

博士論文

**Improving quality and flexibility of
deep neural network based
text-to-speech synthesis**

(深層ニューラルネットワーク型テキスト音声合成における自然性と柔軟性の改良)



2018年6月1日

指導教員 峯松信明教授

電気系工学専攻

37-147262 趙禕

Summary

Although deep neural networks have been applied to text-to-speech systems and shown promising future, it still suffers from the degradation of synthesized speech quality and distortion of speaker identity. In this thesis, we focus on improving quality and flexibility in neural network based text-to-speech system.

We firstly propose a multi-speaker speech synthesis framework that is built on bidirectional long short-term memory networks to synthesize high-quality speech with limited database. In this framework, a speaker independent model is trained combined with speaker identity vector. This speaker independent model can be viewed as a transformation from linguistic features to generalized vocoder parameters. Then speaker dependent models for transforming speaker independent vocoder parameters to those of the target speaker are followed. The proposed framework can also be applied to speaker adaptation.

Further, in multi-speaker speech synthesis, both speaker adaptive training and speaker adaptation techniques need to learn speaker identity precisely. Therefore, we conduct a series of comparative studies on different speaker representations for controlling speaker identity. We focus on speaker identity control at the input layer of our proposed framework, and investigate different speaker representations like i-vector and speaker code when they are used as augmented input vector. We also propose two approaches to estimate a new speaker's code. The first one is estimating a new speaker's code from i-vector using Gaussian Mixture Model. The other is extracting the speaker codes extracted from mel-spectrogram features using recurrent neural networks.

However, traditional training criteria such as mean squared loss can lead to obvious mismatches between generated and natural acoustic parameters. It definitely decreases the quality of generated speech. In addition, vocoders built on speech related prior assumptions give rise to distortion

especially in phase reconstruction. To alleviate these problems, we propose frameworks that incorporate either a conditional generative adversarial network (GAN) or its variant, Wasserstein GAN with gradient penalty, into multi-speaker speech synthesis that uses the WaveNet vocoder. We also extend the GAN frameworks and use the discretized mixture logistic loss of a well-trained WaveNet in addition to mean squared error and adversarial losses as parts of objective functions. Experimental results show that proposed framework using back-propagated discretized-mixture-of-logistics loss achieves the highest subjective evaluation scores in terms of both quality and speaker similarity.

Besides, linguistic features such as prosodic boundaries affects speech naturalness a lot. Instead of traditional cascaded prediction, we propose a unified framework for Chinese prosodic boundary prediction. This framework can be considered to be a multi-task learning process . We also explore various text features and representations to predict prosodic boundaries without task specific knowledge or sophisticated feature engineering. The synthesized speech naturalness can be improved by increasing the accuracy of prosodic boundary prediction.

Contents

1	Introduction	1
1.1	General background	1
1.2	Issues addressed in this thesis	3
1.3	Organization of the thesis	4
2	Neural Networks for Speech Synthesis	5
2.1	Neural Network Structures	5
2.1.1	Feedforward Deep Neural Networks	5
2.1.2	Recurrent Neural Networks	7
2.1.3	Convolutional Neural Networks	9
2.2	Neural network for statistical parametric speech synthesis	9
2.3	Neural network for directly audio sample modeling	11
2.4	Evaluation methods for text-to-speech synthesis	12
2.5	Conclusion	14
3	BLSTM-RNN-based Multiple Speaker Speech Synthesis and Adap- tation	15
3.1	Introduction	15
3.2	Related work	17
3.2.1	DNN-based speaker adaptation	17
3.2.2	BLSTM-RNN for speech synthesis	18
3.3	BLSTM-RNN-based multi-speaker modeling and adaptation	19
3.3.1	BLSTM-RNN-based multi-speaker speech synthesis	19
3.3.2	Speaker adaptation after multi-speaker synthesis	21
3.4	Experiments	21
3.5	Evaluation and discussion	22
3.5.1	Evaluations for speech synthesis	23
3.5.2	Evaluations for speaker adaptation	26

3.6	Conclusions	27
4	Speaker Representations for Multi-speaker Synthesis and Adaptation	29
4.1	Introduction	29
4.2	Framework for multi-speaker speech synthesis and adaptation	30
4.3	General speaker identity representations	32
4.3.1	I-vector	32
4.3.2	Speaker Code	33
4.4	Independently estimated discriminating code	33
4.4.1	Speaker code estimation from i-vector	33
4.4.2	Speaker code estimation from MFCCs	34
4.5	Experimental setup	34
4.6	Evaluation and discussion	35
4.6.1	Evaluations for multi-speaker speech synthesis	36
4.6.2	Evaluations for speaker adaptation	36
4.7	Conclusions	39
5	Wasserstein GAN and Waveform Loss-based Acoustic Model Training Using a WaveNet Vocoder	40
5.1	Introduction	41
5.2	DNN-based Multi-speaker Speech Synthesis	43
5.3	Multi-speaker Speech Synthesis Incorporating GAN and WaveNet Vocoder	44
5.4	Components of Proposed Model Structure	45
5.4.1	SRU	45
5.4.2	Generative Adversarial Network	46
5.4.3	WaveNet	48
5.4.4	DML loss	49
5.5	Training Algorithm	50
5.5.1	Training algorithm for the proposed acoustic model	50
5.5.2	Time resolution adjustment	51
5.6	Experimental Setup	53
5.7	Experimental Evaluation	55
5.7.1	Evaluation methodology	55
5.7.2	Evaluation results and analysis	56
5.8	Conclusion	59

6	Prosody Prediction in Text to Speech Synthesis	61
6.1	Introduction	61
6.2	Neural network based prediction	63
6.2.1	Network Structures and Training	63
6.2.2	Tag inference	64
6.3	Proposed Approach	65
6.3.1	Word-level features vs. character-level features	65
6.3.2	One-hot features vs. embedding features	65
6.3.3	Unified framework vs. Cascaded framework	66
6.4	Experimental Setup	67
6.5	Evaluations and Analysis	68
6.5.1	Objective evaluation	68
6.5.2	Subjective evaluation	70
6.6	Conclusions	71
7	Conclusions	72
A	Speaker code based stimulated training for network visualization	73
A.1	Introduction	73
A.2	Stimulated Deep Learning	74
A.3	Network Interpretation in Terms of Speaker Identity	76
A.3.1	Experiments on network visualization	78
A.4	Conclusion	80
	Bibliography	81
	Dedication	93
	List of Publications	94

List of Figures

2.1	A basic DNN network	6
2.2	Compute process in a feed-forward network.	6
2.3	A recurrent neural network and the unfolding in time of the computation involved in its forward computation.	7
2.4	An LSTM cell	8
2.5	A speech synthesis framework based on deep neural network, cited from [1].	11
2.6	Using WaveNet to generate audio samples from linguistic features or acoustic features.	12
3.1	DNN-based speaker adaptation	17
3.2	DBLSTM-RNN based single speaker speech synthesis model	19
3.3	DBLSTM-RNN based multi-speaker speech synthesis model	20
3.4	Augmented input vector for speaker independent model	20
3.5	Subjective comparison for multi-speaker speech synthesis in terms of speech quality averaged across all speakers	24
3.6	Subjective comparison for multi-speaker speech synthesis in terms of similarity	25
3.7	Subjective comparison for speaker adaptation in terms of speech quality averaged across all speakers	27
3.8	Subjective comparison for speaker adaptation in terms of similarity	27
4.1	BLSTM-RNN based multi-speaker speech synthesis model	32
4.2	Subjective evaluations for multi-speaker synthesis in terms of naturalness (a) and similarity (b).	38
4.3	Subjective evaluations for speaker adaptation in terms of naturalness (a) and similarity (b).	39
5.1	Proposed GAN-trained multi-speaker speech synthesis framework using a WaveNet vocoder.	44

