# A Vietnamese Language Model Based on Recurrent Neural Network

Viet-Trung Tran*, Kiem-Hieu Nguyen*, Duc-Hanh Bui†
Hanoi University of Science and Technology
*Email: {trungtv,hieunk}@soict.hust.edu.vn
†Email: hanhbd.hedspi@gmail.com

***Abstract*—Language modeling plays a critical role in many natural language processing (NLP) tasks such as text prediction, machine translation and speech recognition. Traditional statistical language models (e.g. n-gram models) can only offer words that have been seen before and can not capture long word context. Neural language model provides a promising solution to surpass this shortcoming of statistical language model. This paper investigates Recurrent Neural Networks (RNNs) language model for Vietnamese, at character and syllable-levels. Experiments were conducted on a large dataset of 24M syllables, constructed from 1,500 movie subtitles. The experimental results show that our RNN-based language models yield reasonable performance on the movie subtitle dataset. Concretely, our models outperform n-gram language models in term of perplexity score.**

## I. Introduction

One of fundamental tasks in NLP is statistical language modeling (SLM), statistically formalized as inferring a probability distribution of word sequence. SLM has many applications such as machine translation, speech recognition and text prediction [1], [2].

N-gram has been *de-facto* for language modeling [3]. An n-gram model predicts the next word using $(n-1)$ preceding words in the context $p(w|w_0...w_{n-2})$. Such probability is estimated by occurrence frequency of word sequences of length $n$ from a large text collection. At $n = 2, 3, 4$, or $5$, the model captures frequent word patterns as long as enough training texts are provided. However, as $n$ increases, it faces the curse of dimensionality issue.

- Firstly, given a context, it could only predict a word that has been "*seen*" before with the context in the training texts.
- Secondly, it can only *"see"* a few words back, i.e. $(n-1)$ preceding words. A longer context, such as previous sentences, couldn't be considered in predicting the next word. This is not intuitively the way human learn and capture a context.

Neural language models, i.e. neural networks-based language model, successfully surpassed these two shortcoming [4], [5].

For the first shortcoming, similarly to an n-gram model, a feed-forward neural network (FFNN) predicts the next word using preceding $n-1$ words context [4]. The idea is distributed word representations learned by the network can capture words similarity. For instance, by capturing similarity between '*con chó*' (*dog*) and '*con mèo*' (*cat*), '*căn phòng*' (*room*) and '*phòng ngủ*' (*bedroom*), '*chạy*' (*running*) and '*đi*' (*walking*), and so on, the model could generate the sentence "*Một con chó đang chạy trong căn phòng*" (*A dog was running in a room*) given the sentence "*Con mèo đang đi trong phòng ngủ*" (*The cat is walking in the bedroom*). Their model reportedly outperforms n-gram language model on benchmark datasets.

For the second shortcoming, although the avantage of recurrent neural networks (RNNs) over Feedforward neural networks (FNNs) is still a debating subject [6], RNNs explicitly allow learning from arbitrary long context using a recurrent mechanism within the network [5], [7]. The long-term context is particularly useful in learning via the long-short-term memory architecture [8]. Experimental results reported in [5] show that an RNN-based language model outperforms its FFNN-based counterpart.

One may argue for n-gram approach for its simplicity and efficiency. But as hardware is more and more evolving; and efficient learning algorithms for neural networks such as [9] are proposed, neural language modeling is promising. In another direction, [10] proposes a multi-modality framework for jointly modeling images and texts in a unified neural language model.

In this work, we investigate in experimenting RNNs for Vietnamese language model. Our contributions are summarized as follows:

- Building an RNN-based syllable-level language model for Vietnamese.
- Building an RNN-based character-level language model for Vietnamese.
- Running extensive experiments on a 24M syllables dataset constructed from 1,500 movie subtitles.

The rest of the paper is organized into four sections: Section II discusses our main contributions of building RNN-based language models for Vietnamese; Section III records our test-bed and results of the experiments. Related works are introduced in Section IV. We conclude the paper with final remarks in Section V.

