedding layer in our work.

2.1.2 Transformer Decoder

The Transformer was focused primarily on self-attention, which allows the Transformer to interpret each role in the chain differently while maintaining the same parameters over all positions. This helps Transformer to use a constant number of parameters to process arbitrary-length sequences. The model's architecture consists of the encoder-decoder layers, where the encoder and decoder both consist

Fig. 1 Proposed model for capitalization and punctuation

Fig. 2 Proposed model using transformer decoder and CRF for recovering capitalization and punctuation by at one

of multiple Transformer-specific self-attention and position-wise feed-forward layers [26].

The study in the previous work [25] uses Transformer to do the Capitalization and Punctuation task, it works totally fine, but one problem here is it very time consuming. The reason is the Transformer model uses the sequence to sequence architecture, in the decoder step, it has to generate the output one by one and for each word output, it must calculate the feature for all words generated. The drawback not only is time-consuming, it contains a very sensitive problem is we cannot control what will be generated in the decoder process. It means the Transformer model can generate a new word that not exists in the input (for the plain text model) or it can be more encode word than the number of words input (for the encoding model). Because the Capitalization and Punctuation task is all about tagging the format for a word input (uppercase and punctuation belong to the word), so it better to use the tagging model than generating model to do it. In our work, we use only a part of Transformer model, it is Decoder Transformer to do that tagging task. The Fig. 2 is the view of the propose model.

2.1.3 CRF Layer

For the output layer, the model will yield the tag including ‘U’, ‘L’ to indicate uppercase or remain the input word and the tag ‘\$’, ‘.’, ‘,’ ‘?’ to add the punctuation for the input word that it belongs to (‘\$’ mean a word don’t have punctuation). Totally, we will have 8 tag for each word: {U\$, L\$, U., U., U., U., U?, L?}

After the Decoder Transformer block, we can use softmax to determine whether which tag is should be used for the word input. One thing to consider here is the softmax does not care about the order of tag, that means the softmax can output two tag U. and L. stand side by side (almost

meaningless in all cases). To take care about the order of output tag, in this work we use CRF layer.

The Conditional Random Field (CRF) was used in our model. Given a sequence $s = [s_1; s_2; \dots; s_T]$, the corresponding golden label sequence is $y = [y_1; y_2; \dots; y_T]$, and $Y(s)$ represents all valid label sequences. The probability y is calculated as:

$$P(y|s) = \frac{\sum_{t=1}^T e^{f(y_{t-1}, y_t, s)}}{\sum_{y'}^{Y(s)} \sum_{t=1}^T e^{f(y'_{t-1}, y'_t, s)}} \quad (1)$$

where $f(y_{t-1}; y_t; s)$ computes the transition score from y_{t-1} to y_t . The optimization target is to maximize $P(y | s)$.

The encoded text model results in an encoded format which contains seven classes as presented in Table 1. We prepared encoded text format for training data, example in Fig. 3. The format encoded can support the model train and can infer faster than the plain text as it has fixed and very limited vocabulary. However, it does not have much information about the words and the contextual around the word.

2.2 Algorithm for Overlapped-Chunk Split and Merging

We are aware that ASR’s results are endless. Therefore, before feeding to the model, long input sentences are generally divided into fixed-length. While helping to improve inference by independently and parallel processing of chunks, this approach is prone to the lousy prediction of words around the boundary of the split because there is not enough information about both the left and right context in this field. Instead of using a basic chunk splitting method, we introduce the concurrent techniques of overlap chunk-splitting and merging. Our idea aims to ensure every step of the decoding to have enough neighboring words to achieve a better predict model. We proposed System Architecture for

Original data:

Việt Nam có đưa được toàn dân tham gia Chính phủ điện tử hay không?

Input data:

việt	nam	có	đưa	được	toàn	dân	tham	gia	chính
toàn	dân	tham	gia	chính	phủ	điện	tử	hay	không

Plain text output:

Việt	Nam	có	đưa	được	toàn	dân	tham	gia	chính
toàn	dân	tham	gia	Chính	phủ	điện	tử	hay	không?

