Journal of Computer Science and Cybernetics, V.34, N.4 (2018), 349–363
DOI 10.15625/1813-9663/34/4/13172

DEVELOPMENT OF VIETNAMESE SPEECH SYNTHESIS SYSTEM USING DEEP NEURAL NETWORKS

NGUYEN VAN THINH[1,a], NGUYEN QUOC BAO[1], PHAN HUY KINH[1], DO VAN HAI[2]

[1] *Viettel Cyberspace Center, Viettel Group*
[2] *Faculty of Computer Science and Engineering, Thuyloi University*
[a] *thinhnv20viettel.com.vn*

Abstract. In this paper, we present our first Vietnamese speech synthesis system based on deep neural networks. To improve the training data collected from the Internet, a cleaning method is proposed. The experimental results indicate that by using deeper architectures we can achieve better performance for the TTS than using shallow architectures such as hidden Markov model. We also present the effect of using different amounts of data to train the TTS systems. In the VLSP TTS challenge 2018, our proposed DNN-based speech synthesis system won the first place in all three subjects including naturalness, intelligibility, and MOS.

Keywords. Text-to-speech; Speech synthesis; Deep neural network; Hidden Markov model.

1. INTRODUCTION

Nowadays, with great advance in the field of artificial intelligence, many automatic systems have been built like robots, virtual assistants, autonomous cars, etc... and many are recorded surpassed human-performance. Developing a system with the ability to synthesize human-like speech from raw text becomes crucial. A Text-To-Speech system (TTS) is a computer-based system that automatically converts text into artificial human speech [1]. Note that TTS systems are different from Voice Response Systems (VRS): A VRS simply concatenates words and segments of sentences and is applicable only in situations of limited vocabulary.

Developing a natural, intelligible TTS system has long been a subject of broad interest within science community. From the day Christian Kratzenstein introduced his simple mechanical speech synthesizer [2] in 1779, with the ability to simulate five long vowels (/a/, /e/, /i/, /o/, and /u/), many innovative, efficient systems have been developed and improved ever since. Later, in 1937, the Bell Telephone Laboratory introduced VODER [3] (from Voice Operating Demonstrator). It is considered to be the first system in the world to electronically synthesize human speech by analyzing its acoustic components. Apart from synthesizing English, in 1975 MUSA was a specialized software with the ability to read Italian. A second version, released in 1978, was also able to sing Italian in an "a cappella" style [4]. There had been still no multilingual TTS system until the year of 1997, when Bell labs published their research on synthesizing multi-languages [5], which extensively used natural language processing approach.

Despite the fact that those synthesized speech quality had been improved significantly compared to their predecessor, those systems still performed much worse than human did. By late 1990s and early 2000s, the naturalness and intelligibility have been boosted by using statistical parametric speech synthesis (SPSS) such as hidden Markov model [6]. This approach had incredibly advanced the quality of synthesized speech. Recently, deep neural networks (DNNs) have been applied for TTS and achieved further improvement over the SPSS method [7] while providing great flexibility to user requirement. WaveNet [8] and DeepVoice [9] are two mounting evidences, which in many cases, sound perfectly human.

Speaking of Vietnamese, several researches have been done to tackle TTS problem for Vietnamese. By the years of 2000s, most of Vietnamese TTS systems were built by using *formant* and *concatenative synthesis* [1, 10]. Both of these two approaches have significant disadvantages: while formant synthesized speech usually is lack of naturalness and sounds creepily robotic, concatenative-based methods provides more human speech, but without smooth continuity, mostly caused by distortions and asynchronous at junction between two consecutive segments. In the work of Do and Takara [1], a TTS system named VietTTS was built based on half-syllables with the level tone information, as well as a source-filter model for speech-production and a vocal tract filter (modeled by log magnitude approximation). The speech quality was acceptable then, but still could not resolve its concatenative limitations.

By that times, SPSS had become popular with the proliferation of using HMM with significant improvement had been recorded [11, 12]. In such systems, the frequency spectrum (vocal tract), fundamental frequency (voice source), and duration of speech were modeled simultaneously by HMMs. Speech waveforms are generated from HMMs themselves based on the maximum likelihood criterion [13]. Speaking of synthesized speech quality, generally HMM outperforms other concatenative-based methods, with significant improvement in continuity and naturalness. Many TTS systems were recorded achieve good intelligibility with HMM approach like T. T. Vu et al. system [14], or Q. S. Trinh system [15]. However, since HMM bases on Markov assumption, it is not able to capture long-term dependencies, which can further improve naturalness.

Deep neural network, with the ability to address that problem of HMM, has become popular in not only speech synthesis, but also in many other context-dependent problems like Automatic Speech Recognition [16, 17]. They have proven themselves to be powerful, flexible and require less effort on data processing, compared to other traditional machine learning methods. Many TTS systems, built over DNN architectures, have shown incredible performance. Nevertheless, to the best of our knowledge, there is no published research for Vietnamese TTS system based on DNN. Within the scope of this paper, we present our first DNN-based Vietnamese TTS system, which achieves superior MOS (Mean Opinion Score)[1] of intelligibility and naturalness, compared to other Vietnamese TTS systems like MICA[2] and VAIS[3] (the results were evaluated in the International workshop on Vietnamese Language and Speech Processing - VLSP 2018[4]). Postfilter parallelization to reduce time response of the system without quality degradation will also be presented.

