DaNangNLP Toolkit for Vietnamese Text Preprocessing and Word Segmentation

Nguyen Ket Doan, Nguyen Tran Tien, Nguyen Duc Bao, Ton That Ron, Vo Van Nam, Pham Van Nam, Phung Anh Sang, Huynh Cong Phap and Nguyen Huu Nhat Minh*

The University of Danang, Vietnam-Korea University of Information and Communication Technology, Vietnam
nkdoan.20it7@vku.udn.vn,nttien.20it6@vku.udn.vn,
ndbao.20it6@vku.udn.vn,
rontt.21it@vku.udn.vn,namvv.21it@vku.udn.vn,
nampv.21it@vku.udn.vn,
sangpa.21it@vku.udn.vn,hcphap@vku.udn.vn,nhnminh@vku.udn.vn

Abstract. Recent research has focused on Vietnamese large language models, however, the preprocessing steps play important complementary roles in the future success of Vietnamese language processing. In this paper, we design and develop a novel DaNangNLP toolkit that could cope with important Vietnamese language preprocessing steps. Although there have been many successful modules on Vietnamese language processing, existing toolkits still exhibit certain shortcomings, especially for word segmentation in complex Vietnamese sentences. Therefore, we have developed a practical and robust natural language processing pipeline specifically tailored for the Vietnamese language to address the challenging issues present in previous Vietnamese processing toolkits. The DaNangNLP pipeline based on the novel built-in word dictionaries is designed to handle Vietnamese text for typical preprocessing steps such as sentence segmentation, word regex, word normalization, and word segmentation. Throughout the evaluation, the proposed semantic-based word segmentation has outperformed the frequency-based word segmentation and existing toolkits in complex sentences.

Keywords: Sentence Segmentation, Regular Expression, Word Segmentation, Word Normalization, Vietnamese Language Processing.

1 Introduction

Natural Language Processing [1] (NLP) has become a fundamental research trend in the development of various language processing technologies, ranging from machine

* Corresponding author

 ISBN: 978-604-80-9774-5

translation and sentiment analysis to information retrieval and text generation. While NLP has seen remarkable progress in major languages such as English, Chinese, French, etc… However, Vietnamese language processing still faces unique challenges due to our complicated linguistic characteristics, grammar, and limited training resources. Specifically, Vietnamese is a tonal, high polysemy, and complicated word structure, making it particularly challenging for language processing tasks. One of the crucial steps in Vietnamese language processing is word tokenization, which involves segmenting the continuous text into individual words or tokens. Accurate tokenization is essential for other language tasks, as it directly impacts the performance of language models and other downstream applications. The recent huge attention to developing powerful NLP pre-trained models has shown outstanding superiority in processing English and promising performance in Vietnamese languages. However, after our trials, many errors can be easily detected in existing Vietnamese tokenizers, especially in complex sentences due to the highly contextual dependency and different meanings of polysemous words. As a complement to the Vietnamese language models, we recognized the crucial role of data preprocessing such as word tokenizer might impact the overall performance of language tasks, especially word-based Vietnamese language models such as PhoBERT [8]. As a result, in this work, we propose a novel word preprocessing toolkit, namely DaNangNLP to enhance the quality of the word tokenizer based on the novel built-in dictionaries and contextual-based word segmentation.

In Vietnamese, there are many compound words formed by combining single words depending on the context which would challenge the language models to understand and represent them correctly. There are many ways to combine single words into compound words, each of which causes our sentence to have a different meaning. Therefore, word tokenization in Vietnamese faces many difficulties. Vietnamese processing has differences between syllable-based and word-based tokens. This ambiguity comes from the fact that the white space is also used to separate syllables that constitute words.

E.g.: A 6-syllable written text "Tôi là một nghiên cứu viên" (I am a researcher) forms 4 words "Tôi là một nghiên_cứu_viên". "Tôi thấy ông già đi nhanh quá", this sentence can be understood in another meaning in Vietnamese such as "Tôi thấy ông_già đi nhanh quá".