5.2	Details of the SRU cell. $\sigma(\cdot)$ and $g(\cdot)$ represent sigmoid and ReLU activation functions, respectively.	46
5.3	GAN-based training of TTS acoustic model. L_{ADV} indicates adversarial loss and L_{MSE} indicates L2 loss.	46
5.4	Local condition and global condition used in a WaveNet model. . . .	48
5.5	Loss functions and gradients for updating acoustic models in the proposed method. Note that neither the model parameters of WaveNet nor the discriminator are updated in this step.	50
5.6	Time resolution adjustment of conditional acoustic features. One frame includes four waveform audio points. Transposed convolutional layers are used to upsample the conditional acoustic features. The DML loss was computed using randomly selected waveform audio points within each frame.	53
5.7	Box plots on naturalness evaluation results. Red dots represent the mean of each group averaged across all speakers.	57
5.8	Box plots of the MOS scores of six speakers. Left: WGAN-GP ^W system. Right: AbS system.	58
5.9	Similarity results averaged across all speakers.	58
5.10	Scatter plot matching naturalness and similarity scores for each speaker in system WGAN-GP ^W and AbS. The similarity score is defined as the added percentage of ‘same (not sure)’ and ‘same (sure)’ scores. . . .	59
6.1	The hierarchical prosodic structure in Chinese.	62
6.2	The neural network architecture for prosodic boundary prediction. In tag inference, B, NB and O denote boundary, non-boundary and others (e.g., punctuation), respectively.	64
6.3	Framework of prosodic structure prediction. M represents for neural network model	67
6.4	Subjective comparison for synthesized speech naturalness using different features	71
A.1	Activation of normal DNN vs. stimulated DNN	76
A.2	2-dimensional mapping of utterance level i-vectors	79
A.3	Stimulated activations for speaker ‘jmk’ and ‘slt’	79
A.4	Stimulated output activation in different layers	79

List of Tables

2.1	MOS scale	13
3.1	Objective distortion in terms of speech synthesis. This table is the average loss across all female speakers	25
3.2	Objective distortion in terms of speech synthesis. This table is the average loss across all male speakers	25
3.3	Objective comparison in terms of speaker adaptation. Using 100 utterances of a new speaker for adaptation training.	26
3.4	Objective comparison in terms of speaker adaptation. Using 30 utterances of a new speaker for adaptation training.	26
4.1	Objective evaluations for female speakers in multi-speaker’s speech synthesis. Individual synthesis is trained by only one female speaker’s data.	37
4.2	Objective evaluations for male speakers in multi-speaker’s speech synthesis. Individual synthesis is trained by only one male speaker’s data.	37
4.3	Objective evaluations for new speaker’s adaptation using 100 adaptation utterances . Individual synthesis is trained using the same data of the target speaker.	37
4.4	Objective evaluations for new speaker’s adaptation using 30 adaptation utterances . Individual synthesis is trained using the same data of the target speaker.	37
4.5	Objective evaluations for the new speaker’s speech synthesis using 900 sentences	38
5.1	Statistical significance analysis using t -tests with Holm-Bonferroni correction in terms of quality judgment.	57
5.2	Statistical significance analysis using t -tests with Holm-Bonferroni correction in terms of speaker similarity judgment.	59

6.1	F-score (%) of CRF-based prosody prediction	69
6.2	F-score (%) of N-gram character-level one-hot features	69
6.3	F-score (%) of cascaded prediction. C represents for character-level features, W represents for word-level features	70
6.4	F-score (%) of unified prediction. C represents for character-level features, W represents for word-level features	70
A.1	Objective evaluations for multi-speaker synthesis on the testing data of speaker “slt”.	80

Chapter 1

Introduction

1.1 General background

Text-to-Speech synthesis (TTS), is a technique which can convert any input text into corresponding speech. With the advancement of human-computer interaction technology, TTS becomes an indispensable module in human-machine speech interaction since it can improve the situation of human-computer interaction, making communication between humans and computers more convenient and faster. Whether it is in voice assistance, education, entertainment and other software applications, or in smart speakers, car navigation, robots and other hardware devices, there are lots of applications related to text to speech technology.

The traditional TTS techniques are generally classified into two main lines: one is based on unit selection with waveform concatenation (hereinafter referred to as concatenation synthesis) and the other is Statistical Parameter Speech Synthesis (SPSS) (hereinafter referred to as SPSS). Concatenation synthesis usually stores a large set of speech segments and select the speech fragments in the time domain. It uses statistical models to guide unit selection [2]. The training phase is similar to parametric speech synthesis. In the synthesis stage, the unit selection is guided by the cost of the model, and usually the dynamic programming algorithm is used to select the optimal unit sequence, and then the selected units are putting through energy regularization and waveform concatenation. The synthesized speech can achieve high quality as well as keep the identity of the original speaker. The disadvantage is that the required audio corpus is always very large, and it is not possible to guarantee the synthesis quality of texts which are outside the corpus domain.

Compared with the concatenation synthesis system, SPSS systems are more flexible and requires less database. A general SPSS system usually includes a front-end module, a middle-end module as well as a back-end module. The front-end module

mainly analyzes the input text and extracts the linguistic information needed by the middle module. The front-end module has close relationship to languages and it generally includes sub-modules of text regularization, word segmentation, part of speech prediction, polyphonic word disambiguation, and prosody prediction. The middle-end module is generally called as acoustic model. It usually maps the output of front-end module to the input of the back-end module. The back-end module generates a speech waveform by a certain method according to the results of the front-end and middle-end module. In SPSS [3], acoustic and duration modeling are performed based on context-related information in training phase. In synthesis stage, the acoustic feature parameters are predicted through the acoustic and duration model, and then acoustic parameters are post-processed and put through a signal processing algorithm named vocoder to generated waveform. This method can obtain a relatively stable synthesis effect when the speech corpus is relatively small. And it is more flexible than unit selection based synthesis since it can synthesize speeches which are out of the corpus. However, it still suffers from obvious synthesized speech quality deterioration.

Speech synthesis systems in the past several decades generally use hidden Markov models for statistical modeling. In recent years, neural networks have been increasingly applied to speech synthesis because of their high modeling accuracy. The neural network models used in speech synthesis technology mainly include the general feed-forward network (DNN) [4], recurrent neural network(RNN) with long short term memory(LSTM) [5] [6].

In recent years, some new speech synthesis methods which aim at directly waveform generation or end-to-end processing have been continuously proposed, and a more ideal synthetic effect has been achieved. For example, the WaveNet [7] synthesis system proposed by the DeepMind team constructs an autoregressive model and directly models the time-domain sampling points, resulting in highly natural synthetic speech. The Char2Wav model [8] proposed by the Yoshua Bengio team, the DeepVoice [9–11] proposed by Baidu team and the Tacotron [12, 13] system proposed by Google directly establish the mapping relationship between the input text and the output speech, and realize the end-to-end speech synthesis. The synthesized speech effect can almost approach the level of human speech.

In today’s practical applications, speech synthesis technology has been required to support multi-speaker, multi-dialect, multi-language, choices, with high synthesized speech quality. Flexible speech synthesis with high synthesized speech quality become a new trend. Voice conversion and speaker adaptation are two general approaches for generating multiple speakers’ voice. Voice conversion usually aims at converting an

existing speaker’s voice to another speaker by modeling the source and target speaker’s acoustic features or speeches directly. Speaker adaptation is usually built on speech synthesis framework, and aims at adapt the existing speaker’s voice to a new speaker which is not included in the training corpus. Since speaker adaptation techniques can provide speeches with high naturalness, a lot of recent works are focusing on it. WaveNet Vocoder based multi-speaker synthesis with 4 speakers from CMU arctic corpus [14]. DeepVoice 3 [11] train their multi-speaker model with over two thousand speakers and VoiceLoop [15] involve 109 speakers’ data for acoustic model training. Wang et al. [16] propose to use a bank of style embeddings to deal with animated and emotive storytelling style.