The rest of the paper is organized as follows. Section 2 presents our proposed TTS

[1]For MOS detail, see https://www.itu.int/rec/T-REC-P.10/en
[2]http://sontinh.mica.edu.vn/tts2
[3]https://vais.vn
[4]http://vlsp.org.vn/vlsp2018

system. Section 3 describes our experimental setup and results. Section 4 concludes the paper.

2. THE PROPOSED SYSTEM ARCHITECTURE

This section first gives an overview of our DNN-based Vietnamese TTS system. After that we will present how to implement each module in details.

2.1. System overview

Figure 1 illustrates the proposed TTS system. The input is text, the output is synthesized speech. The system consists of five main modules: text normalization, linguistic features extraction, duration model, acoustic model, waveform generation.

Text normalization is responsible for normalizing input text. In this process, the input text is converted into a form which is speakable words, for example, acronyms are transformed into word sequences or numbers are converted into words. Linguistic feature extraction is used to extract linguistic features from normalized text. Linguistic features include information about phoneme, position of phoneme in syllable, position of syllable in word, position of word in phrase, position of phrase in sentence, tone, part of speech tags of each word, number of phoneme in syllable, number of syllable in word, etc... Duration model is used to estimate timestamps of each phoneme. In this paper, this model is realized by a DNN. Acoustic model is used to generate acoustic features such as F0, spectral envelope which are corresponding to linguistic features [7]. In this paper, a DNN is also used to implement this mapping. Waveform generation (also called as Vocoder [18]) converts acoustic features into speech signal.

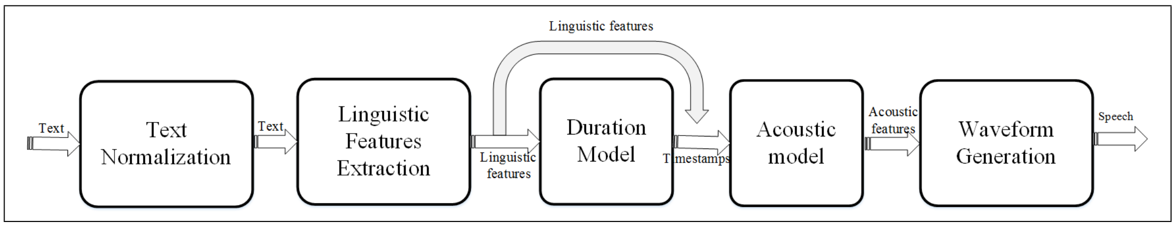

Figure 1. System overview of the proposed TTS system

Figure 2 shows in details the training and testing processes of the TTS system. In the training process, there are two phases. The first phase is to train the duration model and the second phase is to train the acoustic model. The corpus used to train both the acoustic model and the duration model is a set of audio files with corresponding transcription. In the first phase, duration model is trained by the following process. Linguistic features are first extracted by the linguistic feature extraction module, these features are then put into a label aligner which had been modeled by hidden Markov model (HMM), to estimate the initial timestamps for each phoneme. Linguistic features with timestamps of phonemes and audio files are then used to train the duration model. After training, the duration model is used to generate new and better timestamps for each phoneme. In the second phase, a vocoder [18] is used to extract acoustic features including fundamental frequency (F0) [19], spectral envelope (SP) [20] and aperiodicities (BAP) [21] of audio files. The spectral envelope and fundamental frequency are respectively converted into Mel coefficients (MC) and logarithmic fundamental frequency (log F0) by Features Converter module. The acoustic model is trained to estimate the acoustic features given the linguistic features with the new timestamps.

In synthesis process, normalized text is used to extract linguistic features. After that, the duration model is used to estimate the timestamps of each phoneme. The linguistic features together with timestamps are then used as the input for the DNN acoustic model to generate corresponding compressed acoustic features, which include MC, log F0, BAP. The Mel coefficients are put into postfilter [22] to improve synthesized voice, then the compressed acoustic features included postfiltered mel coefficients, logarithmic fundamental frequency, aperiodicities are converted back to acoustic features SP, F0, BAP in Features Converter module. Normally, postfilter takes quite a long time to process. In Section 2.2.4, we present our approach to reduce the response time of postfilter while maintaining the quality. Finally, the acoustic features are used as the input of the vocoder to generate speech signal.

In the next section, we will describe how to implement each module in details. Some module like Features Converter, Label Aligner wont be expressed, because these modules are implemented originally from Merlin Toolkit [23] without any modification.