## II. RNN-based language modeling for Vietnamese

In this section, we first present RNN-based language modeling (Section II-A. We then briefly introduce some important characteristics of Vietnamese (Section II-B). Finally, we proposed RNN-based language models at both character and syllable levels for Vietnamese (Sections II-C and II-D).

### A. RNN

Given a training sequence $(x_1, ..., x_T)$, an RNN computes the output vectors $(o_1, ..., o_T)$ and then applies the $softmax$ function which maximizes the total log probability of the obtained predictive distribution $P(x_{t+1}|x_0x_1...x_t)$. In a simple configuration, our RNN is constructed with three layer: an input layer $x$, a hidden layer $s$ and an output layer $y$. At a time $t$, input, output and hidden layer state of the network are denoted as $x(t)$, $o(t)$, and $s(t)$ respectively. As our network is recurrent, the previous hidden layer state $s(t-1)$ is intentionally input to the hidden layer $s(t)$. Mathematically, these layers are computed as follows:

$$s(t) = f(Ux(t) + Ws(t-1)) \quad (1)$$

$$o(t) = softmax(Vs(t)) \quad (2)$$

$U$, $W$, $V$ denote the parameters of our RNN to be learn. $f$ is an activation function such as $sigmoid$, $ReLU$, and $tanh$. Our network and its unfolding in time of the computation are shown in Figure 1.

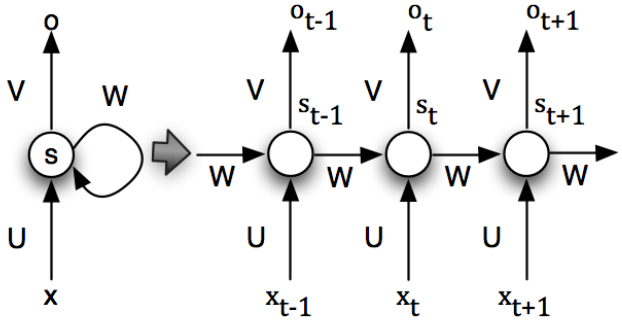


Figure 1. An RNN and its unfolding in time of the computation

RNN has the following characteristics:

- The hidden state $s(t)$ is seen as the memory of the network. $s(t)$ captures all the information of the previous time steps. In practice due to the complexity of the training algorithm, $s(t)$ computation only takes into account a limited number of recently previous steps.
- RNN is different from other type of neural network in which it uses the same parameters across all steps. More precisely, $(U, V, W)$ are shared and are independent of the inputs. This reduce the number of parameters to learn.

As clearly discussed in [11], training a RNN with gradient decent is practically difficult. This means that RNN is hard to learn long-term dependencies as the gradient blows up or vanishes exponentially over time. This is called the vanishing gradient problem. To address this problem, the hidden layer within our RNN will be implemented with a so-called long short term memory (LSTM) layout.

LSTM cells (in hidden layer) are designed to be capable of learning long-term dependencies (Figure 2). It is first introduced in [12]. Each LSTM cell consists of three gates to control the information into the cell and from the cell output back to the network. The forget gate decides whether the cell state at time $(t-1)$ will be filtered out or not. The input gate regulates the inputs. Similarly, the output gate activates or deactivates its outputs to the network (Figure 2).

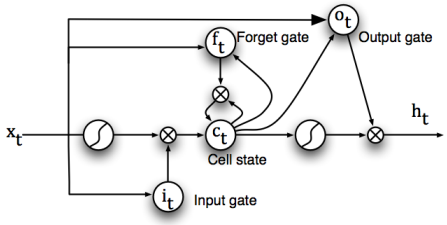


Figure 2. An LSTM cell

Beside powering RNN with LSTM cells, an RNN can also be stacking to form a deeper neural network. In this setting, the output of the first layer will become the input of the second and so on, as depicted in Figure 3. By going deeper, RNN can capture more context, thus, better modeling the sequence distribution [13]. Of course, this increases the model parameters, training time, and requires more input dataset.

