Statistics and Frontend Improvements for Voice of Southern Speech Synthesis

Dang-Tam T. Le
Advance Program in Computer Science
Faculty of Information Technology
VNU-HCM, University of Science
ltdtam@apcs.vn

Do Tri Nhan
Advance Program in Computer Science
Faculty of Information Technology
VNU-HCM, University of Science
dtnhan@apcs.vn

***Abstract*—Text Normalization is an important step in Text-to-Speech systems, helping to filter noise and making the input to be consistent with only Vietnamese syllables. Many Text-To-Speech (TTS) models for Vietnamese are developed. This report aim to analyze the performance of Voice of Southern TTS system and take an overview about Vietnamese TTS. Some statistics and improvements for Frontend of VOS are also given based on the popular syllables nowadays. A python library called ViNorm is also shared for future Vietnamese text-to-speech research.**

I. Introduction

The main task of the front-end of the TTS system is to standardize the text for the back-end system, the input is the raw text, we need to decide how to verbalize non-standard words, convert numbers, abbreviations, and words that cannot be pronounced into syllables, including dots, commas. [1].

Every language needs different normalization processing methods because this problem is language-dependent [2]. It is impossible to build a completely normalized Text normalization because the language is ambiguous and evolves over time [3].

A. Voice of Southern and other TTS systems in Vietnam

1) Voice of Southern System: Voice of Southern (VOS) is a TTS system developed by AILab from VNU-HCM University of Science. The first version of VOS (1.0) was released in 2009, followed by the 2.0 and 3.0 version in 2010 and 2012 respectively. The latest update of VOS is in 2017 with performance and quality improvement. The current VOS support instant response ($\frac{1}{15}$ realtime) with three different Vietnamese accents: Southern, Central and Northern. [4]

2) Other TTS systems: We divide list of TTS systems into two categories: one that supports Vietnamese and counterparts.

TTS system which does not support Vietnamese:

- Vocalware: Vocalware is a cloud based API for integrating real time Text-To-Speech into online web sites and mobile applications. Over 100 TTS voices in over 20 languages, offer APIs for multiple platforms, with demo Oddcast [5]
- Nuance TTS with 119 voices, 53 languages available to support, 25 years of Nuance TTS expertise [6]
- NaturalReader is a professional text to speech program that converts any written text into spoken words. It is a downloadable text-to-speech software for personal use, can read any text such as Microsoft Word files, webpages, PDF files, and E-mails, 74 TTS voices. [7]
- Microsoft Sam TTS Generator is an online interface for part of Microsoft Speech API 4.0 which was released in 1998. [8]
- Waston - IBM cloud: understands text and natural language to generate synthesized audio output complete with appropriate cadence and intonation, available in 13 voices across 7 languages [9]
- iSpeech : 27 languages, 3 reading speeds [10]
- Amazon Polly, uses advanced deep learning technologies to synthesize speech that sounds like a human voice, with 29 languages [11]

TTS system which supports Vietnamese:

- Text-to-Speech on Google Cloud, it converts text into human-like speech in more than 100 voices across 20+ languages and variants. It applies groundbreaking research in speech synthesis (WaveNet) and Google's powerful neural networks to deliver high-fidelity audio, with Vietnamese, there are 4 types of voice. [12]
- Responsivevoice is one of the most popular html5 text-to-speech API, support 51 languages. [13]
- VBee is a solution development, data digitalization and artificial intelligence leaded by Dr. Nguyen Thi Thu Trang. VBee covered 7 Vietnames accents. [14]
- FPT.AI Speech: Based on Amazon's cloud, supporting Nothern accent(male and female voices), Central (only male) and Southern (only female). [15]
- Text2speech of VTCC developed by Viettel which supports Northern accent (2 female voices and 1 male voice), Central voices are not supported and Southern (1 male and female voices). [16]
- eSpeak is a compact open source software speech synthesizer based in Vie-HTS project (Vietnamese Human-based Text-to-Speech) [17]
- Text2Voice developed by FIBO Tecchnology Company which allow voice's speed and automatic innotation adding [18]
- Vnspeak TTS by Le Anh Tuan - Vietnamese Multi-platform Text-to-Speech Engine, supports multiple plat-