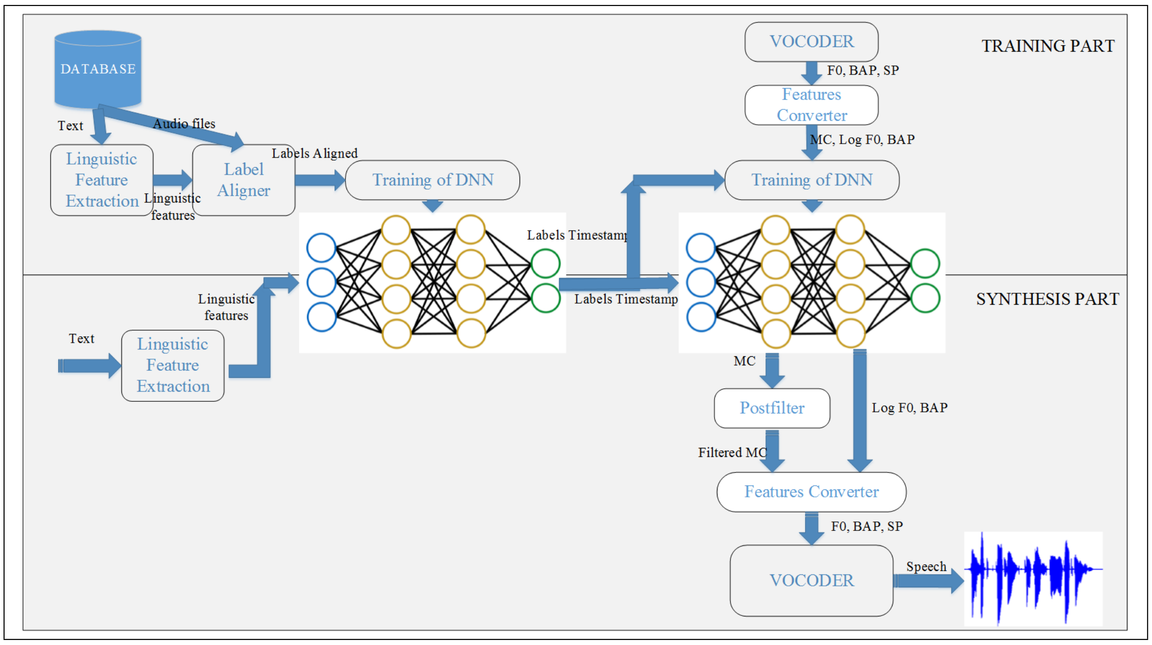

Figure 2. The training and synthesis processes of a speech synthesis system using deep neural network

2.2. Module implementation

2.2.1. Text normalization

The text normalization module is responsible for converting input text into a speakable form. For example, date-time, numbers, acronyms should be converted into words sequence.

Our text normalization procedure is conducted as follows: the input text is initially split into a sequence of elements (could be word or syllable) by whitespace. Each element is then looked up in the Vietnamese dictionary. If an element is not found, it will be converted to speakable syllable sequence by looking it up in the loanword and acronyms dictionaries. For example, *cntt* will be transformed to *cong nghe thong tin.* If it still cannot be found, we apply *regular expression* to search for available form (like sequence number, date, time). For example, *"30/6/2018"* is replaced by *"ba muoi thang sau nam hai nghin khong tram muoi tam"*. Remaining unknown elements are removed, only normalized elements and fixed misspelling words are saved.

2.2.2. Linguistic features extraction

Linguistic features are generated by extracting linguistic information from the input text including part-of-speech tag, word segmentation, text chunking, as well as phone and syllable information, and timestamps of phonemes [24, 25, 26]. Each piece of information is then encoded into a one-hot vector.

For text preprocessing, firstly, input text is transformed to a sequence of phoneme by using a phonetic dictionary for 6700 Vietnamese widely-used syllables. This sequence is then segmentated into meaningful words, each of those words is then tagged with its POS tag and Chunking tag. Since Vietnamese does not have an explicit delimiter for distinguishing words like English and many other languages (commonly the space character and almost every punctuation), word segmentation for Vietnamese is considered to be hard and sometimes, a wrong segmentation can lead to ambiguity and completely alters the meaning of the original sentence. This step also decides the quality of features set, when an accurate segmentation can further improve the accuracy of POS tagging and Chunking.

To train the DNN acoustic model, time information also needs to be added to linguistic features set, which consists of beginning time and ending time of the phoneme. This step is called forced alignment [27] from the Merlin toolkit [23], implemented by using an HMM.

Specially compared to English, Vietnamese have many discernable linguistic features, which mostly come from the tonal characteristic of Vietnamese and the difference between the phoneme set between these two languages. While Englishs phonological system is commonly[5] known as the composition of 44 phonemes [28]. According to Ben Pham & Sharynne McLeod [29], Vietnamese north dialect contains a total of 50 phonemes, in which there are 20 initial consonants, 2 initial semi-vowels, 10 final consonants, 2 final semi-vowels, 9 long singleton vowels, 4 short singleton vowels, and 3 dipthongs. In addition to that, Vietnamese has a total of 6 tones [30, 31, 32]. All those dissimilarities are presented within the labels file, along with those features described above.

After those above steps, the extracted linguistic features contain multi-level information about the input text, which includes:

- *Phoneme-level:* Current, previous and next phoneme and its position in the syllable.
- *Syllable-level:* Number of phonemes, tone of current, previous and next syllable, position of syllable within current word, phrase and sentence.

[5]Some researches claim it to be 35 phonemes [28]

- *Word-level:* POS tags of current, previous and next word; number of phonemes, syllables in current, previous and next word.
- *Phrase-level:* Number of current, previous and next words and syllables.
- *Sentence-level:* Number of words, syllables and phrases.

2.2.3. Acoustic and duration modeling

In our TTS system, both the acoustic and duration models are implemented by neural networks. Specifically, we use conventional feed-forward deep neural networks. The effect of number of hidden layers will be discussed in the experiment section.

Since deep neural networks can only handle numeric or binary values, the linguistic features need to be converted. There are many ways to convert linguistic features into numeric features, one of them is to answer the question about linguistic context e.g., what is the current phoneme? what is the next phoneme? how many phonemes in current syllable?. Compare to the Merlin DNN-based TTS system for English [23], our DNNs for Vietnamese TTS have many more input features because of the vast differences in number of phonemes and tone information. They consist of 752 inputs including 743 features derived from linguistic context and the remaining 9 features from within-phone positional information e.g., frame position within HMM state and phone, state position within phone both forward and backward, and state and phone duration [23].