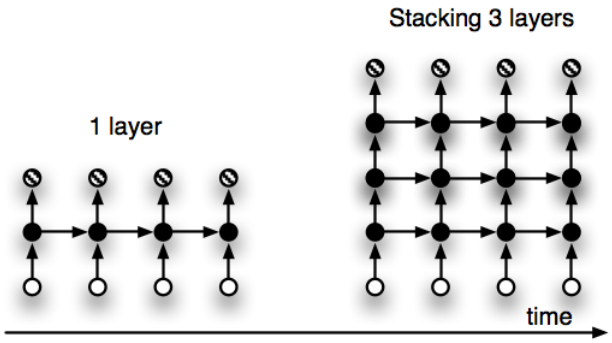


Figure 3. Stacking multiple networks

### B. Characteristics of Vietnamese language

In this section, we briefly describe some important characteristics of Vietnamese regarding under-word level language modeling task.

Vietnamese scripts are written in Latin characters ('*kí tự*' or '*chữ*' in Vietnamese). There are 29 characters in Vietnamese alphabet, including 12 vowels and 17 consonants. There are five accent marks in Vietnamese. An accent mark could be attached to a vowel to modify pronounce tone, resulting in 60 additional '*vowel*' variants (Table I [14]).

Characters are grouped into syllables ('*âm tiết*' in Vietnamese). Syllables, in turn, are grouped into words, which are the smallest meaningful unit in a language. Words could be

| Initial char | Variants |
|---|---|
| a | [a, à, á, ả, ã, ạ, ă, ằ, ắ, ẳ, ẵ, ặ, â, ầ, ấ, ẩ, ẫ, ậ] |
| e | [e, è, é, ẻ, ẽ, ẹ, ê, ề, ế, ể, ễ, ệ] |
| o | [o, ò, ó, ỏ, õ, ọ, ô, ồ, ố, ổ, ỗ, ộ, ơ, ờ, ớ, ở, ỡ, ợ] |
| u | [u, ù, ú, ủ, ũ, ụ, ư, ừ, ứ, ử, ữ, ự] |
| i | [i, ì, í, ỉ, ĩ, ị] |
| y | [y, ỳ, ý, ỷ, ỹ, y] |

Table I
VOWEL CHARACTERS AND VARIANTS IN VIETNAMESE ALPHABET.

composed of a single syllable, e.g. '*mua*' (to buy), or multiple syllables, e.g. '*nghiên cứu*' (research). Typically, words are composed of two syllables, standing for 70% of words in a general-domain Vietnamese dictionary [15].

Due to complicated originality of Vietnamese morphology and lexicon, word segmentation is a fundamental yet challenging task in Vietnamese language processing [16]. In this paper, we shall let word-level language model for further investigation. We instead focus on sub-word levels, i.e. char and syllable levels.

### C. Character-Level Language Model

At character level, our objective is to predict the next character given a sequence of previous characters. The input $X$ is thus a character sequence. $x(t) \in X$ is the character at time $t$. Since the number of character (char-level vocabulary size $V$) is small, we directly use the *one-hot-encoding*. Each $x(t)$ is a vector of size $V$ with the component corresponding to the character itself having value *one* and all the other components having value *zero*.

The output $o(t)$ at each time step $t$ is appropriately a vector of size $V_{char}$. Specifically, each component of $o(t)$ indicates the probability of generating that character giving the previous context from 0 to $t$. Given $h\_len$ as size of the hidden layer $s$ (Section II-A), followings are domain space of our learning parameters:

$$\begin{aligned} x(t) &\in \mathbb{R}^{V_{char}} \\ o(t) &\in \mathbb{R}^{V_{char}} \\ s(t) &\in \mathbb{R}^{h\_len} \\ U &\in \mathbb{R}^{V_{char} \times h\_len} \\ V &\in \mathbb{R}^{h\_len \times V_{char}} \\ W &\in \mathbb{R}^{V_{char} \times V_{char}} \end{aligned} \tag{3}$$

### D. Syllable-Level Language Model

At syllable level, our objective is to predict the next syllable given a sequence of previous syllables. Therefore, $x(t)$ would be the *one-hot encoding* of the syllable at time $t$. We observe that the number of all Vietnamese syllables, $V_{syl}$, is about 7000. Given $V_{syl}$ and $h\_len$ as sizes of syllable vocabulary and the hidden layer $s$, respectively, the domain space of our learning parameters is the same as in Equation 3 except with a much bigger $V_{syl}$.