Encoded output:

U\$	U\$	L\$	L\$	L\$	L\$	L\$	L\$	L\$	L\$
L\$	L\$	L\$	L\$	U\$	L\$	L\$	L\$	L\$	U?

Fig. 3 Data sample of encoded output **Fig. 4** Proposed model architecture for recovering capitalization and punctuation

Recovering Capitalization and Punctuation which consists of three components: overlap chunk split, capitalization and punctuation model, and overlap chunk merging as shown in Fig. 4.

- Firstly, the ASR module's output will be sent to the Overlapped-Chunk Split module to overlap segments.
- Secondly, the Recovery Capitalization model will take the split segments, parallelly process, and produce an output list.
- Ultimately, by using the overlapped Chunk Merging module, the outputs will be merged in the last sentence.

We need to determine the size of the chunk and overlapping which will be split. From the preliminary experiments, we suggested splintering long input sentences into

segments with a k word chunk size and overlapping $k/2$ words into two consecutive chunks.

After consideration, we found that the merge was much more difficult than splitting the overlapped results. Therefore, the question was raised: "How to decide which words will be dropped and which one will be held in a full sentence?". So, we define the `min_words_cut` parameter to specify the number of words at the end of the first chunk and the number of words held in the second chunk. The range is set from 0 to the overlapping scale. If the `min_words_cut` is 0 or equal to the overlapped number, then all overlapping words will be held in the first chunk, and they will be deleted in the second chunk.

As shown in Fig.5, all removed `min_words_cut`

Fig. 5 Overlapped chunk concatenation

and kept `min_words_cut` are highlighted by the red and blue light, respectively. We found that the middle `min_words_cut` part in the overlap chunk has enough context around them to make the model has a better prediction.

3. Experiments and Results

In this paper, we (1) compare the result of Seq2seq, Transformer, Transformer Decoder CRF models on different `chunk_size` with and without chunk-merging (2) affect of the `min_words_cut` parameter and (3) evaluation of speed on the Encode and Plain text output with each model. This shows what the proposed model (Transformer Decoder CRF) will improve compared to the previous models.

3.1 Data Preparation

All data is created automatically by crawling through Vietnamese Internet tools including *vietnamnet.vn*, *dantri.com.vn*, *vnexpress.net*. There was a systematic document on terminology and grammar. Then we just kept the alphabet characters to simulate the ASR output. The standard format for all numbers, info, currencies was retained. From the training data, the input format will be lowercase and without punctuation. Table 1 indicates the number of labels for each punctuation and capitalization class available in the test and training data set.

We performed experiments on Seq2Seq LSTM, Transformer and a new model Transformer Decoder CRF. Our models are implemented based on Fairseq [27] framework. Seq2Seq LSTM and Transformer are encoder-decoder models, each of them has 2 layers in encoder, decoder and have the same hidden size parameters are 1024. One difference of Transformer compare with LSTM is Transformer have the number of heads is 4. To compare in the same conditions, the Transformer Decoder CRF have number of layers is 4 and have the same hidden size are 1024. All three model embedding size is 256. Table 2 show the numbers of parameters of three models.

We train on an NVIDIA 2080ti GPU. The corpus con-

Table 1 Detail of dataset. Uppercase (U), lowercase (L), no punctuation (\$), period (.), comma (,) and question mark (?)

Class	Training	Testing
U	15.4M	74K
L	69.3M	507K
\$	76.6M	525K
.	2.7M	24K
,	5.3M	30K
?	53K	2.6K

Table 2 Compare the number of models parameters

Model	Encode	Plain
LSTM	6.5M	11.3M
Transformer	37M	42M
Transformer Decoder-CRF	7.4M	-

sists of 85 million words. Each example contain 8 to 44 words. Randomized size was between 4 and 22 words for the chunk.

3.2 Evaluation of Chunk-Merging

Figure 6 displays the chart compared to the result of Seq2seq, Transformer, Transformer Decoder CRF models on different `chunk_size` (4-22) with and without chunk-merging.