In this research, we focus on designing a practical DaNangNLP toolkit that is tailored for the Vietnamese language in general written Vietnamese texts. The pipeline as shown in Fig. 1 comprises multiple essential stages of text preprocessing. First, the text will go through *sentence segmentation* to divide the sentences of input text correctly. In this step, we design a regular expression [2] (i.e., regex) to split sentences when encountering sentence-ending punctuation marks such as "\n", ".", "!", "?", except for some special cases like job titles, city names, etc. Next, we adopt another *regular expression* to extract the full tokens presenting for email, date, URL, etc. Subsequently, we perform *word normalization* techniques to standardize different word variants and word acronyms based on built-in dictionaries and regex, thus enhancing linguistic consistency. Some necessary steps for word normalization of the preprocessing for the popular raw Vietnamese texts are as follows:

- Because the dataset could be taken from some unofficial public sources, there will be words that do not follow the Vietnamese grammar standards. Some common

forms such as abbreviations, use spontaneous units of measurement. For example: "Cái bánh này giá 50k phải ko?"

- One of the biggest differences between Vietnamese and English is the sign of words, which is also part of what makes Vietnamese more difficult in both spoken and written language. Punctuation marks in Vietnamese are typed with the characters "s", "f", "j", "x", "r", "x", because these punctuation marks are written with letters of the Latin alphabet. Therefore, it is easy for the writer to mistype the position of the bar in the word. In natural language processing in general and Vietnamese in particular, when the sign of a word is displayed in the wrong position, it can create a token of another word, leading to the case where 2 words with the same sign but in the wrong position will be represented as 2 separate tokens. If we don't process, the above issues may impact the language embedding model and reduce the model's efficiency. The tone mark module as a filter will filter through the text, with the marking rules specified, the module will find and replace the wrong words.

For the word segmentation, we conduct two approaches based on the context of the words such as the frequency and semantics of words regarding the surrounding sentences. As a result, the proposed DaNangNLP toolkit can effectively handle the many complex cases of Vietnamese texts.

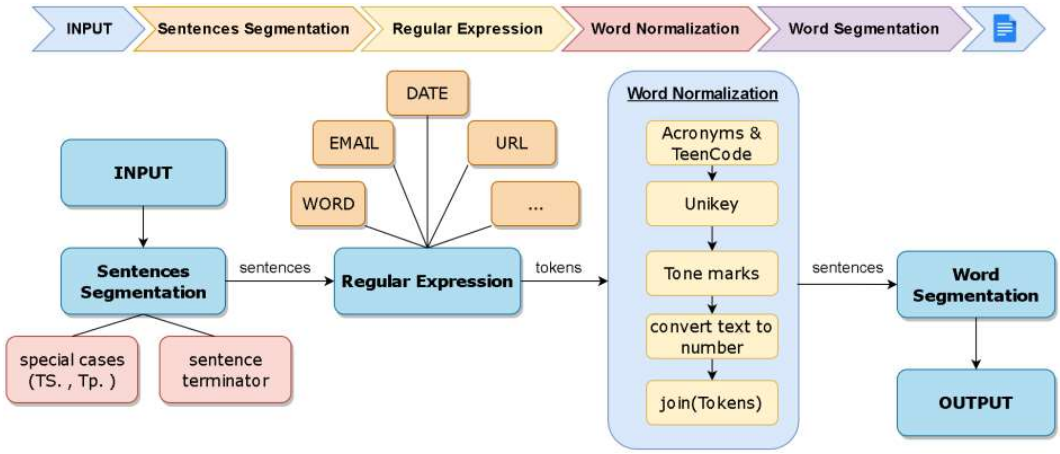

Figure 1. The proposed pipeline for Vietnamese pre-processing and word segmentation.

2 DaNangNLP pipeline for Preprocessing Vietnamese Text

As shown in Figure 1, we propose a text processing process before entering the models including the following components: input is unnormalized Vietnamese text put into the function of sentence separation to process words sentence by sentence, then each sentence will be passed to the regex function to separate each token(word) according to the patterns defined in the source code from those tokens to the word normalize function to word normalization (detailed steps will be discussed below) and the result is single word tokens (1 syllable) and compound words (multiple syllables) labeled with the "_" sign (e.g: "Trong tiết học, học sinh học rất tốt" -> "Trong tiết_học, học_sinh học rất tốt").