## III. EXPERIMENTS

### A. Datasets

This work focuses on conversational data instead of general texts (e.g. from news articles). Therefore, we have manually collected movie subtitles from 1,500 movies available on the Internet. Subtitles were contributed by volunteers translators from its original languages to Vietnamese. For instance, three sequences {"*How many bags can you carry?*", "*If you don't think for yourself*", "*think for your mother!*"} were extracted from the part as shown in Figure 4.

132
00:07:33,920 –> 00:07:35,560
*Chú có thể vác được bao nhiêu bao cơ chú?*
133
00:07:35,680 –> 00:07:36,760
*Nếu chú không nghĩ đến bản thân...*
134
00:07:36,880 –> 00:07:39,280
*...thì cũng phải nghĩ đến mẹ ở nhà!*

Figure 4. A sample movie subtitle original file with triples of (frame number, frame offset, and subtitle).

We picked movie subtitle for two reasons. First, it is widely available and public over Internet. Second, the language itself is close to conversational Vietnamese, generated by thousands of contributors, thus barely imply specific writing style.

The following steps were processed to clean data:

- Tokenizing: Each sequence is tokenized into tokens using a regular expression processor. A token could be a syllable, a foreign word, or a punctuation mark.
- Detecting and removing incomplete translations: We could not expect high commitment from volunteer translators. In fact, we found many subtitles which were partially translated. Fortunately, most of the origin subtitles were in English. Such origin subtitles obviously generate noise when learning a language model for Vietnamese. In order to detect and remove such subtitles, we use Princeton WordNet ([17]) to extract an English vocabulary. We then scan each subtitle for tokens that could be found in the vocabulary. If a half of all tokens matches this criteria, the subtitle is considered untranslated. We also used *langdetect* package[1] to filter out sentence others than in Vietnamese. Manual observation on a random subset of removed subtitles shows that this heuristic filter works sufficiently for our purpose.
- Normalizing foreign words: For tokens that were detected by the English vocabulary, the tokens were all assigned the value '*OOV*'.
- Adding start symbol: Each sequence in our dataset is a subtitle displayed in a movie frame. As seen in conversational data and particularly movie subtitle data, a sequence is not necessarily a complete sentence or

[1] https://pypi.python.org/pypi/langdetect

syntactic phrase in linguistics. Unlike general texts, this characteristic is challenging for language modeling. To signify the beginning of a sequence, we added a '*START*' symbol ahead.

- Punctuation: We keep all punctuation as in original subtitles.

*Data for syllable models*: Since the inputs to our RNNs are vectors, not strings, we create a mapping between syllables and indices, INDEX_TO_SYLLABLE, and SYLLABLE_TO_INDEX. For example, a syllable '*anh*' has index 1043, which holds two entries in the two mapping tables. A training sample $x$ looks like $[0, 56, 78, 454]$ where 0 is the index of '*START*'.

*Data for char models*: Similarly to syllable models, a mapping between characters and indices have to be created using two mapping tables, INDEX_TO_CHAR and CHAR_TO_INDEX. Again, 0 is the index of '*START*'.

### B. Evaluation

We built four neural language models on the Tensorflow framework [18], with three variable parameter values $10, 20, 40$ of the sequence length:

- *Char-1-layer-40-seq*: A simple char-level model with only one hidden layer and sequence length of 40.
- *Char-3-layer-40-seq*: A more complicated char-level model with three hidden layers and sequence length of 40.
- *Syl-1-layer-[10,20,40]-seq*: Three variant of a simple syllable-level model with one hidden layer and sequence length of 10, 20, and 40, respectively.
- *Syl-3-layer-[10,20,40]-seq*: Three variant of a deep syllable-level model with three hidden layers and sequence length of 10, 20, and 40, respectively.