Clearly, models using chunk-merging always gives better results. Especially, at the propose model is Transformer Decoder CRF with chunk-merging has the highest achievement is 0.88. The results confirm our hypothesis that the model is not enough context for an efficient forecast at the start and end of each sample. This shortcoming can be overcome by adding more context in a way that overlapping and chunk-merging.

It can be seen clearly that, chunk-merging model always provides much better results. Especially, the proposed Transformer Decoder CRF model using chunk-merging illustrates the highest achievement (0.88). The results again confirm our hypothesis that the model is not enough context

Fig. 6 F1-score on different Chunk_size (4-22) using Seq2seq, transformer and transformer decoder CRF with and without chunk merging

Table 3 Comparison of transformer decoder CRF result with and without using chunk merging

Model	Class	Precision	Recall	F1-score
Chunk Merging Transformer Decoder CRF	U	0.90	0.86	0.88
	.	0.71	0.57	0.63
	,	0.66	0.53	0.59
	?	0.75	0.52	0.62
Non-Chunk Merging Transformer Decoder CRF	U	0.89	0.85	0.87
	.	0.69	0.54	0.61
	,	0.65	0.50	0.57
	?	0.74	0.47	0.58

for an efficient forecast at the beginning and the end of each sample. This shortcoming can be overcome by adding more context by overlapping and chunk-merging.

How does chunk_size affect the results? In the seq2seq model, chunk_size is quite sensitive to the accuracy of the output, possibly due to the nature of the encode, decode loss of information despite the attention architecture applied. In the proposed model, the Transformer Decoder CRF, the information is transmitted directly without going through the encode process, chunk_size does not too affect the output.

We provide a table of results for the proposed model - the Transformer Decoder CRF with and without chunk-merging. Inside, we only make statistics in classes ('U', '.', ',', '?'), ignore classes ('L', '\$'), because the exact number is much, not necessary to compare the effectiveness.

As we can see, in Table 3, chunk-merging method is better performance than non-chunk-merging, F_1 -score on all classes are improved from 1% to 5%. The results show that the overlapped words provide the model with more prediction information, and our chunk-merging process can select a proper portion of the overlapping area.

3.3 Evaluation Using Output of Encoded-Text and Plain-Text

The result for the models using encoded text and plain text is compared in Fig. 7. We see that, Seq2seq and Transformer with plain text have better result than using encoded text output.

Nonetheless, the model that uses encoded text is reduced in size and quicker for inference. Besides, if using plain text output, the vocab for the CRF output will become too large. Theoretically would be 8 times the vocab of the input. It should not be reasonable. So, in this article, we only evaluate a new model, the Transformer Decoder CRF using encoded text. The chart shows that the new model gives much better results.

To study the min_words_cut effect on the resulting quality, since we chose chunk-size to be 10, so we experimented with min_words_cut in the range of 0 to 10 in the proposed model - the Transformer Decoder CRF. The results from Fig. 7 demonstrate that the result is the maximum f1-scores in the middle of the chunk size. It shows that the upper and lower predictions of min_words_cut are stable and independent.

Figure 8 displays the confusion matrix of the Transformer Decoder CRF model with Encode output. The matrix shows that "?" is the most difficult class to predict. Besides, the matrix also indicates that the model always predicts efficiently when the input is a word, and lowercase prediction is better than uppercase prediction.

3.4 Evaluation of Speed

The result compares the execution time of 3 models with encoded and plain text output show in Table 4 with 2080ti (GPU), batch_size: 128. The encode text output even shows

Fig. 7 F1-score models with encode and plain text output on different min_word_cut

Fig. 8 Confusion matrix of transformer decoder CRF with encode output

Table 4 Evaluation of speed (tokens/second)

Output	Transformer	Seq2seq	Transformer Decoder CRF
Encode	263s → 2209t/s	217s → 2678t/s	90s → 6457t/s
Plain text	355s → 1637t/s	230s → 2526t/s	—

superior performance when it is used with proposed model - the Transformer Decoder CRF.