 ISBN: 978-604-80-9774-5

2.1 Sentences Segmentation

Sentence segmentation is one of the important parts of NLP in general and Vietnamese in particular. It is the first step in many NLP tasks that are more elaborate such as detecting and normalizing errors in input text. The sentence segmentation task aims to precisely identify these boundaries, ensuring that each sentence is considered an independent unit for further analysis. The Vietnamese language uses punctuation marks to segment sentences (period ".", exclamation mark "!", question mark "?") along with the distinction between uppercase and lowercase letters, which also contributes to the segmentation of sentences. However, these punctuation marks can be ambiguous, especially when there are abbreviations, acronyms, or ellipses. To handle this segmentation task, we deal with the aforementioned punctuation-splitting cases. However, in Vietnamese, there are special cases where the dot "." to serve abbreviations (for example, Tiến sĩ Nguyen Huu Nhat Minh is often written as TS. Nguyen Huu Nhat Minh, or Thua Thien Hue is often written T.T.Hue), so it is also necessary to handle these cases because they significantly affect the task of sentence segmentation. Besides, for the ellipsis, if the ellipsis (i.e., '...') is followed by a lowercase letter, the sentence will not be split. If the ellipsis is followed by a capital letter, it will separate the sentence.

Therefore, we built a dictionary of abbreviations and their full written forms. To handle these abbreviations, it is necessary to browse the words in the input text sentence to compare with the words in the dictionary, if any are in the dictionary, the acronym will be converted to its full form. Thereafter, we write regular expressions that define sentence-ending rules. In addition, this expression also ensures that sentences will not be separated by periods of abbreviations, i.e., TS. Nguyen Huu Nhat Minh will not be split by periods.

We tested existing Vietnamese processing libraries and APIs, and most of these approaches ignore the special cases in the sentence segmentation task. To clearly see the difference, we use the following two sentences as an example: TS. Nhật Minh rất vui vẻ, hòa đồng... Ông ấy cũng chính là thầy của tôi." In this case, our toolkit will split into two separate sentences: "TS. Nhật Minh rất vui vẻ, hòa đồng..." and "Ông ấy cũng chính là thầy của tôi.." Meanwhile, the Underthesea library produced the results "TS. Nhật Minh rất vui vẻ, hòa đồng... Ông ấy cũng chính là thầy của tôi.." It can be seen that the Underthesea library cannot separate the above two sentences in case the first sentence ends with an ellipsis. Another Vietnamese processing API, Viettel, separates "TS." and "Nhật Minh" because it cannot recognize "TS." is an acronym and produces inaccurate results. As a result, the accuracy of these models can be affected, leading to subsequent errors in understanding texts. Our model not only handles basic cases but also many special cases that other models skipped. This ensures that subsequent NLP tasks can operate correctly on segmented sentences, leading to more reliable results. Therefore, the accuracy of this model is also significantly higher than existing Vietnamese processing models.

2.2 Tokenization via Regular Expression

A tokenizer is a component of the NLP pipeline that is responsible for breaking down a text input into individual tokens, such as words or punctuation marks. To tokenize the

raw text, first, it is split based on whitespace characters. Then, the tokenizer processes the text from left to right and results in the list of substrings, acronyms, and tokens. In doing so, the first check of the tokenizer looks for tokenizer exception rules, such as splitting "lần/ngày" into "lần", "/" and "ngày" or keeping "PGS.TS" as one token. Or keep "11:15:30 SA" as one token. The second check examines prefixes, suffixes, or infixes like commas, periods, hyphens, or quotes and splits them off if appropriate. The last check hyphenated words, and symbols. Whenever a match is found, the tokenizer applies the rule and continues looping, starting with the newly split substrings. As a result, they can effectively tokenize complex, nested tokens such as those containing abbreviations and multiple punctuation marks.

Regular Expression in Underthesea

This typical tokenization technique known as RegEx is a sequence of characters that specifies a match pattern in text and is being utilized in various language models. In this step, we extend the Underthesea toolkit [3] by adding many special cases for regular expression. Underthesea begins with a limited pattern including special tokens used by the symbol and the Vietnamese character. For example, initially the word "đồng/tháng" is split into a set of tokens as “đồng”, “/”, “tháng” or “10:30 AM” which is also split into a set of tokens as “10”, “:”, "30", "SA", ... Consequently, regex algorithm defines the merging rules instead of choosing the most common pairs, it split tokens using the following:

f"(?<=[.,\d]\s)({CHARAC}+/+{CHARAC}+)+", etc.