*BASELINE*: We used a bi-gram language model as baseline[2]. To confront unknown words, we used an add-one Laplace smoothing.

| Model | PPL |
|---|---|
| Char-1-layer-40-seq | 116 |
| Char-3-layer-40-seq | 109 |
| Syl-1-layer-10-seq | 85.6 |
| Syl-3-layer-10-seq | 89.1 |
| Syl-1-layer-20-seq | **83.8** |
| Syl-3-layer-20-seq | 95.5 |
| Syl-1-layer-40-seq | 99.7 |
| Syl-3-layer-40-seq | 95.7 |
| Bi-gram | 145 |

Table II
PERPLEXITY PER SYLLABLE ON OUR MODELS.

We measured the perplexity which is the inverse probability of the input sequence. The perplexity is computed as in 4:

$$PPL = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1...w_{i-1})}} \quad (4)$$

[2]Note that this is not the most accurate in n-gram approach. However, we select this model because it is easy to implement, parameter-free, and is a relatively strong baseline.

Where $N$ is the number of words and $P(w_i|w_1...w_{i-1})$ is the probability of word $w_i$ given the previous sequence $w_i...w_{i-1}$.

As shown in Table II, all neural models significantly outperform the bi-gram baseline. Our best model obtained a perplexity of 83.8 for syllable-based, one layer and trained with RNN sequence length of 20. Note that RNN sequence length signifies the maximum size of an unfolded recurrent network, which means the largest context used to generate a new character/syllable. This shows that increasing the context size to 40 in syllable model does not help reduce overall perplexity. This could be due to the nature of our training corpus: it contains mostly conversational short sentences. We let the investigation of this hypothesis as a future work.

Intuitively, the character language models in our settings are not better than the syllable models as they have to first predict multiple characters in order to form a syllable. Stacking 3-layers RNN increases the number of learning parameters, thus giving the models more expressive power to predict new words. However, in the cases of *Syl-3-layer-[10,20]-seq*, more number of learning parameters hurts perplexity scores.

We show below in Figure 5 several predictions from our best-performed syllable model i.e. *Syl-1-layer-20-seq* in Table II.

START *cô*
Suggestions 1: *ấy*, 2: *OOV*, 3:

START *cô ấy*
Suggestions 1: , 2: *bị*, 3: *đã*

START *cô ấy là*
Suggestions 1: *OOV*, 2: *một*, 3: *người*

START *cô ấy là bác*
Suggestions 1: *sĩ*, 2: *sỹ*, 3:

START *cô ấy là bác sĩ*
Suggestions 1: , 2: *OOV*, 3: *của*

START *cô ấy là bác sĩ tâm*
Suggestions 1: *thần*, 2: *lý*, 3: *lí*

START *cô ấy là bác sĩ tâm lý*
Suggestions 1: , 2: *của*, 3: *OOV*

Figure 5. A sample syllable prediction that suggested 3 options with highest probabilities for the sentence: "cô ấy là bác sĩ tâm lý" (she is a psychiatrist).

## IV. RELATED WORK

This section discusses related works in language modeling for Vietnamese. Language modeling is an important NLP tasks. However, to our knowledge, only a handful works focus on this task for Vietnamese [19], [20], [21].

In a pioneering work on language modeling for automatic speech recognition in Vietnamese, [19] applies n-gram statistical language modeling to train on a corpus of 10M sentences

collected from Web documents. They report a perplexity per word of 252 on the best configuration. *A priori* and *a posterior* methods for calculating perplexity over n-gram language models for Vietnamese are also discussed in [20].

Perhaps closer to our direction is the work presented in [21]. They investigate neural language model at syllable level. Experiments were conducted on a corpus of 76k sentences in Vietnamese text. As reported, a feed forward neural language model achieves a perplexity of 121.

Word-level RNN language models can also be built for Vietnamese providing word segmented texts. Both word and syllable-level language models have the drawback of learning new words. They need to be retrained upon adding new words to the vocabulary. On the other hand, character level language model has no problem with learning new words since words are constructed by only predefined characters. Hence, a character level language model not only predicts the next word, but also completes current word. We also plan to investigate as embedding vectors trained from a large text collection to better capture word semantic of meaning and to reduce vector size.

## V. Conclusions

This paper presents a neural language model for Vietnamese. We investigate RNNs at both character and syllable levels. Experimental results show that the best configuration yields a perplexity of 83.8, which is a reasonable performance for a language model. All of our models outperform n-gram language model in the experiments.