4. Conclusion

In this article, we propose a method to recover capitalization and punctuation for Vietnamese ASR output which uti-

lizes the new Transformer Decoder – CRF model combined with a sentence augmentation named Overlapped-Chunk Split and Merging. This combination method aims to improve the ability to extract the longer-range context information so that improves performance while working with the long paragraph. After evaluation, our proposed method presents the superior performance compared to the most re-

cent baseline models in both speed and accuracy. Under the same condition as the original Transformer model, the Transformer Decoder – CRF provides a significantly smaller number of parameters, therefore increases the speed. The chunk-merging method indicates a better performance than the original one at about 1% to 5% of F1-score. Furthermore, using encoded text input also improves the performance of the overall system. In the future work, we will integrate this solution with an ASR model to build an end-to-end model that can transform speech into a useful normalization text document. In addition, as the result of the ASR can be standardized, we can also perform named entity recognition with much better accuracy than raw the ASR output. Finally, we will conduct further experiments on other languages to verify the generalization of the method.

Acknowledgments

This work is supported by the National Science Project under number KC4.0/19-25.

References

- [1] L.V. Lita, A. Ittycheriah, S. Roukos, and N. Kambhatla, "Truecasing," *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pp.152–159, Association for Computational Linguistics, 2003.
- [2] C. Chelba and A. Acero, "Adaptation of maximum entropy capitalizer: Little data can help a lot," *Computer Speech & Language*, vol.20, no.4, pp.382–399, 2006.
- [3] W. Wang, K. Knight, and D. Marcu, "Capitalizing machine translation," *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pp.1–8, 2006.
- [4] R.H. Susanto, H.L. Chieu, and W. Lu, "Learning to capitalize with character-level recurrent neural networks: an empirical study," *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp.2090–2095, 2016.
- [5] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech," *Seventh International Conference on Spoken Language Processing*, 2002.
- [6] O. Tilk and T. Alu, "Lstm for punctuation restoration in speech transcripts," *Sixteenth annual conference of the international speech communication association*, 2015.
- [7] W. Lu and H.T. Ng, "Better punctuation prediction with dynamic conditional random fields," *Proceedings of the 2010 conference on empirical methods in natural language processing*, pp.177–186, 2010.
- [8] N. Ueffing, M. Bisani, and P. Vozila, "Improved models for automatic punctuation prediction for spoken and written text," *Inter-speech*, pp.3097–3101, 2013.
- [9] C. Xu, L. Xie, G. Huang, X. Xiao, E.S. Chng, and H. Li, "A deep neural network approach for sentence boundary detection in broadcast news," *Fifteenth annual conference of the international speech communication association*, 2014.
- [10] O. Tilk and T. Alu, "Bidirectional recurrent neural network with attention mechanism for punctuation restoration," *Interspeech*, pp.3047–3051, 2016.
- [11] A. ktem, M. Farr, and L. Wanner, "Attentional parallel rnns for generating punctuation in transcribed speech," *International Conference on Statistical Language and Speech Processing*, pp.131–142, Springer, 2017.
- [12] G. Szasz, "An audio-based sequential punctuation model for asr and its effect on human readability," *Acta Polytechnica Hungarica*, vol.16, no.2, 2019.
- [13] M.. Tndik, B. Tarj, and G. Szasz, "A bilingual comparison of maxent-and rnn-based punctuation restoration in speech transcripts," *2017 8th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, pp.000121–000126, IEEE, 2017.
- [14] M.. Tndik and G. Szasz, "Joint word-and character-level embedding cnn-rnn models for punctuation restoration," *2018 9th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, pp.000135–000140, IEEE, 2018.
- [15] H. Christensen, Y. Gotoh, and S. Renals, "Punctuation annotation using statistical prosody models," *ISCA Workshop on Prosody in Speech Recognition and Understanding*, 2001.
- [16] A. Gravano, M. Jansche, and M. Bacchiani, "Restoring punctuation and capitalization in transcribed speech," *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp.4741–4744, IEEE, 2009.
- [17] A. Caranica, H. Cucu, A. Buzo, and C. Burileanu, "Capitalization and punctuation restoration for romanian language," *University "Politehnica" of Bucharest Scientific Bulletin*, vol.77, no.3, 2015.
- [18] F. Batista, H. Moniz, I. Trancoso, and N. Mamede, "Bilingual experiments on automatic recovery of capitalization and punctuation of automatic speech transcripts," *IEEE Trans. Audio, Speech, Language Process.*, vol.20, no.2, pp.474–485, 2012.
- [19] A. Vravs and A. Salimbajevs, "Restoring punctuation and capitalization using transformer models," *International Conference on Statistical Language and Speech Processing*, pp.91–102, Springer, 2018.
- [20] J. Devlin, M.W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
- [21] S. Tripathi, C. Singh, A. Kumar, C. Pandey, and N. Jain, "Bidirectional transformer based multi-task learning for natural language understanding," *International Conference on Applications of Natural Language to Information Systems*, pp.54–65, Springer, 2019.
- [22] M. Arkhipov, M. Trofimova, Y. Kuratov, and A. Sorokin, "Tuning multilingual transformers for language-specific named entity recognition," *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pp.89–93, 2019.
- [23] L. Zhang and H. Wang, "Using bidirectional transformer-crf for spoken language understanding," *CCF International Conference on Natural Language Processing and Chinese Computing*, pp.130–141, Springer, 2019.
- [24] B. Nguyen, V.B.H. Nguyen, H. Nguyen, P.N. Phuong, T.-L. Nguyen, Q.T. Do, and L.C. Mai, "Fast and accurate capitalization and punctuation for automatic speech recognition using transformer and chunk merging," *2019 22nd Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA)*, pp.1–5, IEEE, 2019.
- [25] H.N.T. Thu, B.N. Thai, H.N.V. Bao, T.D. Quoc, M.L. Chi, and H.N.T. Minh, "Recovering capitalization for automatic speech recognition of vietnamese using transformer and chunk merging," *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, pp.1–5, IEEE, 2019.
- [26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, . Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, pp.5998–6008, 2017.
- [27] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," *Proceedings of NAACL-HLT 2019: Demonstrations*, pp.48–53, 2019.