	Number	VOS	VBEE	FPT.AI	Google	Responsive
Normal text	5	100%	100%	100%	100%	100%
Time-Date	15	73.30%	33,3%	53.30%	80%	80%
Acronyms	12	41.6%	66.60%	83.30%	75%	75%
Proper Noun	5	20%	60%	40%	100%	100%
Address	5	20%	40%	40%	40%	40%
Measurement	10	70%	100%	60.00%	70%	70%
Teen code	5	20%	20%	40%	40%	20%
Upper-Lowercase	2	50%	0%	50%	100%	100%
Mathematics	6	50%	66.60%	83.30%	83.30%	83.30%
Cross language	5	20%	80%	60%	100%	100%
Special case	5	60%	40%	60%	60%	60%
Total	**75**	**52%**	**59%**	**61%**	**73%**	**72%**

Table I
OVERALL RESULT

forms, such as Windows 32 bit, Windows 64 bit, Linux and other embedded systems. [19]

In this paper, we conducted the survey about the text normalization of the current Vietnamese TTS systems to get an overall look at section 2. Some of the major improvements to the VOS Frontend are described in section 3 and statistics about Vietnamese syllables usage in current online newspapers in section 4. Section 5 and 6 are the results obtained based on test cases and suggest some improvement solutions in the future.

II. SURVEY OF VIETNAMESE SPEECH SYNTHESIS

We conducted experiments with the TTS systems which had proved to have significant result on basic TTS tasks. Our surveys and comparisons are only intended to give an overview of current speech synthesis systems for Vietnamese, and are only valid at the time of the survey because those systems are still updated and maintained. We choose 5 TTS systems:

TTS system that support Vietnamese

- Google TTS WaveNet [12]
- Responsivevoice [13]

TTS system that developed only for Vietnamese

- VOS [4]
- VBee [14]
- FPT.AI [15]

A. Test cases

The test cases we provide aim to test the performance of the VOS systems. It consist of many sentences which are used on regular basis. We categorize them into 9 semiotic classes [20] [21]:

1) Normal text : Normal sentences which contains only letters.
2) Addresses: Addresses in Vietnam. This is a challenging task because of the inconsistent in the way to write and read an address. Many cases contain multiple numbers mixed in letter characters.
3) Unit of measurement: This is a basic requirement of an TTS system because unit of measurement is widely used on variety kinds of document. There are many units, some easily leading to confusion with common words or between them. Units can be grouped together to create new units, resulting in the complexity of separation and identification.
4) Time-Date: There are many ways to represent time, which leads to confusion with mathematical patterns. Time and date come in many forms, the most typical of which is "From-To".
5) Acronyms: The main challenge with acronyms is whether the TTS system should pronounce the full word that acronyms stand for or pronounce letter by letter. We analyze the performance of those systems on this test case based on the context of the document.
6) Teencode: Teencode is words that used in social network and messenger applications, it contains acronyms combined with special characters. This is very difficult test cases, just like the mis-spelling correction problem, however, not a requirement for every TTS system to implement.
7) Upper-Lowercase recognizing: Some special words have different pronunciation when written in uppercase or lowercase, especially unit of measurements.
8) Proper Noun: Common English proper nouns.
9) Mathematics: Mathematics formulas.
10) Cross language: Document have some English words. These words is from the dictionary and name of person, place, etc.

B. Criteria

MOS is the commonly used measure to evaluate a speech synthesis system [22], but in this survey, we only measure at the frontend level, we only focus on the text normalization part, to check the system be able to detect and handle special cases properly.

- Preserving information, TTS system need to read most of words in sentences correctly.
- Fluency, the system need to read sentences regular speed and pause when encountering commas and periods.
- Make the listener easier to understand, they don't try to guess the meaning of sentences leads to wrong understanding.
- Contextual, sustainable with the plurality of words, result need to match the context of documents.