For future work, we plan to investigate RNNs at word level. We anticipate that embedding representations using techniques like Convolutional Neural Networks will also improve both accuracy and efficiency of our approach.

## Acknowledgment

This work is supported by Hanoi University of Science and Technology (HUST) under project No. T2016-PC-047

## References

[1] R. Rosenfeld, "Two Decades Of Statistical Language Modeling: Where Do We Go From Here?" in *Proceedings of the IEEE*, vol. 88, 2000. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.463

[2] K. Yao, G. Zweig, M. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013*, 2013, pp. 2524–2528. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2013/i13_2524.html

[3] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ser. ACL '96. Stroudsburg, PA, USA: Association for Computational Linguistics, 1996, pp. 310–318. [Online]. Available: http://dx.doi.org/10.3115/981863.981904

[4] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944966

[5] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, 2010, pp. 1045–1048.

[6] M. Sundermeyer, I. Oparin, J.-L. Gauvain, B. Freiberg, R. Schl
"uter, and H. Ney, "Comparison of feedforward and recurrent neural network language models," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vancouver, Canada, May 2013, pp. 8430–8434.

[7] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *2012 IEEE Spoken Language Technology Workshop (SLT), Miami, FL, USA, December 2-5, 2012*, 2012, pp. 234–239. [Online]. Available: http://dx.doi.org/10.1109/SLT.2012.6424228

[8] M. Sundermeyer, R. Schl
"uter, and H. Ney, "Lstm neural networks for language modeling," in *Interspeech*, Portland, OR, USA, Sep. 2012, pp. 194–197.

[9] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*, 2005. [Online]. Available: http://www.gatsby.ucl.ac.uk/aistats/fullpapers/208.pdf

[10] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Multimodal neural language models," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 595–603. [Online]. Available: http://jmlr.org/proceedings/papers/v32/kiros14.html

[11] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[13] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks." *CoRR*, vol. abs/1312.6026, 2013.

[14] K. Nguyen and C. Ock, "Diacritics restoration in vietnamese: Letter based vs. syllable based model," in *PRICAI 2010: Trends in Artificial Intelligence, 11th Pacific Rim International Conference on Artificial Intelligence, Daegu, Korea, August 30-September 2, 2010. Proceedings*, ser. Lecture Notes in Computer Science, B. Zhang and M. A. Orgun, Eds., vol. 6230. Springer, 2010, pp. 631–636.

[15] D. Dinh, P. Pham, and Q. Ngo, "Some lexical issues in building electronic vietnamese dictionary," in *Proceedings of PAPILLON-2003 Workshop on Multilingual Lexical Databases*, 2003.

[16] H. P. Le, N. T. M. Huyên, A. Roussanaly, and H. T. Vinh, "A hybrid approach to word segmentation of vietnamese texts," in *Language and Automata Theory and Applications, Second International Conference, LATA 2008, Tarragona, Spain, March 13-19, 2008. Revised Papers*, ser. Lecture Notes in Computer Science, C. Martín-Vide, F. Otto, and H. Fernau, Eds., vol. 5196. Springer, 2008, pp. 240–249.

[17] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995. [Online]. Available: http://doi.acm.org/10.1145/219717.219748

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[19] V. B. Le, B. Bigi, L. Besacier, and E. Castelli, "Using the web for fast language model construction in minority languages," in *8th European Conference on Speech Communication and Technology, EUROSPEECH 2003 - INTERSPEECH 2003, Geneva, Switzerland, September 1-4, 2003*. ISCA, 2003.

[20] L. Q. Ha, T. T. T. Van, H. T. Long, N. H. Tinh, N. N. Tham, and L. T. Ngoc, "*A Posteriori* individual word language models for vietnamese language," *IJCLCLP*, vol. 15, no. 2, 2010. [Online]. Available: http://www.aclclp.org.tw/clclp/v15n2/v15n2a2.pdf

[21] A. Gandhe, F. Metze, and I. R. Lane, "Neural network language models for low resource languages," in *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, H. Li, H. M. Meng, B. Ma, E. Chng, and L. Xie, Eds. ISCA, 2014, pp. 2615–2619.