After the merge, the regex algorithm stores the tokens as the following example lists:

['15', 'triệu', 'đồng/tháng']

['10:01:22 SA', '03.07.2023', '14H20P']

['PGS.TS', 'Huỳnh', 'Công', 'Pháp', 'là', 'hiệu', 'trưởng', 'CNTT-TT', 'Việt-Hàn']

List of improved patterns based on Underthesea

First, We've improved the RegEx pattern of Underthesea by adding a dictionary of numerical units in Vietnamese such as "lẻ", "một", "mươi", "nghìn", etc. Second, abbreviations such as the dot in the middle of the word “PGS.TS.” etc., or the license plate “43R1-999.10”. Next is the phone number, before Underthesea only used the general case "09-1934-2001", now we improved it by "+8491342001", and "008491342001". In addition, the date time is also corrected, we add cases like "12:21", "12H21P11S", and "11:11 SA". The added pattern is "units" which is the pattern we implemented while developing the old ones. In this part, we will catch cases like "lần/ngày", "1tr5", "1.05Jx19", etc. Finally, the pattern includes numeric cases like "255/255/3", "255-255-3", "255x255x3", etc. as well as the popular emoji patterns that appear in many conversational texts.

 ISBN: 978-604-80-9774-5

2.3 Word Normalization

2.3.1. Normalize Acronyms and Teencode

The use of abbreviations and teencodes is prevalent in informal written communication, often leading to ambiguity and difficulty in understanding. In this paper, we propose a method to standardize common abbreviations and teencodes in the Vietnamese language. We employ a dictionary of hundreds of commonly used abbreviations and teencodes.

Additionally, we have compiled a list of special cases, including units of measurement and currency symbols, to provide more accurate replacements for specific abbreviations. It converts the current token into its non-abbreviated form based on the rules defined in the dictionary and special cases list. If the preceding token is a numerical value and the current token is present in the list of special cases, the function returns the corresponding full form from the special cases list (e.g: "Bn có 100k ko?" => "Bạn có 100 nghìn không?" instead of "Bạn có 100 không không?"). Subsequently, if the current token is found in the dictionary, the function returns its non-abbreviated version. The proposed method enhances the clarity and coherence of informal written communication in Vietnamese by promoting standardized and easily understandable expressions. Simultaneously, reducing the number of tokens in the vocabulary to improve the efficiency of representing Vietnamese in the group's Vietnamese Embedding.

Here are some examples of the Normalize Acronyms and Teencode functions:

- *"Bn có 100k ko ?" -> "Bạn có 100 nghìn không ?"*
- *"PGS.TS. Nguyễn Văn A" -> "Phó Giáo sư Tiến Sĩ Nguyễn Văn A"*

2.3.2. Normalize Number

We have developed the Normalize Number module to preprocess the input data by performing a series of normalization steps:

- Removing the comma as a thousand separator in numbers. E.g.: *"1,000" -> "1000"*
- Standardizing numeric expressions and measurement units. E.g.: *"1km" -> "1 km" ; "1km/h" -> "1 km/h" ; "1km5" -> "1.5 km" ; "1 km 5" -> "1.5 km"*
- Converting numbers written in words to digits. E.g.: *"mười lăm" -> "15", "sáu trăm mốt" -> "610"*

We can simplify the input text, as there are many token representations that convey the same meaning (e.g., "1km5", "1.5km", and "1 km 5" all represent the same meaning "1.5 km"). This leads to a larger vocabulary size and difficulties in the subsequent word segmentation steps. The main goal of this function is to improve the efficiency of the machine learning model training. By reducing the number of tokens, the model will have fewer vocabulary items to learn, which can potentially lead to faster training times and improved performance. Overall, our text-to-number conversion function will play a necessary role in reducing the complexity of the text data.

2.3.3. Normalize Unikey

For Vietnamese typing, the users traditionally rely on specific software such as Unikey or Vietkey to input Vietnamese characters and diacritics. These software tools allow

users to type Vietnamese diacritics using special key combinations like "f," "j," "r," "s," and "x." However, in casual online communication, like chat messages or social media comments, some users may unintentionally forget to enable Unikey or intentionally choose not to use it, leading to the creation of words without diacritics. For instance, the words "trách" and "cô" might be typed as "trachs" and "coo" respectively, resulting in text that lacks proper meaning and context.

When such text is used as input for language models, it poses a challenge as the absence of diacritics can make it difficult for the model to produce accurate and meaningful results. To address this issue, our team has developed a function to standardize these words back to their correct form with Unikey diacritics. By converting words like "trachs" back to "trách" and "coo" back to "cô," we aim to ensure that the model generates the best possible outcomes.