1.2 Issues addressed in this thesis

In this paper, we focus on improving quality and flexibility of text-to-speech system. Although neural networks have become a new trend for speech synthesis, it still faces with problems in practical application. For example, neural networks based speech synthesis framework usually need a lot of specific speaker’s data to train a high quality acoustic model. Multi-speaker speech synthesis incorporating speaker adaptive training is a good approach to solve this problem. By using neural network based multi-speaker speech synthesis framework, each speaker can benefit from the knowledge of other speakers and synthetic speech of the target speaker can be obtained robustly and steadily even if speech samples available for the target speaker are very small. However, the generated speaker identity which use multi-speaker models are easily to be mixed. Controlling of speaker identity during speaker adaptive training become a serious problem. To improve the accuracy of speaker identity distinction, the representation vectors of speaker identity used in acoustic modeling need to be carefully treated. Neural networks provide us new ways to solve such problem. Multi-speaker based approaches can also be applied to the state of the art generative models such as GAN or WaveNet. Improving the quality as well as keeping speaker characteristic require with these models need to be fully studied. In addition to acoustic modeling, increasing the accuracy of prosody prediction can also improve generated speech naturalness.

1.3 Organization of the thesis

In the second chapter, we give a brief introduction on the state of the art text-to-speech techniques based on deep neural networks. We point out that flexible speech synthesis with high fidelity has been a frontier topic. The third chapter will introduce a proposed framework for multiple speaker speech synthesis as well as speaker adaptation based on BLSTM-RNN. Based on the proposed framework, chapter 4 mainly discuss speaker representations for controlling speaker identity during speaker adaptive training. And two methods for estimating a new speaker’s code from the existing speakers’ identity space have been proposed. We integrate the speaker code with gate and state activation function by additional connection weights, and automatically estimate a a new speaker’s code by back propagation during speaker adaptive training. In chapter 5, we propose a new multi-speaker synthesis framework by incorporating GAN-based acoustic modeling and WaveNet vocoder which could improve the generated speech quality obviously. After that, we present an exploration study on speech prosody prediction. A conclusion of the thesis is given at last.

Chapter 2

Neural Networks for Speech Synthesis

Deep Learning is one of many machine learning algorithms. It allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. So far, it is the most powerful type of algorithm in the field of “supervised learning”. At the same time, it is also spreading to the fields of “unsupervised” and “enhanced learning” [17]. Deep Learning relies on powerful generalization ability to blossom in more and more application fields, but it is extremely lack of theoretical basis. Regardless of the interpret ability of the model, the design of the structure, the selection of parameters, etc., most of them rely on experience, trial and error at this stage. Many of the tools that might have been known as trick before may be an important method of tuning in Deep Learning. This chapter is intended to begin with different neural network structures and introduce the application of deep learning in text-to-speech synthesis.

2.1 Neural Network Structures

2.1.1 Feedforward Deep Neural Networks

Distinguished from the common single-hidden-layer neural networks, Feedforward Deep Neural Networks (DNNs) pass the data into a number of node layers in a multi-step process of pattern recognition. A basic DNN network is shown in Fig.2.1. Feed-forward structure is the basic structure of DNNs. Here is a simple explanation of what happens during learning with a feed-forward neural network. We use x represents a training sample vector and y represents the desired output vector. z_i^l denotes the input of the i -th neuron in the l -th layer, a_i^l denotes the output of the i -th neuron in the l -th layer, and b_i^l denotes the offset corresponding to the i -th neuron in the

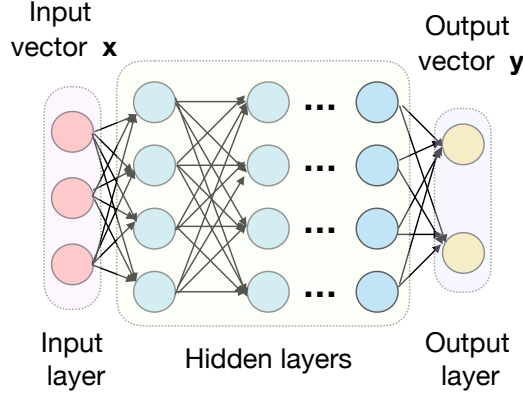


Figure 2.1: A basic DNN network

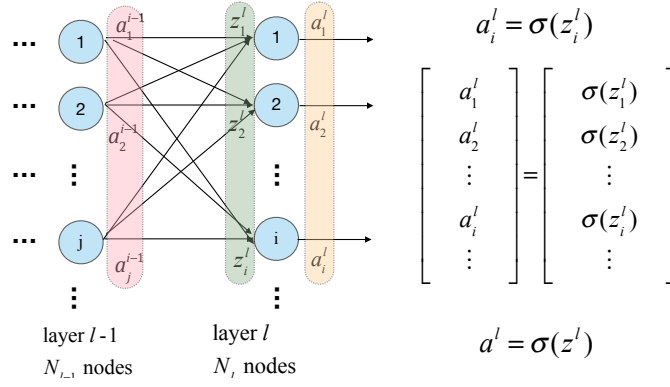


Figure 2.2: Compute process in a feed-forward network.

l -th layer. w_{ij}^l represents the weight from a_j^{l-1} to z_i^l , so that z_i^l is essentially a linear combination of all $l-1$ -th output and b_i^l . At the same time, the weights of the 1st layer to the 1st layer can be written in the form of a vector W^l , where the i -th column shows the weights issued from a_i^{l-1} . We use z^l , a^l , and b^l to represent the vectors of all inputs in the first layer, the vectors of all outputs, and the vectors of all offsets. Thus, the relationship between z^l and a^{l-1} can be expressed as $z^l = W_l * a^{l-1} + b^l$. The process is described as Fig.2.2.

In practice, DNNs generally use backpropagation (BP) algorithm to optimize the weights of the model. By taking advantage of the chain rule, the BP algorithm try to find the optimum optimization path from top to the bottom layer. Weight updates can be solved using the stochastic gradient descent method using the following formula:

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial L}{\partial w_{ij}} \quad (2.1)$$

where η is the learning rate, and L is the cost function. The choice of cost function is

related to learning method (eg: supervised learning, unsupervised learning, enhanced learning) and activation functions.

Although DNN can represent high dimensional and correlated features efficiently and model highly complex mapping function compactly, it still suffers. Due to its feed-forward architecture, DNN provides only limited temporal modeling by operating on a fixed-size sliding window of several feature frames. Also, DNN can only model the data within the window and are unsuited to handle different speaking rates and longer term dependencies.

2.1.2 Recurrent Neural Networks

To compensate the deficiencies in DNN based acoustic model, recurrent neural networks (RNNs) are proposed to be used for tasks that involve sequential inputs, such as speech and language. RNNs process an input sequence one element at a time, maintaining in their hidden units a 'state vector' that implicitly contains information about the history of all the past elements of the sequence [17]. An RNN architecture is showed in Fig. 2.3. RNN embodies correlations of time sequence. It can access

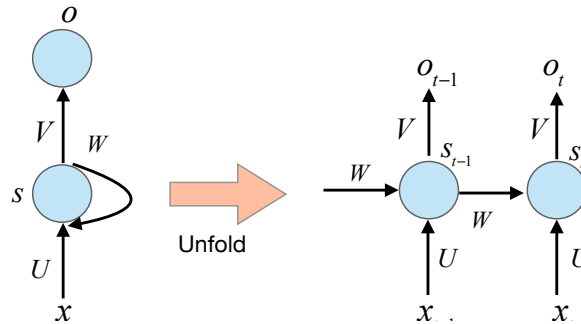


Figure 2.3: A recurrent neural network and the unfolding in time of the computation involved in its forward computation.

input information at both past and future frames. In RNN(Fig. 2.3), an artificial neuron receives input from other neurons at a previous time step. For example, a hidden unit under node s having a value of s_t at time t , it receives information from s_{t-1} . In this way, a recursive neural network can map an input sequence with element x_t to an output sequence with elements o_t , where each o_t depends on all previous $x'_t (for t' \leq t)$. Matrices (U, V, W) are used at each time step. The unfolded RNNs can be seen as very deep feedforward networks in which all the layers share the same weights.