C. Results

1) Overall Results: Overall result could be seen in Table I

2) Detail Results: For the detail test cases and result, please see at: Test cases and detail results

3) Comparison: From test cases provided, we compare advantage and foible of each system. Google TTS and Responsive is good at multilingual handling but not natural. FPT.AI has a smooth and fluently voice but limited with some special cases. VBEE and VOS have many similarities, just different in rules for special cases, and there are also many shortcomings when handling abbreviations and cases containing numbers. In most cases of English, Vietnamese TTS cannot read correctly, even skip reading English words.

III. Proposed Methods

Text Normalization of Vietnamese Speech Synthesis today is still building grammars by hand instead of using automatic inference from large corpora because it has been the lack of annotated data. [23] To standardize text into readable words, the TTS system process through two steps, Rule-based and Dictionary-Checking.

A. Rules with Regular Expression

By using the Regular Expression to catch patterns that need normalization, which appears frequently in the language, containing numbers and characters, then replaced by syllables found in the dictionary. The output of this step is the paragraph without any digital characters.

Compared to the old VOS-frontend system, we do not lower the entire input before processing, thus we can handle cases with different pronunciation for uppercase and lowercase of the same words, and create a premise for backend processing when the capitalized words will be emphasized more through speech synthesis step.

We propose a new set of rules that are more systematic and general, scalable for future works. Based on the different contextual characteristics, rules are divided into four main categories to handle cases need standardization, including: Special case, Timedate, Address and Mathematical. The patterns in each rule set will be proceeded one by one, browse through the entire text to match text need normalization, then return the corresponding normalized string

1) Special cases: These are the rules to capture specific cases that are out of context, with specific formats, not to be confused in different contexts, including Phone number, Football, Website, Email, etc.

- For phone numbers, the identifying feature is a sequence of numbers starting with 0 or a plus sign, the number of digits from 10 to 14 digits, with only the characters [-,., Space] in between. Phone numbers in each country will have a different way of writing, and in fact there will be different formats, so the three types of phone number rules listed are representative. 1 rule in accordance with international standards, 1 rule in the USA and 1 rule in the way of writing in Vietnam. After the phone number is matched, the [-,., Space] marks are omitted, the plus sign and the numbers are converted into syllable that represents the character. We also have a pattern for capturing hotline numbers.
- Website: patterns have identifiable characteristics that must have prefix is https, www, ftp or Suffix is popular domains. In order to handle matching matching, each each letter and character is pronounced, preferably correct over fluency, don't read letter to sound like old VOS version. The word "com" reads the whole word to make it more natural.
- Email: Use common email pattern, then matching gets Spell each letter and symbol by English letter, "@" is read as symbol "a còng", Dot and Slash: is sounded: "chấm" and "phẩy". If contains "gmail.com" then replace.
- Sport: includes specific formats such as lineups, scores. With the matching, Hypen-minus and Dot will not be spelled.

2) Time-date: are rules for capturing phrases that show the date and time element.

- With the time indicating hours, minutes, and seconds of the day, we have ways to identify such as Signal of time as "h, g", suffixed by AM / PM signs, rules to catch cases of time. need to ensure The validity of time, the case of invalid time, the system will remain the same for later processing. If "-" followed by captured regex will be read as "đến". Finally, "giờ" is added to suitable places in the replaced string.
- With the times shown in the date include types such as Typical pattern of the date month year DD / MM / YYYY, Contain roman numerals when displaying the quarter of the year, or Prefix is a word of time: "Ngày, sáng, trưa, chiều, tối, đêm, hôm, etc". In addition, the timedate rules also capture and handle cases "FROM-TO", cách đơn giản nhất là sau khi bắt được một cụm Timedate, check phía sau có kí tự "$„ khng, ccnpatternnonakhng. Beforestandardization, thematch$

3) Mathematical: to capture cases containing Normal number, Floating Point Number, Roman numerals, mathematical expression, or Unit of measurement.