Thi Thu Hien Nguyen working at the Thai Nguyen University of Education, Vietnam (2005 - present). Since 2018, has joined the Vietnam Artificial Intelligence System project team in Hanoi, Vietnam. She is now a Ph.D. candidate at the Vietnam Academy of Science and Technology and studying the Natural Language Processing.

Thai Binh Nguyen received the B.S and M.S of Computer Science at Hanoi University of Science and Technology in 2017 and 2019, respectively. Currently, he working as a Machine Learning Engineer in VinBigdata, Vietnam.

Ngoc Phuong Pham received the BS of Computer technology and MS of Computer Science at Thai Nguyen University in 2007 and 2013. Currently, he is a PhD student of Information Systems at Graduate University of Science and Technology. He is working at the Learning Resource and Information Technology Center - Thai Nguyen University. His research interests include Speaker Adaptation, Speech Synthesis, Networking and Security.

Quoc Truong Do received his B.E. from University of Engineering and Technology, Hanoi, Vietnam, in 2013, and his M.S. and Ph.D. degree from the Graduate School of Information Science, NAIST, Nara, Japan in 2015 and 2018. He is currently the CEO of Vietnam Artificial Intelligence Solutions company. He interested in speech and natural language processing, with a focus on speech recognition, and speech translation.

Tu Luc Le received M.S degree at Hanoi University of Science and Technology (HUST) in 1999. He worked as director of Center for Informatics - Public Gazette of Hanoi City from 2006 to 2018, Deputy Chief of the Office of the Hanoi People's Committee 2018 up to now.

Chi Mai Luong received the B.Sc. degree from the Faculty of Applied Mathematics, Kishinev University (Soviet Union former) and the Ph.D. from the Institute of Information Technology (IOIT), in 1981 and 1991 respectively. She joined the Laboratory of Pattern Recognition, IOIT, in 1982 and was promoted to an Associate Professor in 2005. Her research interests include pattern recognition, machine learning, speech recognition and synthesis. She published numerous scientific papers in the fields and received the Kovalevskaja Prize for her Outstanding Contribution on Research and Development in Vietnam, in 2010. She is a member of IEEE.