Although the main purpose of RNN is to learn long-term dependencies, theoretical and empirical evidence shows that it is difficult to learn to store information for very long [18]. Gradient Vanishing is prone to occur in RNN because when the gradient is passed backwards, the gradient tends to gradually disappear due to too many times of the same matrix multiplication with a derivative < 1 . As a result, the following nodes cannot update parameters, and the whole learning process cannot be performed normally.

LSTM can overcome the gradient vanishing problem in RNN and model signals that have a mixture of low and high frequency components. An LSTM cell is showed in 2.4. $\sigma(\cdot)$ is the logistic function, and i , f , o and c are the input gate, forget gate, output gate and cell memory, respectively.

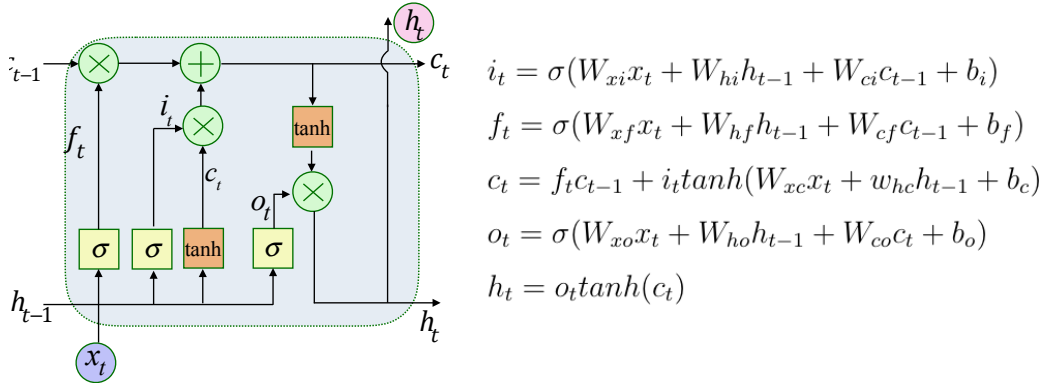


Figure 2.4: An LSTM cell

In the recurrence of the LSTM the activation function, the effective weight of the recurrence is equal to the forget gate activation. Although the identity function always has a derivative of 1.0 and the backpropagated gradient neither vanishes or explodes when passing through, it remains constant. But with the help of a forget gate activation which is close to 1.0 but never > 1.0 , the gradient would not vanish and can't explode either. Thus LSTM is good at learning long range dependencies. Deep bidirectional LSTM (DBLSTM) is the integration of deep bidirectional RNN and LSTM. By taking the advantages of DNN and LSTM, it can model the deep representation of long-span features.

2.1.3 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image (or other 2D input such as a speech signal). This is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units.

2.2 Neural network for statistical parametric speech synthesis

Although the HMM based acoustic model has achieved great success in statistical parametric speech synthesis (SPSS), it faces with many disadvantages. For example, it typically uses lower dimensional approximation of speech spectrum as acoustic feature (e.g., cepstrum, line spectral pairs), because it is hard to model spectrum directly by HMM-GMM due to high dimensionality and strong correlation [19] [20]. Another problems occurs during the Linguistic-to-acoustic mapping by decision trees. Decision tree splits input space into sub-clusters, which is inefficient to represent complex dependencies between linguistic and acoustic features [21]. Other problems include the assumption of frame-wise conditional independence of state-output probabilities, and simple geometric state-duration distributions. None of these assumptions hold for real speech [3].

Because of the inevitable drawbacks of HMM based SPSS, deep learning methods were tried to be used in SPSS and aimed at improving the performance of HMM based acoustic model. Deep learning is a machine learning methodology using multiple-layered models which has been successfully applied to image processing, speech recognition. Deep learning based acoustic models can model high-dimensional, highly correlated features efficiently. Deep learning can also be exponentially more efficient than fragmented representation and has better representation ability with fewer parameters [1].

Neural networks have been proposed as the poster filter to simulate the relationship between the reconstructed spectral and real spectrum of speech signal in [22]. In

work [23] and [24], the authors proposed HMM-DBN joint training model. In HMM-DBN model, DBNs were used to replace GMMs and linguistic features are clustered by decision tree based on HMM model with DBN state-output distributions. However, this joint model still cannot get out the reliance of handcrafted cluster trees. In [25], DBN was proposed to replace decision trees and GMMs and represents joint distribution of linguistic and acoustic features. [26], F0 contour prediction with a deep belief network-Gaussian process hybrid model was proposed. The author replaced last layer of DNN by GP regression, the last hidden layer output was used as input for Gaussian Process (GP) regression.

DNN was used to replace decision trees and GMMs and used as the acoustic model between linguistic labels and acoustic features directly in recent two years. The work [4] introduced speech synthesis with DNN based acoustic model instead of HMM based acoustic models directly. In [27], it introduced the results based on different pre-training algorithms and different number of hidden layers with DNN based acoustic model, and also compared HMM based technology and DNN based technology. In [28], the authors proposed HMM+NN combined training method in order to compensate information of the relationship between adjacent frames. The paper [29] proposed multi-task training and bottleneck features for improving the accuracy of DNN based speech synthesis. DNN based acoustic model is much more hierarchical clearly and express complicated mapping functions more efficiently thus provide better generalization. Frame-to-frame DNN based TTS architecture is showed in Fig. 2.5.

Although DNN can represent high dimensional and correlated features efficiently and model highly complex mapping function compactly, it still suffers. Due to its feed-forward architecture, DNN can provide only limited temporal modeling by operating on a fixed-size sliding window of acoustic frames. DNN can only model the data within the window and are unsuited to handle different speaking rates and longer term dependencies. In DNN based acoustic model, it keeps the assumption of frame independence without considering time sequence relationship of speech signal.

In addition to DNN, other neural networks structures such as bi-directional and unidirectional LSTM-RNN [5, 6, 30], GAN have also been used for statistical parametric speech synthesis (SPSS) and achieved rather good results.

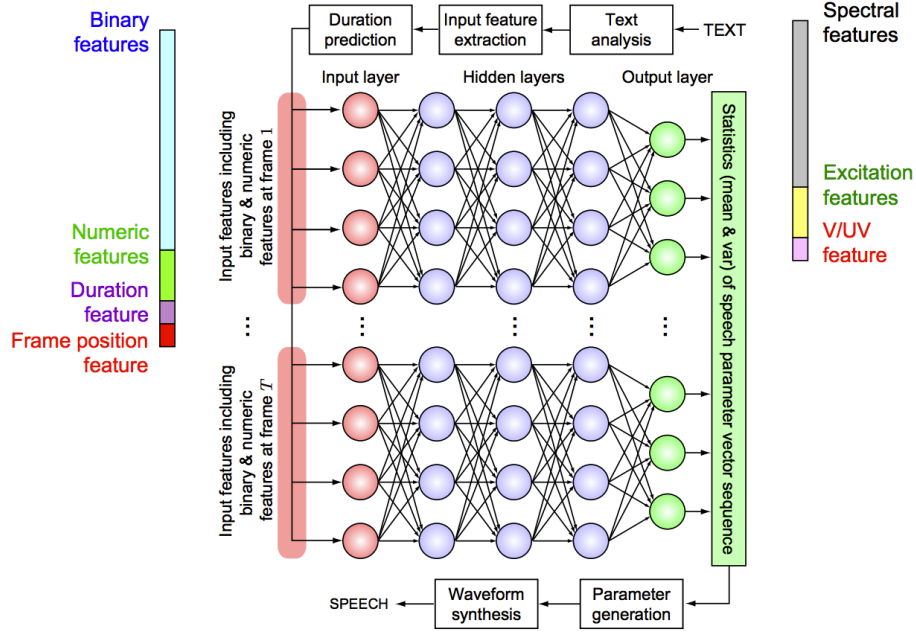


Figure 2.5: A speech synthesis framework based on deep neural network, cited from [1].