- With normal number, we implement a function to convert numbers to letters. However, the normal number not only consists of successive digits, but also has a way of writing that splitting them into groups of 3 numbers, separated by commas, dot or space. This style of writing leads to confusion with floating point numbers, for example, when the text contains "2,045", we don't know whether it is exactly two thousand and forty five or two point forty five. For strings longer than 15 characters, each character will be converted individually, and if the normalized text of number is too large, the string will be separated by commas to read more fluently. If there is an "- +" in front of the string, it will be replaced with "cộng, "trừ", except in the case where the number stands at the beginning of a sentence, "+ -" is treated as a bullet symbols.
- For Floating point numbers, in addition to standard Vietnamese writing, consider Floating point as a comma, in fact the dot can also be a Floating point. So we categorized into two types with floating point: dot or coma, Floating point is replaced by "phẩy", integer part is read as normal number and discard frontier zero, with fractional part, frontier zero is replaced with "không" and the following numbers read as normal number.
- Unit of measurement are terms that refer to quantities, percentage or currency, followed by numbers, which can be an Alphabetic or a symbol. Because the system does not lower the input text, it can correctly standardize for upper and lowercase units such as Mbps and MBps, Units dictionary will distinguish upper and lower case units. Most units are SI units with metric prefixes, but we only

take common SI units for direct normalization, building a strict system that can capture the entire SI unit can lead to confusion with other cases, such as the case "12C", we are not sure whether it is "Coulomb" or an address. Besides the usual units, the patterns also need to capture Unit/Unit formats, matching strings will be checked in the unit dictionary for mapping out readable words, if the matching is not in List of Units then we return the origin matching, keep for later processing. When it is sure that both sides of the slash are units, the "/" is replaced by "trên", this will avoid confusion in the case "nam/nữ". Unit of measurement is also captured as a FROM-TO pattern, there are two common types which are "10-20 km/h" and "10 km/h - 20 km/h", regular expressions make sure not to be confused with cases where "-" is read as "minus". The values are assumed to be positive numbers, to avoid cases too strictly such as "-20°C - -10°C".

- For Roman numerals, to ensure accuracy does not fail in capturing, compare to the old VOS, we just consider uppercase is able to be Roman numerals and alphabet of Roman is [X, I, V] . [L, C, D, M] are also Roman characters, but they rarely appear in the text, sometimes even leading to confusion with acronyms. The Roman numerals also has a FROM-TO structure, such as "XVI-XXI", but it is easier to handle and more consistent than Unit of measurement. The matching is checked for correctness by converting it into a decimal number, then converting the result back into roman form to compare with the original matching, then the matching is converted to normalized text.

4) Address-Code: Address - Code are rules for capturing phrases about addresses, locations, codes and all phrases containing numbers, after this step all standardized text no longer contains numbers.

- The Political Division contains the terms for the administrative region abbreviated, The captured phrase will be mapped with the original phrase.
- Class, Office consists of alphanumeric clusters with prefix is "đường|số|số nhà|nhà|địa chỉ|tọa lạc|xã|thôn|ấp|khu phố|căn hộ|cư xá|Đ/c" or "phòng|lớp|đơn vị". The "/" in matching string is replaced by "xuyệt", the phrase and the number will be read each character.
- Codenumber: will Match All remain cases contain numbers, the matching sequence will trim punctuation at the front and back of the string. The captured phrases will be separated into each alphanumeric substring. For each substring, if it is a fully capitalized letter sequence, it will be transcribed accordingly, if it contains lowercase letter, the system will keep that letter clusters and add space separate from other clusters. If the substring is numeric, if the length of continuous number string is greater than 4, each number will be transcribed, otherwise it will read the whole number as a normal number. Case includes special character or symbol will be mapped with corresponding phonetics, but with "-" is not spelled.