This normalization function plays a crucial role in improving the accuracy and relevance of language models when dealing with informal Vietnamese text. It helps ensure that the model understands the intended meaning and context of the input, allowing it to provide more reliable and useful responses. With this enhancement, our language model can better assist users in understanding and generating content in Vietnamese, particularly in casual online conversations where diacritics are often omitted or forgotten.

Here are some examples for the Normalize Unikey module:

- *"Xuwr lys ngoon nguwx tuw nhieen" -> "Xử lý ngôn ngữ tự nhiên"*
- *"Tieengs Vieetj trong tris tueej nhaan taoj" -> "Tiếng Việt trong trí tuệ nhân tạo"*

2.3.4. Tone Mark Correction

One of the biggest differences between Vietnamese and English is the tone of words, which is also part of what makes Vietnamese more difficult in both spoken and written languages. Punctuation marks in Vietnamese are typed with the characters "s", "f", "j", "x", "r", "x", because these marks are written with letters of the Latin alphabet. Therefore, it is easy to type in the wrong positions of marks in words. When the sign of a word is represented in the wrong place, it can create another token of the word, leading to the case where two words with the same sign but in the wrong place will be represented as two separate tokens. With training data taken from a variety of sources including official documents and everyday communication, the data will inevitably be disturbed by words whose marks are misplaced. Tone mark correction plays the role of a filter for the text regarding specified marking rules to find and replace the wrong words.

We have applied the rule of using the "new style" diacritics in Vietnamese Romanized script. For example, "hoá" instead of "hóa", and "lũy" instead of "luỹ". Here are some examples for the Tone marks module:

- "Tíêt học môn hóa học" -> "Tiết học môn hoá học"
- "Trừơng sau lũy tre gìa" -> "Trường sau luỹ tre già"

 ISBN: 978-604-80-9774-5

2.4 Word Segmentation

In DaNangNLP toolkit, we have built a better Word Segmentation module for the Vietnamese language. Unlike English or French, which use syllables and subwords for segmentation, Vietnamese requires word segmentation to avoid misinterpretation of the meaning of words, especially for compound words. Our main goal is to segment by considering the frequency of words or the contexts of the processing sentence and the preceding sentences. These steps are presented in Figure 2. The normalized sentences are built and are input into Word Segmentation. The algorithms commonly used for Word Segmentation in the Vietnamese language include Maximum Matching [4], CRFs [5], etc., as seen in tools like VNCoreNLP [6] and Underthesea [3]. However, a common issue with these word segmentation tools is that they still struggle with ambiguous meanings in Vietnamese sentences.

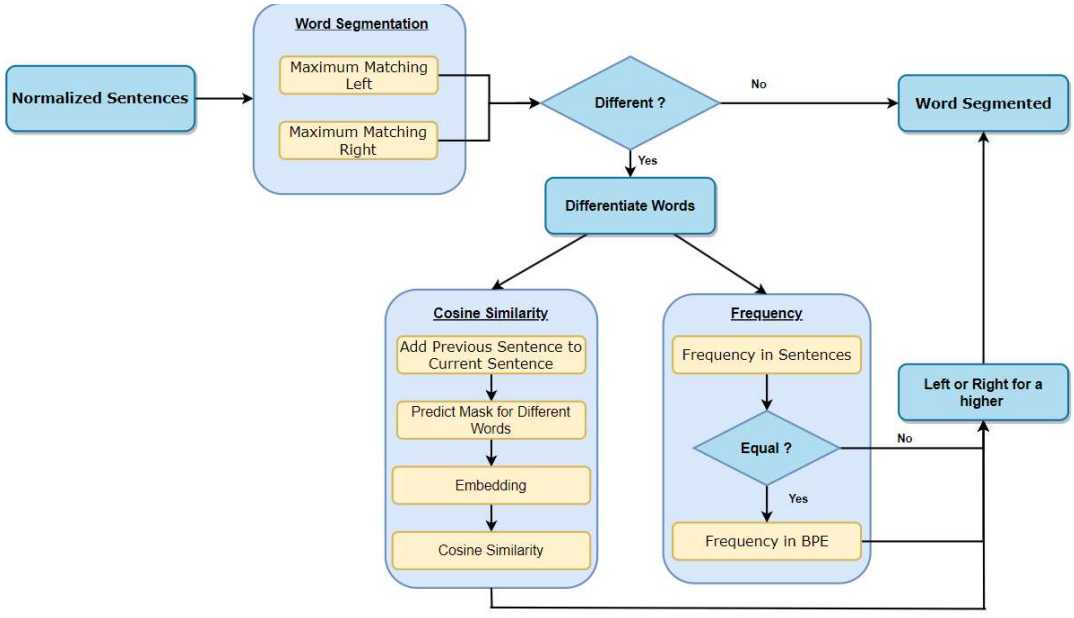

Figure 2. Diagram workflow word segmentation.