2.3 Neural network for directly audio sample modeling

Although neural networks has achieved great success in SPSS, it still suffers from the quality deterioration due to the distortion of statistic vocoders like STRAIGHT [31] or WORLD [32]. Waveform-based method like WaveNet [7] and SampleRNN [33] are proposed to problems caused by parametric vocoders. WaveNet is Google’s latest deep-learning-based speech generation model. The model can directly model the original speech data and works well in text-to-speech and speech generation tasks. Wavenet uses a method that dilated convolution and causal convolution, allowing the receptive field to multiply as the depth of the network increases, allowing the original speech data to be modeled. The main idea of WaveNet is the casual layer. In order to cover a larger receptive field, a larger kernel size convolution kernel is used. The larger kernel size brings about a rapid increase in computation. In order to solve this problem, the article uses The dilate convolution described above. In addition to the first layer or the traditional convolution operation, the remaining layers are all dilate convolution. The residual and skip connection techniques are used in the WaveNet to make the model converge faster and allow the gradient to reach deeper models. As is shown in Fig.2.6, WaveNet can be used to model speeches from either linguistic

labels [7] or acoustic parameters [34].

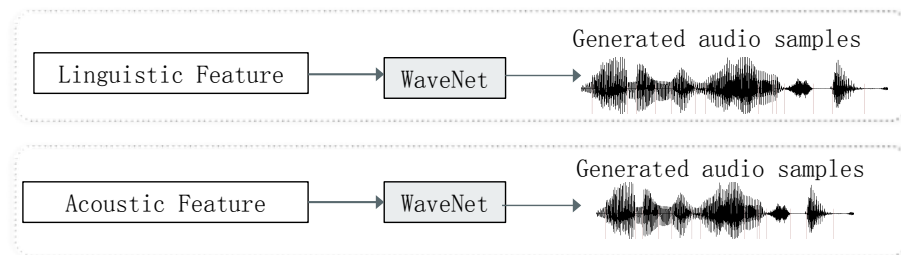


Figure 2.6: Using WaveNet to generate audio samples from linguistic features or acoustic features.

In a preference test, listeners are presented with a pair of samples and asked to indicate which one has more of a certain property. In AB test, two samples are given, the question would be “which one do you prefer”. In ABX test, an extra reference sample is also given. Listener has to judge if A or B more like X. This can be used for naturalness as well as speaker similarity.

2.4 Evaluation methods for text-to-speech synthesis

Effective TTS system evaluation has been an old topic for several decades [35, 36]. There are a number of different approaches to evaluate synthetic voices. Although subjective evaluations which rely on the perceptual performance of human listeners are generally considered to be the most reliable methods of speech synthesis evaluation, it is often criticized by the professional level of listeners and also the influence of the external environment. To perform a credible listening test, a large amount of well trained listeners are usually required to do the listening test under the same environment. It is very time consuming hard to satisfy all the requirements for general researchers. For these reasons, objective evaluation measures are usually utilized to assess the accuracy of parameter prediction, with a desire of being able to determine the quality and intelligibility of a speech synthesis, without needing to use significant numbers of human listeners.

Subjective evaluations include rating tests as well as preference tests. Among all the absolute category rating methods, mean opinion scores (MOS) is the most favourite one. During testing, the listeners are presented with the test speech stimuli and asked to rate the quality of the stimuli on a numerical scale, typically a 5-point

scale with one indicating poor quality and a five indicating excellent quality. The five quality categories are shown in Table 2.1.

Rating	Speech quality	Level of distortion
1	bad	Very annoying and objectionable
2	poor	Annoying, but not objectionable
3	fair	Perceptible and slightly annoying
4	good	Just perceptible, but not annoying
5	excellent	Imperceptible

Table 2.1: MOS scale

The absolute category rating methods includes Mean Opinion Scores (MOS), a “degradation” or “differential” MOS test (DMOS), Diagnostic Acceptability Measure (DAM) [37] and Multiple Stimuli with Hidden Reference and Anchor (MUSHRA) [38]. In this thesis, we mainly use When performing a subjective test, some general points that need to be considered when thinking about designing the evaluation, such as Listeners are asked to select the stimuli they prefer the most. In the rating tests,

Most objective measures of speech quality are implemented by computing a distortion measure between the original and processed signals frame by frame averaged across all frames. The distortion measure computation are always done in frequency domain by measuring the dissimilarity of acoustic parameters between the predicted and natural speeches. In our thesis, we mainly compute the distortion of cepstrum coefficients (MGC), fundamental frequency (F0), band of aperiodicity parameters (BAP) and voiced/unvoiced marks (V/UV). A measure based on cepstrum coefficients of signal x can be computed as follows:

$$d_{cep}(c_x, \hat{c}_x) = \frac{10}{\log_e 10} \sqrt{2 \sum_{t=1}^T [c_x(t) - \hat{c}_x(t)]^2} \quad (2.2)$$

where $c_x(t)$ and $\hat{c}_x(t)$ are the t -th frame cepstrum coefficients of the predicted and natural signals respectively. T is the total number of frames.

The root-mean-square error (RMSE) is used for computing the loss of F0.

$$d_{f0}(f_x, \hat{f}_x) = \sqrt{\frac{1}{T} \sum_{t=1}^T (f_x(t) - \hat{f}_x(t))^2} \quad (2.3)$$

where f_x, \hat{f}_x represents the F0 parameters of the predicted and natural signals respectively.

The calculation of BAP distortion depends on the composing method of aperiodicity parameters. If BAP are produced by converting aperiodicity parameters using the mel-scale, it usually uses the same algorithm as cepstrum distortion like 2.2. Otherwise it is calculated using the RMSE.

U/VU distortion is simply calculated as the ratio between the number of mis-predicted frames and the total number of frames.

2.5 Conclusion

In this chapter, we have briefly introduced the deep learning technology and its application in text-to-speech synthesis. In the last decades, major progress in text-to-speech synthesis came about through systems that utilize deep neural networks. We believe such technology will make a large impact in both research and industrial fields in the following few years.

Chapter 3

BLSTM-RNN-based Multiple Speaker Speech Synthesis and Adaptation

In this chapter, a deep bidirectional long short-term memory recurrent neural network (DBLSTM-RNN) based multi-speaker synthesis model is proposed to improve the synthesis quality for a target speaker whose corpus is limited. This model consists of speaker independent network (SIN) and speaker dependent network (SDN), where SIN is jointly trained by multiple speakers and SDN is designed for the target speaker. In particular, gender code as well as speaker code or i-vector are prepared as augmented input information to help SIN realize better distinction among different speakers. Experimental results show that our proposed model improves the quality of synthesized speech with a fairly small database for each speaker compared to DNN-based multi-speaker TTS and conventional DBLSTM-RNN based TTS. In addition, this multi-speaker model can also be used to perform speaker adaptation, and is experimentally shown to be capable of achieving good quality speech of a new speaker in terms of naturalness and speaker identity.

3.1 Introduction

Although various kinds of neural networks have shown promising results in acoustic modeling for statistical parametric speech synthesis (SPSS), they still suffer from the necessity of a large recording corpus for one speaker to train a high quality acoustic model. In [4], a DNN based acoustic model was trained with about 35000 sentences and achieved a better performance than traditional HMM-GMM based methods. [6] introduced unidirectional long short-term memory recurrent neural net-

works (ULSTM-RNNs) with 34632 training utterances and it outperformed DNN. [5] used DBLSTM for acoustic modeling, which was trained with 5000 utterances. [27] claimed that DNN can also be well-trained with 5000 utterances per speaker but it is still a huge data cost. In practice, it is practically difficult to prepare such a large amount of data.