B. Dictionary Checking

After running through the rules sets, string just contains letters, space and special character. This string will be splitted into segments separated by spaces, each token containing no space and no special characters before and after of the string. The system run over each token to validate if it is readable by checking it in Dictionary Vietnamese syllable Includes 7698 syllables, which are supported by the current VOS-backend. If the token does not exist, we will look it up in the mapping dictionary instead, search and replace with the corresponding word. The check order in the mapping dictionary is based on the popularity of words between dictionaries to avoid conflicts (a token is in two Mapping dictionaries).

1) Abbreviations: This mapping dictionary includes 3 different mapping ways, firstly, Acronyms such as "NSƯT, GDĐT", we have to normalize to its original form. The second type is Initialisms, including words like STEM and UNESCO, so we have to transcribe the reading for it. The last type is the acronyms that need spell each letter such as PNJ, FPT, AFF, we do not handle it in this step and consider as unknown to limit misunderstandings when not sure. One difference is that the acronym list can include characters such as "NQ/TW", which makes the mapping process more accurate and sure-footed. With dictionary mapping abbreviations, uppercase and

Figure 1. Common abbreviations in Vietnamese

lowercase tokens are often referred as two separate objects, with different contexts and probabilities, an acronym can have more than one meaning. Acronyms that appear less frequently are filtered out to limit misunderstandings

2) Teen Code - Slang - Lingo: We updated this mapping dictionary by adding more than 300 slangs like "H'Hen Nie, Ea H'leo", added some Lingo that not appear in Popular Dictionary, words that backend doesn't support like "Đắk Lắk, Pleiku". It also handles inconsistency in writing of the same word, such as "thuỷ - thủy" or "tuỳ - tùy". Loanwords like "oxy, axit" and common misspellings are also updated.

3) Special Symbols: After browsing through the dictionaries, if the token does not belong to any Mapping Dictionaries, we continue to split the token into smaller substring separated by symbols and special characters. The system shows the corresponding reading for each symbols, non-stop punctuation such as brackets, quotes will be removed. The system continue checking each substring in the mapping dictionaries again, this process, (Tokenize with space first, if the token is unknown, then tokenize with special character) ensure normalization avoids errors when the string is written adjacent to each other. This way will handle both cases “GD-ĐT” and “ê-kíp”, hyphen in the first case will be omitted to "GDĐT", the second case will be replaced with space to "ê kíp", and many similar cases with symbolic problems are also solved. If the token is still not in any mapping dictionary, we treat it as unknown word, if it is uppercase, spell each character as English letter, if it is lowercase and containing vowels, we will leave the backend to handle by letter to sound, if it is not contains the vowel, spell each character as Vietnamese letter.

4) Punctuations: The final step is to standardize the output text, including removing duplicate white spaces, handling punctuations including removing no voice marks like “()[]", and replace all punctuation marks with two punctuation symbols, comma and dot, which represent as the sound unit.

IV. Vietnamese Syllables Statistics

In order to update the Vietnamese syllables used today, we proceeded to build a Text News Corpus with sources of 9 online newspapers, crawled from 7/2018. The Corpus is divided at the sentence level, consisting of a total of 6,308,173 sentences, the number of non-standard sentences is 3,740,507 sentences, accounting for more than 59.29 %, the rest are sentences that need not be processed, containing Vietnamese-only vocabs.

dantri	danviet	nld	thanhnien	tto
653545	386444	103218	634974	177287
tuoitre	vnexpress	vnn	zingnews	
167162	157202	481566	979109	

Table II
Number of sentences from popular Vietnamese newspapers

We continue to word-tokenize every sentence, totaling 10.88 million tokens.The tokens will be classified into groups such as Special Case, Time-date, Math and Number, English, Slang, Teencode, Acronyms, Initialism, Proper Noun, etc.

Some notable points from the statistics on News Corpus are: special cases only account for 0.2%, most of the Timedate cases are the publication date of the news, English accounts for 40 percent, and most of the abbreviations are all in uppercase.