2.2.4. Postfilter

Statistical averaging of parameters creates trajectories that are overly smoothed across frames in the time domain but also within a frame in the spectrum domain [33]. A postfilter technique on Mel cepstral coefficients represented in [22] was applied to statistically generated speech trajectories. Postfilter is also recorded to enhance the formant structure in speech coding [34]. Moreover, it can be used to compensate the oversmooth spectrum in speech synthesis as well [33]. Furthermore, the result in [22] showed that by combining a mixed excitation model with a postfilter, we can significantly improve the quality of synthesized speech of HMM-Based Text-to-Speech Synthesis. For those reasons, we have applied postfilter in DNN-Based Speech Synthesis and got improvement of synthesized speech quality (see more details in Section below). The cost of such enhancement is a considerable increase in system response time. In this paper, we propose a method to reduce the response time while maintaining the quality.

Because the spectra are expressed as Mel-ceptral coefficients, therefore, we use a postfilter on the Mel-cepstral coefficients as in [22] and [34]. The implementation of postfilter was introduced in HTS with the architecture presented in Figure 3. The first block, ACC-Mcep is an autocorrelation block which takes Mel-ceptral coefficients as input and outputs corresponding autocorrelation coefficients. The same ACC-Postfiltered Mcep block also calculates autocorrelation coefficients, but the given inputs of this block are Mel-coefficients and weight vector. The filtered Mel-ceptral coefficients, which are used for calculating output autocorrelation coefficients, are the result of multiplying each Mel-ceptral coefficients with

[6]http://hts.sp.nitech.ac.jp/

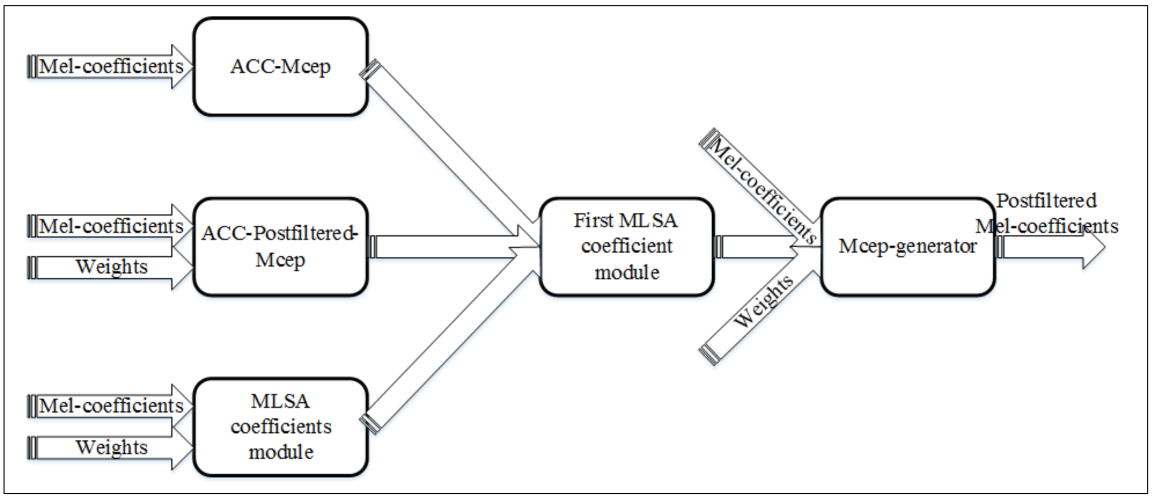

Figure 3. The architecture of postfilter (adapted from HTS[6])

corresponding weight factor in weight vector. The weight vector has 0^{th} value and 1^{th} value are 1, the other values are tunable values, which control the amount of perceptual postfiltering. In the MLSA coefficients module, the MLSA coefficients (MLSA: Mel Log Spectral Approximation [35]) is calculated from the filtered Mel-coefficients by the following function

$$\begin{cases} b(m) = c(m), \quad m = M, \\ c(m) - \alpha * b(m+1), \quad 0 \leq m < M \end{cases} \tag{1}$$

where α is a parameter to approximate the Mel scale [22].

Autocorrelations coefficients of both Mel-ceptral coefficients and filtered Mel-ceptral coefficients are used to compute 0^{th} MLSA coefficient. It was then added into 0^{th} MLSA coefficient of output vector. Finally, the postfiltered Mel-coefficients are calculated from Mel-coefficients, weight vector and MLSA coefficients in Mcep-generator module.

After calculating time response of the functions of the postfilter implementation, we have realized that both autocorrelation modules take the most of response time. The autocorrelation module takes Mel-ceptral coefficient vector c as an input

$$c(0), c(1), ..., c(M_1)$$

the corresponding autocorrelation coefficient vector is given by r

$$r(0), r(1), ..., r(M_2)$$

where M_1 is the order of cepstral coefficients and M_2 is the order of autocorrelation coefficients.

In the autocorrelation module, each r is calculated from Mel-ceptral coefficient vector c, independently with other vector. For that reason, parallelization can be applied for each cepstral coefficient (within a frame). In our proposed system, the cepstral coefficient vectors are grouped into six different groups, hence six parallel threads are also used for calculating autocorrelation vectors of each group. Applying of parallel processing archives significant improvement of time response. Time performance can also be even better by using more threads for computation (see more details of the time response comparison of those systems in Section 3.3.5).