To solve this problem, [39] proposed a multi-speaker DNN. In this DNN-based approach, the hidden layers of DNN are shared across different speakers while the regression layers are speaker independent. The multi-speaker DNN was jointly trained with different speaker’s data and it was experimentally shown to benefit the quality of each speaker’s synthesized speech. It seemed that, to realize speech synthesis of a target speaker, some knowledge of other speakers was helpful. In [39], however, the input to DNN did not contain any information on speaker identity, the shared hidden layers were incapable to distinguish inputs with the same linguistic content but spoken by different speakers.

Wu, et al. [40] proposed to use i-vector and gender code as speaker information and appended them to input vectors of DNN for speaker adaptation purpose. We expect that a similar approach can improve multi-speaker TTS.

Further, DNN has been recently claimed not to be an optimal model due to complexity of speech signal and its insufficiency to capture the temporal dependencies of speech sequences. Instead, by incorporating LSTM and RNN, the resulting network was experimentally shown to be capable of learning long-range dynamics of speech [5]. Besides speech synthesis, BLSTM-RNN also showed great potential in the task of voice conversion [41], and speaker adaptation in speech recognition[42]. Though DNN-based multi-speaker synthesis and speaker adaptation have been well studied, no previous work has dealt with DBLSTM-RNN in such area of TTS to the best of our knowledge.

To synthesize high-quality speech with limited database, we proposed a DBLSTM-RNN-based multi-speaker speech synthesis model. Unlike [39], we use speaker-specific features such as gender code as well as speaker code or i-vector to form an augmented input vector to SIN. SIN and SDN are trained separately but sequentially connected. The former is shared across multiple speakers and the latter is trained for a specific target speaker. SIN can be viewed as transformation from linguistic features to generalized vocoder parameters and SDN works as transformation of the vocoder parameters to those of the target speaker.

The proposed framework can also apply to speaker adaptation. Different from [40], we choose DBLSTM-RNN-based SIN as speaker independent model rather than

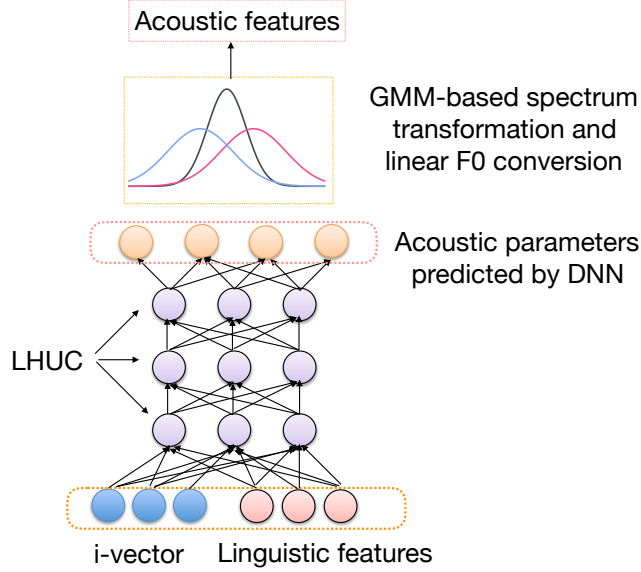


Figure 3.1: DNN-based speaker adaptation

DNN, and use SDN for feature space transformation instead of Gaussian Mixture Model (GMM).

3.2 Related work

3.2.1 DNN-based speaker adaptation

Wu et al. proposed DNN-based speaker adaptation with a combination of augmented i-vector, learning hidden unit contribution (LHUC) and GMM-based feature space transformation. I-vector is used as speaker identity labels. Compared with general DNN, LHUC which rescales the amplitudes of the hidden units in the model without actually modifying their feature receptors can achieve better performance when applied to unseen data. The experimental results confirmed the flexibility of DNN-based synthesis, also demonstrated that DNN based adaptation can achieve even better performance than HMM-based adaptation. It also claimed that feature transformation at the output layer works well and the adaptation performance can be improved by combining with model based adaptation in this work the LHUC. The framework is shown in Fig. 3.1.

3.2.2 BLSTM-RNN for speech synthesis

Deep BLSTM-RNN is established by stacking multiple BLSTM-RNN hidden layers on top of each other. BLSTM-RNN is the integration of deep bidirectional RNN and LSTM. For bidirectional RNN, given an input sequence $x = (x_1, \dots, x_T)$, the forward sequence $\vec{h} = (\vec{h}_1, \dots, \vec{h}_T)$, the backward sequence $\overleftarrow{h} = (\overleftarrow{h}_1, \dots, \overleftarrow{h}_T)$ and output vector $y = (y_1, \dots, y_T)$ can be computed from $t = 1$ to T as follows:

$$\vec{h}_t = \mathcal{H}(W_{f\vec{h}}x_t + W_{\vec{h}\vec{h}}\vec{h}_{t-1} + b_{\vec{h}}), \quad (3.1)$$

$$\overleftarrow{h}_t = \mathcal{H}(W_{f\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}), \quad (3.2)$$

$$y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y. \quad (3.3)$$

$\mathcal{H}(\cdot)$ is usually a sigmoid or hyperbolic tangent function, which leads to the limitation of inability to learn long-range relations of input sequences. An LSTM architecture, which uses purpose-built memory cells to store information, can overcome this problem and model signals that have a mixture of low and high frequency components [43]. For LSTM, $\mathcal{H}(\cdot)$ is implemented by the following functions:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (3.4)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (3.5)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (3.6)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \quad (3.7)$$

$$h_t = o_t \tanh(c_t) \quad (3.8)$$

$\sigma(\cdot)$ is the logistic function, and i , f , o and c are the input gate, forget gate, output gate and cell memory, respectively.

The framework of deep BLSTM-RNN based TTS synthesis for specific speaker proposed by [5] is shown in Fig. 3.2. Linguistic features of rich contexts are used as input vectors. The output vectors are vocoder parameters like spectral envelope and fundamental frequency. Input features and output features are time-aligned in frame level. The configurations of model training for DBLSTM-RNN is a hybrid structure, where bottom layers are feed-forward structure with sigmoid activation functions, while the top layers are BLSTM-RNN structure.

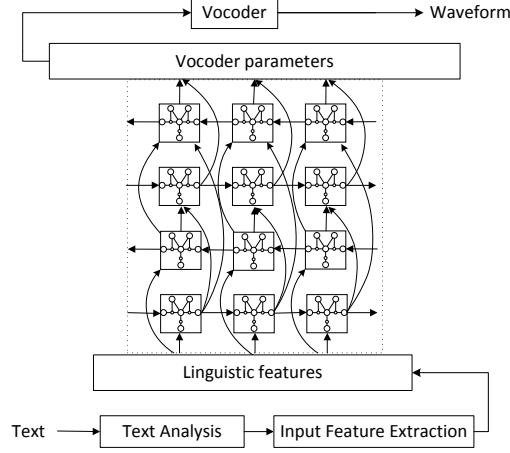


Figure 3.2: DBLSTM-RNN based single speaker speech synthesis model

3.3 BLSTM-RNN-based multi-speaker modeling and adaptation

In this section, we mainly introduce the DBLSTM-RNN-based multi-speaker synthesis model and speaker adaptation after multi-speaker synthesis. This model is expected to benefit each speaker’s synthesis and new speaker’s adaptation with augmented input vector, shared SIN and individual SDN.