V. Experiment

A. Programming Language and Library

To select the language for the text normalization problem, we considered choosing between C ++, Python and Perl. A comparison of common programming languages used in bioinformatics, also requires performing different computing tasks on the sequences, indicating that Python and Perl are often called script languages and suitable for web scripting, parsing, C and C ++ are fully compiled languages, suitable for system-intensive tasks [24]. We decided to choose C ++ because it is suitable for Cross-Platform Deployment, has a Fast running

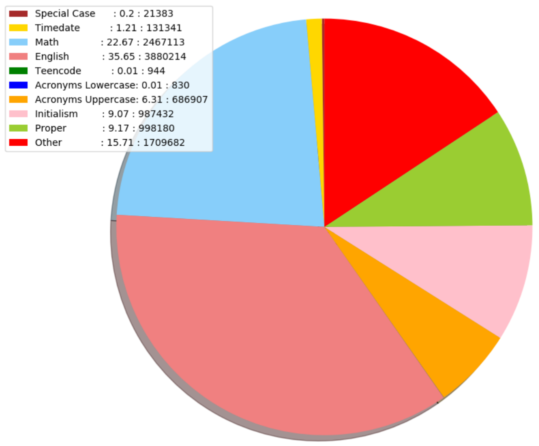

Figure 2. Statistics on Tokens need to be standardized in Vietnamese

time, Use less memory than Perl and is compatible with the current VOS backend. Some famous frameworks for text to speech system also use C and C ++ such as Festival Speech Synthesis System of University of Edinburgh, Flite of CMU, Hts-engine use for Jtalk, Sinsy, and eSpeak is also written in C and C ++.

One of problems when using C ++ is to handle Vietnamese Unicode, we use ICU4C library version 64.2, an International Components for Unicode. This library is opensource,Well-documented, robust and reliable. The ICU provides basic regular expression operators and especially Case Insensitive Matching, which helps the regular expression to capture both uppercase and lowercase letters, preserving the properties of the input text, which will be beneficial for handling backend, helping voice more natural, change the overall intonation like stress on capitalized words.

We provide a python package on Ubuntu 18.04 that can be installed at the Python Package Index called ViNorm.

B. Testing

From the data collected as mentioned above, we extracted 100 tricky need-normalized cases to use as the baseline for improvement, 500 random cases in practical contexts for test our proposal. These test cases do not include normal sentences, foreign words and proper nouns. With the 100 case test suite, we improved the frontend of VOS from 60% to 97%. With the 500 test cases, our method achieved 96%, but there are still many cases that only temporarily accept, retain the sentence meaning but lead to unnatural. Some cases are wrong when mapping acronyms due to its plurality, such as BTC, we can read as "Ban tổ chức", "Bộ tài chính", or it can also be a stock symbol.

VI. Conclusion

A. Performance of VOS

VOS system has natural and fluent voice. However, when dealing with informal document, VOS has few minor errors result in unnatural results. This could be explained by the rapid change in the informal language, especially ones used in Internet. In this report, we improve the numeric processing modules, the processing special character module to produce the result to match the context of document. Larger dictionaries are used to cover more words and implementing module to recognize words which have different pronunciation in uppercase and lowercase. Updates to the regular expression step and the expansion of mapping dictionaries have helped VOS solve many special cases, especially for tokens that contain both numbers and letters.

B. Future Improvements

There are still many issues that text normalization needs to improve, as the statistics mentioned above show that English and the Proper Nouns make up nearly 40% of the tokens to handle. To handle multilingual words, there are several approaches such as using the International Phonetic Alphabet to transcribe Vietnamese [25], or improving the backend to handle out-of-domain and complex words like neural network models [26].

The current improvements are mainly based on the regular expression and mapping by checking lexicon, there are still some unintended matches. One of the painful things when working with Text Normalization is handling ambiguous cases. To reduce ambiguity, we need to rely on the context of the word to be standardized, but this takes a lot of effort because with an ambiguous word, we need to label different meanings for that word in each specific case, then we can put it into the statistical model for training. Currently RNN-like models for text normalization, a mix of rule-based and model-based system is state-of-the-art and gradually applied to each specific language [27] [28] [29].