2.2.5. Vocoder

Vocoder is a module that analyzes and synthesizes speech. In our TTS system, vocoder is used in both the training phase and the synthesis phase. In the training phase, vocoder extracts acoustic features from training audio files. In the synthesis phase, the estimated acoustic features are used as the input of vocoder to generate speech signal.

In this paper, we use a popular vocoder called WORLD [36] to analyze and synthesize speech signal. The acoustic features analyzed by WORLD vocoder include 60-dimensional vector of Mel coefficients containing spectral envelope information, 25-dimension vector of aperiodicities and logarithm of F0 [23]. In the training phase, these vectors are used as the output of the acoustic model deep neural network. In the synthesis phase, these feature vectors (predicted by the DNN acoustic model) are used as input for the vocoder to synthesize speech signal.

3. EXPERIMENTS

3.1. Corpus preparation

Corpus preparation is one of the most important processes to make a high quality speech synthesis system. To have a good training dataset, we first need to collect a large enough amount of data. The dataset then needs to be further processed to improve the data quality.

To achieve the most natural synthesized speech, we have collected around 7 hours of pre-recorded audio from an audio news web site (http://netnews.vn/bao-noi.html). However, there are several issues by using this corpus for speech synthesis for example the volume of audio is not consistent sometimes too loud or too soft, noise sometimes appears within the pauses, the acronyms and loanwords exist in the corpus, and there is no transcript at the sentence level.

To improve the quality of training data, the following procedure is proposed:

- Step 1: The long audio files are split into sentences.
- Step 2: The volume of the audio files is then normalized at the sentence level.
- Step 3: Manually clean the noise within pauses. The sentences with high background noise are removed.

Finally, we obtain a corpus with 3504 audio files that are equivalent to 6.5 hours.

3.2. Experimental setup

The corpus is divided into three subsets for training, testing and validating with 3156, 174 and 174 sentences respectively.

The 6 hidden layer feed-forward deep neural networks are used for both the duration model and acoustic model. Each hidden layer contains 1024 neurons. The WORLD vocoder is chosen to analyze and synthesize speech signal.

The input features of the acoustic model and the duration model consist of 752 features that include 743 features derived from linguistic context and 9 features from within-phone positional information. The acoustic features include 60 dimensional Mel-cepstral coefficients, 25 band aperiodicities and F0 on log scale.

We also build an HMM-based TTS system as the baseline to compare with our DNN-based TTS system. The same training data set was used as in the DNN system. Each observation vector consists of 25 Mel-cepstral coefficients, F0 on log scale and their delta and delta-delta features. The number of questions for decision tree was 743. The five-state-HMM is used for modelling the phonemes, and a multi-space probability distribution is used for modelling log F0 sequences consisting of voiced and unvoiced observation.

3.3. Experimental results

We first describle about the evaluation metrics used in TTS. After that various experiments are presented to show the advantages of our proposed system.

3.3.1. Evaluation metrics

We evaluate the quality of TTS systems based on two criteria i.e., objective and subjective results.

- Objective evaluation (lower is better): We evaluate TTS systems using 4 popular metrics [7] including:

  - MCD: Mel-cepstral distortion.

  - BAP: Distortion of band aperiodicities.

  - F0-RMSE: Root mean squared error in log F0.

  - V/UV: Voiced/unvoiced error.

- Subjective evaluation (higher is better): three metrics including naturalness, intelligibility with maximum score of 100 and Mean Opinion Score (MOS) with maximum score of 5 are used for subjective evaluation. 6 people are asked to listen to each synthesized sentence and provide their score.

3.3.2. Effect of cleaning training data

We first examine the effect of applying the training data cleaning procedure (Section 3.1). Two DNN-based TTS systems are trained without and with using the data cleaning procedure. Table 1 shows that by carefully cleaning training data, a significant improvement in synthesized speech quality is achieved both in objective and subjective evaluation. Specifically, 4 objective metrics are reduced, while 3 subjective metrics are increased.

Table 1. The objective and subjective evaluations for the two DNN-based TTS systems without and with using the training data cleaning procedure. (MCD: Mel-Cepstral Distortion; BAP: distortion of band aperiodicities; F0 RMSE: Root mean squared error in log F0; V/UV: voiced/unvoiced error)

Training data cleaning	Objective evaluation				Subjective evaluation		
	MCD (dB)	BAP (dB)	F0 RMSE (Hz)	V/UV (%)	Naturalness	Intelligibility	MOS
No (DNN1)	4.758	0.171	23.038	6.084	92.67	94.00	4.50
Yes (DNN2)	4.721	0.163	22.119	6.052	94.67	96.33	4.61

3.3.3. Effect of DNN architecture

In the previous experiments, 6-layer DNN were used for the duration and acoustic models. Now we investigate the effect of DNN architecture to the quality of the TTS system. Note that in all cases, we use training data after cleaning i.e., DNN2 in Table 1.

Table 2 shows the results given by the DNN-based TTS systems with different DNN architectures The last row is the result given by the HMM-based TTS baseline. We can see that by increasing the number of hidden layers from 1 to 6, we can improve both objective and subjective metrics. However, when more than 4 hidden layers are used, not much improvement is observed for objective evaluation except voice/unvoice error. For subjective evaluation, no improvement is achieved by using more than 5 hidden layers for the DNN models.