3.3.1 BLSTM-RNN-based multi-speaker speech synthesis

The schematic diagram of our proposed method is shown in Fig. 3.3. The basic idea of our scheme can be considered as a combination of traditional DBLSTM-RNN-based speech synthesis and voice conversion. Our proposed framework contains both SIN and SDN. SIN is trained by all the speakers and SDN is learned with the target speaker’s data only. Since DNN-based multi-speaker speech synthesis described in [39] has experimentally shown that the knowledge of multi-speakers conveyed by shared hidden layers of DNNs can benefit each speaker, SIN is expected for the same effect.

Compared with the conventional TTS mentioned in [5], SIN of our proposed method is trained with all speakers’ data simultaneously and takes an augmented vector as input. As is shown in Fig. 3.4 which contains linguistic features and speaker-specific features such as gender code as well as i-vector. Gender code is a mark for male and female.

I-vector is an alternative as code of a speaker and it is a low-dimensional vector estimated from the supervector of that speaker. According to [44], by factor analysis,

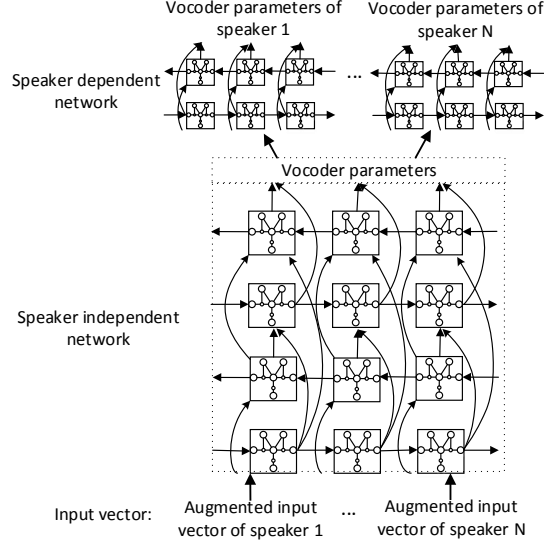


Figure 3.3: DBLSTM-RNN based multi-speaker speech synthesis model

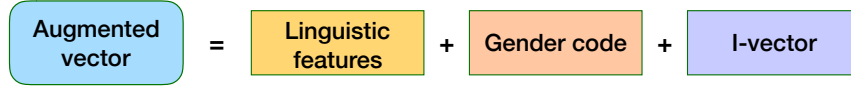


Figure 3.4: Augmented input vector for speaker independent model

that speaker's supervector M is approximated as

$$\mathbf{M} \approx \mathbf{m} + \mathbf{T}\mathbf{w}, \quad (\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})), \quad (3.9)$$

where m denotes the speaker-independent super vector, which can be extracted easily from the universal background model, often obtained as GMM. T is a total variability matrix and w is a weight vector, called i-vector, and it contains the information of speaker identity. I-vector is widely used to perform speaker adaptation in DNN-based speech recognition.

The generalized vocoder parameters produced by well-trained SIN by using a specific speaker's input vectors, are prepared as input of that speaker's SDN. The training of SDN can be viewed as a conversion process between the vocoder parameters produced by SIN and true vocoder parameters extracted from target speaker. It is optimized with specific speaker's data, and supposed to help refining the similarity of synthesized speech. The basic architecture of our scheme can be considered as decomposed BLSTM-based multi-task learning.

3.3.2 Speaker adaptation after multi-speaker synthesis

Speaker adaptation after multi-speaker synthesis is performed in three steps. Firstly, linguistic features and speaker-specific codes of the new speaker are both prepared. Since augmented vectors are used as input, the hidden layers of SIN may have learned speaker-specific information as well as speaker-independent information. Then with a small amount of data of the new speaker, the SIN is updated. However, due to limited data size, resemblance of the output of SIN to the new speaker is not sufficient. Consequently, we prepare a SDN for that new speaker, which is trained by using that speaker’s data. It is expected that the SDN can enhance the speaker’s identity of the final output from our proposed framework. We can even keep the SIN unchanged and only use the output of SIN to train a SDN for that new speaker.

3.4 Experiments

In our experiments, the data we used is the CMU ARCTIC corpus [14], which contains 7 speakers (5 males, and 2 females). We choose 900 utterances from 4 speakers (2 males and 2 females) respectively as training dataset and 100 utterances as validating set. and the utterances are common among the four speakers. 50 utterances with the same transcriptions are chosen as the testing database for each speaker, and the transcriptions of these utterances are never covered in the training or validating set. We use other two male speakers as new speakers.

All the wav files are converted into 16KHz sampling and mono channel raw files, windowed by 25ms. The frame shift is 5ms. For the speech data and their transcriptions, state-level phonemic alignment is done with the help of HTS [45]. Linguistic features are extracted and converted to 331 dimensional binary vectors. State indexes and frame indexes are also attached. Acoustic features including logF0, 25-dimensional mel-cepstral coefficients, gain and their dynamic counterparts are extracted with the help of SPTK [46]. Linear interpolation of F0 is done over unvoiced segments. 80% of silence frames are removed from training data to reduce computation cost.

To extract i-vectors, two gender-dependent UBMs and total variability matrices are trained using the EM algorithm, for male and female respectively. The input speech is first classified by the speaker’s gender and then used the corresponding UBM and T matrices to extract i-vectors. The dimensions of i-vectors are set to 30.

Prior to training, both input and target features are normalized to zero mean and unity variance. Implementation of the network training is done with the help of a

machine learning library “CURRENTT” [47]. SIN has 4 hidden layers with 256 nodes per layer, 2 lower hidden layers are feed-forward layers and 2 upper hidden layers are BLSTM-RNN layers. We test SDN with two kinds of neural networks: one is the feed-forward network, the other is BLSTM-RNN-based networks. For multi-speaker synthesis, SDN has 4 hidden layers. DNN-based SDN has 500 nodes per layer, and BLSTM-RNN based SDN has 200 nodes per layer. For a new speaker’s adaptation, we use a smaller speaker dependent network. We use SDN with 2 hidden layers with 50 nodes per layer for RNN-based SDN and 100 nodes per layer for DNN-based one. The learning rate is set to $1e - 6$ and the momentum is set to 0.9. The maximum number of iteration is set to 300 and the training is stopped if no improvement observed within the latest 50 iterations. For speech synthesis, we use a sinusoidal synthesizer to generate waveforms.

We conduct six groups of experiments by utilizing BLSTM-RNN-based speaker independent model:

1. BASELINE: Using BLSTM-RNN-based speaker independent network for multi-speaker speech synthesis.
2. SIN_DNN: Using both speaker independent network and speaker dependent network. Speaker dependent network is composed of feed-forward layers.
3. SIN_RNN: Both SIN and SDN are composed of BLSTM-RNN layers.
4. SIN^I: SIN is trained with gender code and i-vector. No speaker dependent network.
5. SIN^I_DNN: SIN is trained with gender code and i-vector. Speaker dependent network is composed of feed-forward layers.
6. SIN^I_RNN: SIN is trained with gender code and i-vector. Speaker dependent network is composed of BLSTM-RNN layers.

3.5 Evaluation and discussion

Objective evaluations aim at assessing observed distortions between natural speech and its corresponding synthesized speech. Objective measures used in this chapter are Mel-cepstral Distortion (MCD), F0 distortion in the root mean squared error (RMSE) and voiced/unvoiced (V/UV) swapping errors. Even though objective evaluation results might not correlate well with perceived speech quality and speaker similarity,

they are still helpful to analyze the performances of systems that are tested experimentally.