To address this problem, we have improved the traditional Maximum Matching algorithm [4] by incorporating bidirectional processing, considering both left-to-right and right-to-left approaches. We refer to the left-to-right approach as Maximum Matching Left (**MML**) and the right-to-left approach as Maximum Matching Right (**MMR**). For example, given the sentence *"Trong tiết sinh học, học sinh phải học sinh học"*, the **MML** segmentation would be *"Trong tiết sinh_học, học_sinh học"* while the **MMR** segmentation would be *"Trong tiết sinh_học, học sinh_học"*. We apply both **MML** and **MMR**, then identify the different segments, such as in this case: **MML** is *"học_sinh học"*, **MMR** is *"học sinh_học"*. In this paper, we propose two methods to select the correct phrase based on the contexts of the current sentence and the preceding sentences. Our first proposed method is designed by comparing the frequency (i.e. counting) of the occurrence of these phrases (i.e., *học_sinh* and *sinh_học*) in the current sentences as well as the two preceding, two subsequent sentences. If their frequencies

are the same, we further compare their frequencies in the broader context of the Vietnamese language such as Vietnamese PBE [7], and choose the segment with the highest frequency.

In the second method, we first put the <mask> token for the ambiguous words that have multiple segmented ways. We then use the PhoBERT model [8] to predict the masked token in the context of the previous sentence and the current different phrases between **MML** and **MMR**. In the above example, *"học_sinh học"* and *"học sinh_học"* are segmented by **MML** and **MMR**, respectively. First, it will predict the word ***"học"***. Then the word ***"học"*** will be added to the current sentence, and it will continue to predict the next word, "**thực_hành**", and add it to the sentence. At the end of the prediction process, there will be a new sentence *"Trong tiết sinh_học, học_sinh phải học thực_hành"* with two prediction words "***học thực_hành***". Next, we will take the cosine similarity from the PhoBERT [8] embedding between the predicted sentence embedding with the embedding of two MML and MMR sentences. Thereafter, we choose the higher cosine similarity score to determine the more suitable segmented phrase from **MML** or **MMR**.

Table 1. Example of mask prediction process from different words.

Trong tiết sinh_học, học sinh phải học sinh học	**Predicted word**
Trong tiết sinh_học, học sinh phải **<mask>**.	***học***
Trong tiết sinh_học, học sinh phải ***học*** **<mask>**.	***thực hành***
Trong tiết sinh_học, học sinh phải ***học thực hành***.	-

In summary, our goal is to develop a word segmentation method for the Vietnamese language that addresses the shortcomings of previous models and enhances the word segmentation capability.

Table 2. The evaluation performance on DaNangNLP evaluating datasets.

Dataset	**DaNangNLP_frequency**	**DaNangNLP_embedding**
Vitk evaluating wordseg	96.2%	96.8%
DaNangNLP evaluating_wordseg	97.1%	98%

3 Experimental Results

In order to evaluate the effectiveness, and performance of the proposed pipeline, i.e., DaNangNLP - A Vietnamese Natural Language Processing - Word preprocessing and Segmentation. We used two separate evaluation datasets for our research. The first dataset consists of *22,500* sentences and has been labeled with word segmentation [9]. This dataset was labeled using Vitk - a Vietnamese Text Processing Toolkit [10]. We reviewed the dataset and good quality, but there are still quite a few incorrect word

segmentations. Therefore, we decided to build a second valuable dataset - DaNangNLP_evaluating_wordseg. We manually labeled this dataset with *1,200* sentences covering *14* domains such as life, technology, science, sports, and more. From these two datasets, we evaluated the accuracy of our module and obtained results of *96.2%* accuracy on the Vitk_evaluating dataset and *97.1%* accuracy on the DaNangNLP_evaluating dataset for the DaNangNLP_frequency version. The results are shown in Table 1.