Table 2. The objective and subjective evaluations for the TTS systems with different DNN architectures, the last row is the result for the HMM-based TTS system. (MCD: Mel-Cepstral Distortion; BAP: distortion of band aperiodicities; F0 RMSE: Root mean squared error in log F0; V/UV: voiced/unvoiced error)

Model	Objective evaluation				Subjective evaluation		
	MCD (dB)	BAP (dB)	F0 RMSE (Hz)	V/UV (%)	Naturalness	Intelligibility	MOS
1 layer-DNN	5.104	0.173	24.158	7.097	88.33	91.67	4.31
2 layer-DNN	4.875	0.169	23.010	6.577	91.67	94.00	4.47
3 layer-DNN	4.769	0.166	22.434	6.310	92.33	94.33	4.49
4 layer-DNN	4.729	0.163	22.051	6.212	92.33	94.67	4.50
5 layer-DNN	4.724	0.163	21.969	6.141	94.67	96.33	4.67
6 layer-DNN	4.721	0.163	22.119	6.052	94.67	96.33	4.67
HMM	4.790	0.185	23.012	8.528	89.67	90.00	4.40

Comparing to the HMM-based system in the last row, the DNN-based system (6 hidden layers) has a similar performance in Mel-cepstral distortion and root mean squared error in log F0. However, the DNN system is significantly better than the HMM system in distortion of band aperiodicities and voiced/unvoiced error. In the subjective evaluation, the DNN system outperforms consistently the HMM system in all three metrics including naturalness, intelligibility and MOS. This shows that by using deeper architectures we can achieve better performance for the TTS than using shallow architectures such as HMM or neural network with 1 hidden layer.

3.3.4. Effect of training data size

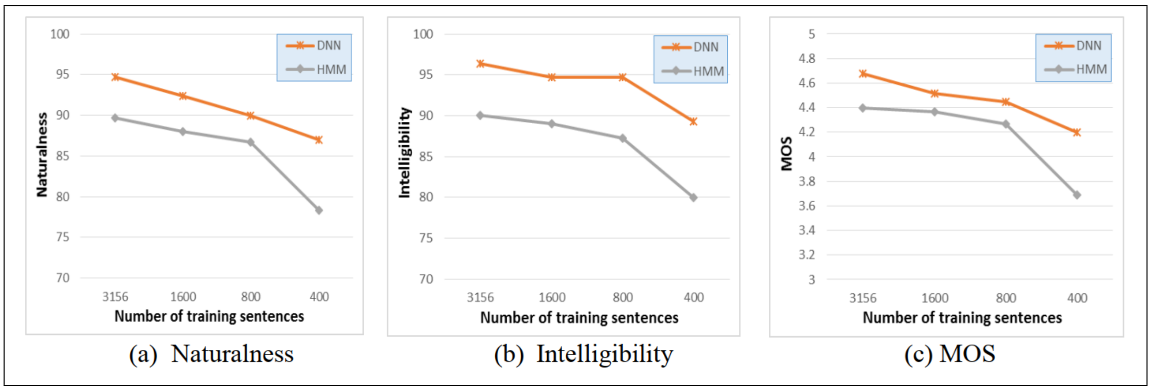

Figure 4. Subjective evaluation for both the DNN-based and HMM-based TTS systems with different amounts of training data

Now, we investigate the effect of training data size to TTS performance. We randomly sample the full training set (3156 sentences) to smaller subsets i.e., 1600, 800, and 400 sentences. Figure 4 shows subjective evaluation given by the DNN-based system (with 6 hidden layers) and the HMM-based system with different amounts of data to train the model. It can be seen that performance degradation is observed when using less training data for both the DNN and HMM systems. The DNN system achieved a significantly better performance in all aspects: naturalness, intelligibility and MOS metrics.

3.3.5. Effect of applying postfilter

In this section, we discuss the effect of applying postfilter to synthesized quality. Two DNN-based system with 6 hidden layers are compared: the first system is configured with postfilter and the second system is a normal system without postfilter. The subjective evaluation is shown in Table 3. It can be seen that the DNN-based system with postfilter archive better results in naturalness, MOS and Intelligibility.

3.3.6. Effect of applying parallel processing to postfilter

The result of previous section shows that, by applying postfilter to DNN-based speech synthesis system, notable improvement in synthesized quality has been recorded. In this section, we compared time response of three DNN-based text to speech systems with 6 hidden layers: the Original Postfilter system (system with original postfilter from HTS),

Table 3. Subjective evaluation for both the DNN-based TTS with applying postfilter and DNN-based TTS without applying postfilter

Apply Postfilter	MOS	Naturelness	Intelligibility
No	4.39	83.73	92.05
Yes	4.67	94.67	96.33

the No Postfilter system (system without postfilter) and the Parallel Postfilter (system with parallelized postfilter).

We made a performance test to compare time response of three systems above. The test corpus is a set of the sentences with variable length (like 4 word, 5 word, 6 word, 10 word,). For each length, three sentences were used for testing. The average response time of each system for each length group is demonstrated in Figure 5. It is clear that by using parallel processing, the systems response faster and the difference in time performance is getting more significant as the length of the sentence increases.