There are also many issues that need to be solved when the unofficial input text such as misspellings, no accents, containing sticky phrases such as thegioididong, dienmayxanh, etc.

References

[1] R. Sproat, A. W. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards. (2001) Normalization of non-standard words.

[2] M. Chu, H. Peng, and Y. Zhao. (2009, Feb. 24) Front-end architecture for a multi-lingual text-to-speech system. US Patent 7,496,498.

[3] D. Yarowsky. (1993) Text normalization and ambiguity resolution in speech synthesis.

[4] V. Q. D. Ha, N. M. Tuan, C. X. Nam, P. M. Nhut, and V. H. Quan. (2010) Vos: the corpus-based etnamese text-to-speech system.

[5] Vocalware. Vocalware's text-to-speech. [Online]. Available: https://www.vocalware.com/index/demo

[6] N. Communications. Nuance's text-to-speech. [Online]. Available: https://www.nuance.com

[7] N. Ltd. Natural 's text-to-speech. [Online]. Available: https://www.naturalreaders.com/online/

[8] Microsoft. Online microsoft sam tts generator. [Online]. Available: https://tetyys.com/SAPI4/

[9] IBM. Waston text to speech. [Online]. Available: https://text-to-speech-demo.ng.bluemix.net

[10] iSpeech. Demo. [Online]. Available: https://www.ispeech.org

[11] A. W. Services. Amazon polly. [Online]. Available: https://aws.amazon.com/polly/

[12] Google. Demo. [Online]. Available: https://cloud.google.com/text-to-speech

[13] Responsivevoice. Responsivevoice's text-to-speech. [Online]. Available: https://responsivevoice.org/

[14] VBee. Vbee 's text-to-speech. [Online]. Available: https://vbee.vn/

[15] F. CORPORATION. Fpt.ai speech. [Online]. Available: https://fpt.ai/tts/

[16] T. Van Nguyen, B. Q. Nguyen, K. H. Phan, and H. Van Do. (2018) Development of vietnamese speech synthesis system using deep neural networks.

[17] F. S. Foundation. Vietnamese human-based text-to-speech. [Online]. Available: http://espeak.sourceforge.net/

[18] F. T. Company. Fibo text-to-speech. [Online]. Available: https://fibo.vn/voice/demo/

[19] L. T. Anh. Vnspeak tts. [Online]. Available: http://www.vnspeak.com/vnspeak-tts/

[20] S. Ritchie, R. Sproat, K. Gorman, D. van Esch, C. Schallhart, N. Bampounis, B. Brard, J. F. Mortensen, M. Holt, and E. Mahon. (2019) Unified verbalization for speech recognition & synthesis across languages.

[21] P. Taylor. (2009) Text to speech synthesis.

[22] R. C. Streijl, S. Winkler, and D. S. Hands. (2016) Mean opinion score (mos) revisited: methods and applications, limitations and alternatives.

[23] R. Sproat. (2010) Lightly supervised learning of text normalization: Russian number names.

[24] M. Fourment and M. R. Gillings. (2008) A comparison of common programming languages used in bioinformatics.

[25] T. T. L. Ly, B. T. Sơn, N. K. Duy *et al.* (2019) MỘt giẢi pháp viỆt hóa cách phát âm các tỪ vỰng tiẾng anh trong văn bẢn tiẾng viỆt.

[26] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan *et al.* (2018) Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. IEEE.

[27] B. Roark, F. Stahlberg, H. Zhang, K. Wu, K. Gorman, R. Sproat, and X. Peng. (2019) Neural models of text normalization for speech applications.

[28] R. Sproat and N. Jaitly. (2016) Rnn approaches to text normalization: A challenge.

[29] S. Yolchuyeva, G. Németh, and B. Gyires-Tóth. (2018) Text normalization with convolutional neural networks.