We continue to compare the effectiveness of our DaNangNLP word segmentation module with previously successful Vietnamese language processing modules such as Underthesea [3], PyVi [12], and Viettel API [11] on Vietnamese complex sentences as shown in Table 2. As a result, the DaNangNLP word segmentation exhibits more robust results in handling tricky examples in Vietnamese compared to previous Vietnamese language processing modules. We also develop the demo website for DaNangNLP toolkit for each function as shown in Figure 3. The demo of the other functions can be found via the following link:
https://drive.google.com/file/d/1rhSWT7dLSFQg-3N2KO5XtW0IRXqIKu_1/view

Table 3. Comparison of the effectiveness of DaNangNLP with existing Vietnamese language processing libraries.

Sentences 1	**Trong tiết sinh học, học sinh họcsinh học rất tốt**
DaNangNLP	Trong tiết sinh học, **học sinh** học **sinh học** rất tốt
Viettel	Trong tiết sinh học, học sinh học sinh học rất tốt
Underthesea	Trong tiết sinh học, học sinh học sinh học rất tốt
PyVi	Trong tiết sinh học, học sinh học sinh_học rất tốt
Sentences 2	**Người dùng phải chứng thực tại các điểm dịch vụ**
DaNangNLP	Người dùng phải **chứng thực** tại các điểm dịch_vụ
Viettel	Người dùng phải **chứng thực** tại các điểm dịch_vụ
Underthesea	Người dùng phải chứng thực tại các điểm dịch_vụ
PyVi	Người dùng phải chứng thực tại các điểm dịch_vụ
Sentences 3	**Không thu tiền ôn tập trong thời gian học chính khoá**
DaNangNLP	Không thu tiền ôn_tập trong thời_gian học **chính_khoá**
Viettel	Không thu tiền ôn_tập trong thời_gian học chính khoá
Underthesea	Không thu tiền ôn_tập trong thời_gian học_chính khoá
PyVi	Không thu tiền ôn_tập trong thời_gian học **chính_khoá**

Figure 3. DaNangNLP demo for work segmentation.

4 Conclusions

In this study, we have proposed DaNangNLP toolkit, a robust text preprocessing pipeline for the Vietnamese language to enhance the practical performance of Vietnamese Tokenizer and word segmentation. The preprocessing steps in DaNangNLP pipeline have addressed typical and necessary steps for many other word-based Vietnamese embedding models to further improve their performance as well as many language tasks. Our toolkit incorporates state-of-the-art tokenization techniques optimized for the Vietnamese language, enabling precise word segmentation and improved language understanding in many complicated sentences. Throughout the evaluation, DaNangNLP toolkit can effectively handle the many complex cases of Vietnamese texts to produce cleaner and well-processed tokens. Therefore, our tokenizer facilitated better model robustness, leading to superior results across various NLP tasks. In the future, we will adopt the DaNangNLP toolkit in finetuning the language models for Vietnamese language tasks and applications.

References

1. Eisenstein, Jacob. Introduction to natural language processing. MIT press, 2019.
2. Regular expressions. https://regexr.com/ accessed in 2024.
3. Undertheseа. “Undertheseа - Vietnamese NLP Toolkit” , https://github.com/undertheseanlp/underthesea, last accessed in 2024.
4. Wong, Pak-kwong, and Chorkin Chan. "Chinese word segmentation based on maximum matching and word binding force." COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics. 1996.
5. Yu, Bengong, and Zhaodi Fan. "A comprehensive review of conditional random fields: variants, hybrids and applications." Artificial Intelligence Review 53, no. 6 (2020): 4289-4333.
6. Vu, Thanh, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. "VnCoreNLP: A Vietnamese natural language processing toolkit." arXiv preprint arXiv:1801.01331 (2018).
7. Sennrich, Rico, Barry Haddow, and Alexandra Birch. "Neural machine translation of rare words with subword units." arXiv preprint arXiv:1508.07909 (2015).

8. Dat Quoc Nguyen, Anh Tuan Nguyen. "*PhoBERT: Pre-trained language models for Vietnamese" arXiv:2003.00744v3 [cs.CL] 5 Oct 2020.*
9. Lahitani, Alfirna Rizqi, Adhistya Erna Permanasari, and Noor Akhmad Setiawan. "Cosine similarity to determine similarity measure: Study case in online essay assessment." In 2016 4th International Conference on Cyber and IT Service Management, pp. 1-6. IEEE, 2016.
10. Vitk. https://github.com/phuonglh/vn.vitk accessed in 2024
11. Viettel natural language processing. viettelgroup.ai/service/nlp, last accessed in 2024.
12. PyVi. https://github.com/trungtv/pyvi, last accessed in 2024.