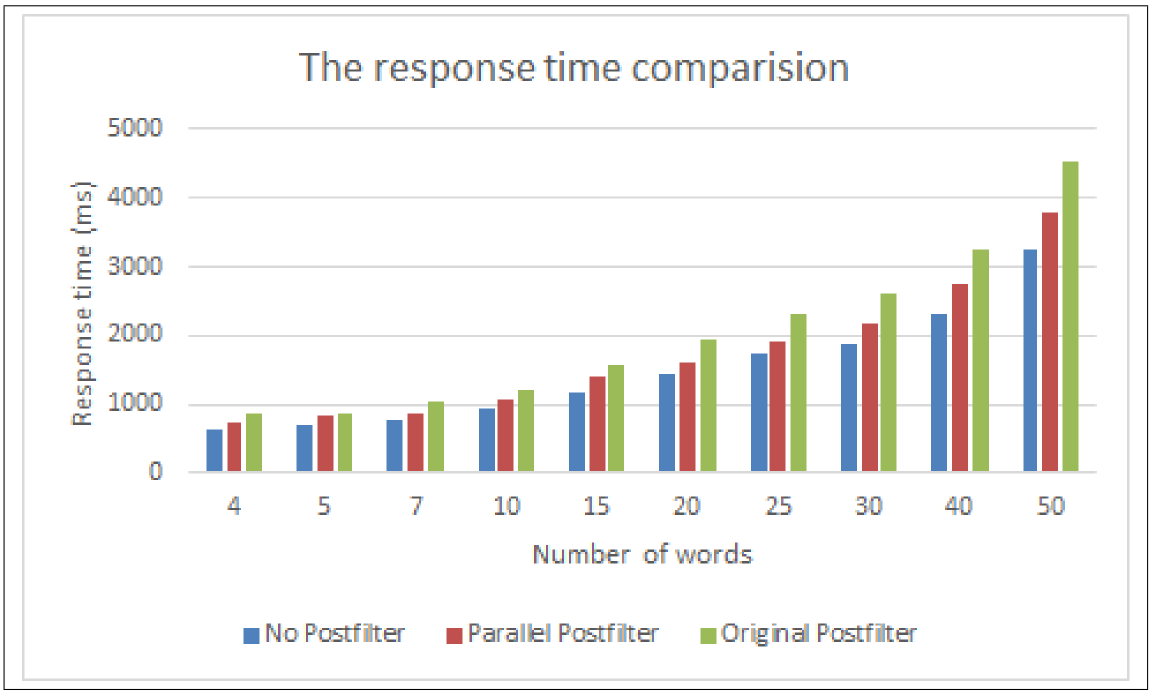

Figure 5. The response time comparison of three systems: No postfilter is the speech synthesis system without postfiltering, Original Postfilter is the system with the postfilter originated from HTS, and Parallel Postfilter is the system with the postfilter implemented by applying parallel processing

3.3.7. Performance in the VLSP TTS challenge 2018

Our proposed TTS system was also submitted to the VLSP TTS challenge 2018. The test set consists of 30 sentences in news domain. Each team needs to submit 30 corresponding

Table 4. The scores given by 3 teams in the VLSP TTS challenge 2018

Team	Naturalness	Intelligibility	MOS
VAIS	65.50	72.54	3.48
MICA	72.69	76.94	3.79
Our system (Viettel)	90.54	93.02	4.66

synthesized audio files. 20 people including males/females, different dialects, phoneticians and non-phoneticians were asked to provide score for naturalness, intelligibility and MOS.

As shown in Table 4, our TTS system (Viettel) won the first place and outperformed other TTS systems significantly in all subjects including naturalness, intelligibility, and MOS.

4. CONCLUSIONS

In this paper, we presented our effort to build the first DNN-based Vietnamese TTS system. To reduce the synthesized time, a method of using parallel processing postfilter was proposed. Experimental results showed that using cleaned data improves the quality of synthesized speech given by the TTS system. We also showed that by using deeper architectures, we can achieve better synthesized speech quality than using shallow architectures such as HMM or neural network with 1 hidden layer. The results also indicated that less training data also reduces speech quality. Generally talking, in all cases, the DNN system outperforms the HMM system. Our TTS system also won the first place in the VLSP TTS challenge 2018 in all three subjects including naturalness, intelligibility, and MOS. Our future work is to optimize the TTS systems for different dialects in Vietnam.

REFERENCES

[1] T. T. Do and T. Takara, "Precise tone generation for vietnamese text-to-speech system," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, vol. 1. IEEE, 2003, pp. 504–507.

[2] J. J. Ohala, "Christian gottlieb kratzenstein: pioneer in speech synthesis," *Proc. 17th ICPhS*, 2011.

[3] H. Dudley, "The carrier nature of speech," *Bell System Technical Journal*, vol. 19, no. 4, pp. 495–515, 1940.

[4] R. Billi, F. Canavesio, A. Ciaramella, and L. Nebbia, "Interactive voice technology at work: The cselt experience," *Speech communication*, vol. 17, no. 3-4, pp. 263–271, 1995.

[5] R. W. Sproat, *Multilingual text-to-speech synthesis.* KLUWER academic publishers, 1997.

[6] A. W. Black, H. Zen, and K. Tokuda, "Statistical parametric speech synthesis," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4. IEEE, 2007, pp. IV–1229.

[7] H. Ze, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on.* IEEE, 2013, pp. 7962–7966.

[8] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR abs/1609.03499*, 2016.

[9] S. O. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, A. Ng, J. Raiman *et al.*, "Deep voice: Real-time neural text-to-speech," *arXiv preprint arXiv:1702.07825*, 2017.

[10] D. T. Nguyen, M. C. Luong, B. K. Vu, H. Mixdorff, and H. H. Ngo, "Fujisaki model based f0 contours in vietnamese tts," in *Eighth International Conference on Spoken Language Processing*, 2004.

[11] A.-T. Dinh, T.-S. Phan, T.-T. Vu, and C. M. Luong, "Vietnamese hmm-based speech synthesis with prosody information," in *Eighth ISCA Workshop on Speech Synthesis*, 2013, pp. 55–59.

[12] L. He, J. Yang, L. Zuo, and L. Kui, "A trainable vietnamese speech synthesis system based on hmm," in *Electric Information and Control Engineering (ICEICE), 2011 International Conference on.* IEEE, 2011, pp. 3910–3913.

[13] S. Kayte, M. Mundada, and J. Gujrathi, "Hidden markov model based speech synthesis: A review," *International Journal of Computer Applications (0975–8887) Volume*, 2015.

[14] T. T. Vu, M. C. Luong, and S. Nakamura, "An hmm-based vietnamese speech synthesis system," in *Speech Database and Assessments, 2009 Oriental COCOSDA International Conference on.* IEEE, 2009, pp. 116–121.

[15] Q. S. Trinh, "Hmm-based vietnamese speech synthesis," in *2015 IEEE/ACIS 14th International Conference on Computer and Information Science (ICIS).* IEEE, 2015, pp. 349–353.

[16] P. G. Shivakumar and P. Georgiou, "Transfer learning from adult to children for speech recognition: Evaluation, analysis and recommendations," *arXiv preprint arXiv:1805.03322*, 2018.

[17] K. Yun, J. Osborne, M. Lee, T. Lu, and E. Chow, "Automatic speech recognition for launch control center communication using recurrent neural networks with data augmentation and custom language model," in *Disruptive Technologies in Information Sciences*, vol. 10652. International Society for Optics and Photonics, 2018, p. 1065202.

[18] M. Airaksinen, "Analysis/synthesis comparison of vocoders utilized in statistical parametric speech synthesis," *Master's thesis, Aalto University*, 2012.

[19] M. Morise, H. Kawahara, and H. Katayose, "Fast and reliable f0 estimation method based on the period extraction of vocal fold vibration of singing voice and speech," in *Audio Engineering Society Conference: 35th International Conference: Audio for Games.* Audio Engineering Society, 2009.

[20] M. Morise, "Cheaptrick, a spectral envelope estimator for high-quality speech synthesis," *Speech Communication*, vol. 67, pp. 1–7, 2015.

[21] M. Morise., "Platinum: A method to extract excitation signals for voice synthesis system," *Acoustical Science and Technology*, vol. 33, no. 2, pp. 123–125, 2012.

[22] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Incorporating a mixed excitation model and postfilter into hmm-based text-to-speech synthesis," *Systems and Computers in Japan*, vol. 36, no. 12, pp. 43–50, 2005.

[23] Z. Wu, O. Watts, and S. King, "Merlin: An open source neural network speech synthesis system," *Proc. SSW, Sunnyvale, USA*, 2016.

[24] Z. Zhang, M. Li, Y. Zhang, W. Zhang, Y. Liu, S. Yang, and Y. Lu, "The i2r-nwpu-ntu text-to-speech system at blizzard challenge 2016," in *Proc. Blizzard Challenge workshop*, 2016.

[25] K. Pärssinen and M. Moberg, "Multilingual data configurable text-to-speech system for embedded devices," in *Multilingual Speech and Language Processing*, 2006.

[26] Z.-Z. Wu, E. S. Chng, and H. Li, "Development of hmm-based malay text-to-speech system," in *Proceedings of the Second APSIPA Annual Summit and Conference*, 2010, pp. 494–497.

[27] D. Jurafsky and J. H. Martin, "Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition," pp. 1–1024, 2009.

[28] A. Bizzocchi, "How many phonemes does the english language have?" *International Journal on Studies in English Language and Literature (IJSELL)*, vol. 5, pp. 36–46, 10 2017.

[29] B. Phm and S. McLeod, "Consonants, vowels and tones across vietnamese dialects," *International journal of speech-language pathology*, vol. 18, no. 2, pp. 122–134, 2016.

[30] M. Brunelle, "Northern and southern vietnamese tone coarticulation: A comparative case study," *Journal of Southeast Asian Linguistics*, vol. 1, no. 1, pp. 49–62, 2009.

[31] M. Brunelle., "Tone perception in northern and southern vietnamese," *Journal of Phonetics*, vol. 37, no. 1, pp. 79–96, 2009.

[32] J. Edmondson and N. V. Li, "Tones and voice quality in modern northern vietnamese: instrumental case studies.," *Mon-Khmer Studies*, vol. 28, 1997.

[33] L.-H. Chen, T. Raitio, C. Valentini-Botinhao, Z.-H. Ling, and J. Yamagishi, "A deep generative architecture for postfiltering in statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 11, pp. 2003–2014, 2015.

[34] K. Koishida, K. Tokuda, T. Kobayashi, and S. Imai, "Celp coding based on mel-cepstral analysis," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE, 1995, pp. 33–36.

[35] T. Masuko, "Hmm-based speech synthesis and its applications," *Institute of Technology*, p. 185, 2002.

[36] M. Morise, F. Yokomori, and K. Ozawa, "World: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

Received on October 04, 2018
Revised on December 28, 2018