Subjective evaluations aim at assessing perceptual preferences of synthesized voices. Mean Opinion Score (MOS) is one of the most popular measures for assessing the overall quality of synthesized speech. And preference tests are also used here to select the closest voice to the target speaker. We performed an MOS test to evaluate the seven systems. 20 utterances of the four training speakers were generated by all the systems and they were listened to by 10 native speakers of English. The listeners were asked to give their evaluation scores in terms of overall speech quality using a 5-point scale. We also conducted ABX preference tests for speaker similarity. Here, A and B stand for synthesized speech samples after adaptation and they are generated by two different systems. X represents the target speaker’s natural voice. 10 native speakers of English were asked to select A or B based on their perceptual similarity to X in terms of speaker identity. If no difference is perceived, the listeners were asked to select the other option, that is neutral.

3.5.1 Evaluations for speech synthesis

For multi-speaker speech synthesis, the four speakers’ data are used to train SIN simultaneously. Then input vectors of each speaker are put through well-trained SIN, and the corresponding output of SIN are used as input to each SDN. The outputs of four well-trained SDN are converted into speeches for evaluation.

Table 3.1 and Table 3.2 shows the average objective evaluation results for female and male speakers respectively. The distortion of F0 and MGC of baseline system for both male and female systems are much higher than other systems, which suggests that a shared speaker independent network cannot distinguish speaker identity by itself. But when SIN combined with either i-vector, or speaker dependent network, the distortion can be dramatically decreased. This denotes that both i-vector or speaker dependent network can help to distinguish different speakers’ identity. I-vector controls the speaker identity during speaker adaptive training, while the SDN controls speaker identity after speaker adaptive training. We also can see that BLSTM-RNN-based speaker dependent networks can achieve less distortion than DNN-based systems.

Across all the systems, SIN^I achieves the lowest BAP error and the proposed system SIN^I_RNN achieves the lowest MCD, F0 RMSE and V/UV error. Interestingly, females speakers could achieve much lower cepstrum distortion than male speakers while the F0 error is obviously higher.

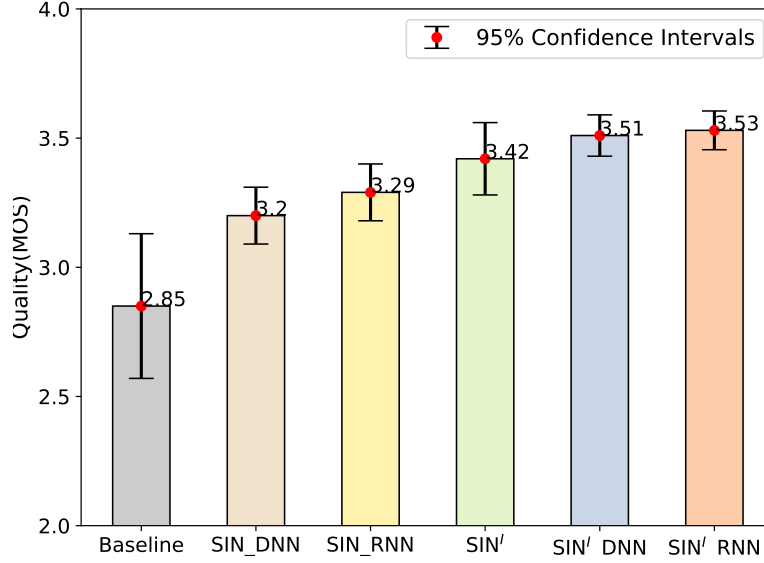


Figure 3.5: Subjective comparison for multi-speaker speech synthesis in terms of speech quality averaged across all speakers

The MOS scores averaged over all the training speakers and all the listeners are shown in Fig. 3.5. From the figure, we can find that $\text{SIN}^I_{\text{RNN}}$ achieved the highest MOS score and $\text{SIN}^I_{\text{DNN}}$ obtained a similar score, although the difference between them is not significant. There is also an obvious phenomenon that speaker dependent network with system SIN could obviously improve the synthesized speech quality. But when SDN is applied to SIN^I , the improvement is not so much obvious. This may indicate that if speaker identity can be fully controlled during speaker adaptive training, speaker dependent network would be less necessary.

The preference test results in terms of speaker identity are shown in Fig. 3.6. Speaker identity network with i-vector can achieve higher preference than the baseline. Also, $\text{SIN}^I_{\text{RNN}}$ is preferred than $\text{SIN}^I_{\text{DNN}}$.

Both objective and subjective results indicate that DBLSTM-RNN-based multi-speaker TTS with i-vector and speaker dependent network can outperform baseline in a certain degree, and also confirm the necessity of both speaker-specific features for SIN training and SDN for target speaker’s vocoder parameter refining. In work [48], it shows that i-vector with speaker independent DNN does not work well as expected when combined with feature transformation. However, in our experiment, we can find that speaker dependent training can help improve the performance.

Table 3.1: Objective distortion in terms of speech synthesis. This table is the average loss across all **female speakers**

Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP dB	V/UV err(%)
SIN(Baseline)	5.67	31.40	3.89	4.39
SIN_DNN	5.31	24.79	3.66	4.73
SIN_RNN	4.89	20.66	3.66	3.89
SIN ^I	5.11	20.15	3.69	3.96
SIN ^I _DNN	4.94	18.71	3.34	3.68
SIN ^I _RNN	4.63	17.57	3.38	3.32

Table 3.2: Objective distortion in terms of speech synthesis. This table is the average loss across all **male speakers**

Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP dB	V/UV err(%)
SIN(Baseline)	5.52	18.33	3.89	7.24
SIN_DNN	5.31	17.79	3.86	6.92
SIN_RNN	5.22	14.27	3.28	7.01
SIN ^I	5.26	13.85	3.30	6.83
SIN ^I _DNN	5.11	13.71	3.37	6.30
SIN ^I _RNN	4.93	13.05	3.39	6.46

SIN ^I 60.7%	Neural 22.1%	Baseline 17.2%
SIN ^I _DNN 42.9%	Neural 24.5%	SIN ^I 32.6%
SIN ^I _RNN 38.4%	Neural 32.2%	SIN ^I _DNN 29.4%

Figure 3.6: Subjective comparison for multi-speaker speech synthesis in terms of similarity. The p -value of the three pairs are $< 1.2e - 16$, 0.27 and 0.41 respectively.

Table 3.3: Objective comparison in terms of speaker adaptation. Using **100 utterances** of a new speaker for adaptation training.

Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP dB	V/UV err(%)
SIN(Baseline)	5.14	20.90	3.35	7.05
SIN_DNN	5.10	21.79	3.36	7.14
SIN_RNN	5.05	20.62	3.44	6.89
SIN ^I	5.21	23.04	3.31	7.06
SIN ^I _DNN	5.11	22.72	3.48	7.18
SIN ^I _RNN	5.03	19.57	3.58	6.66

Table 3.4: Objective comparison in terms of speaker adaptation. Using **30 utterances** of a new speaker for adaptation training.

Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP dB	V/UV err(%)
SIN(Baseline)	5.70	23.56	3.95	7.21
SIN_DNN	5.85	24.79	3.63	7.14
SIN_RNN	5.86	24.48	3.53	7.07
SIN ^I	5.68	23.23	3.32	7.70
SIN ^I _DNN	5.61	24.94	3.41	7.75
SIN ^I _RNN	5.74	22.67	3.62	7.66

3.5.2 Evaluations for speaker adaptation

We used two new male speakers for testing our proposed adaptation approach and conducted the tests with different amounts of adaptation utterances, 100 and 30 respectively. The adaptation data are used to update the SIN in a small number of epochs (e.g. 10 epochs in our experiments) firstly and then constructed their own SDN.

The average distortions over a new speaker are shown in Table 3.4 and Table 3.3 for 100 and 30 training utterances respectively. . We can see that distortions are reduced to a large degree with more adaptation data. When using 100 utterances for adaptation, SIN^I_RNN can achieve lowest MCD and F0 RMSE. When using much very few utterances, the RNN-based speaker dependent network tend to be easily over-fitted. From Table 3.3, we can find that RNN-based speaker dependent network achieves higher distortion in MCD than DNN-based one and also those without speaker dependent network.

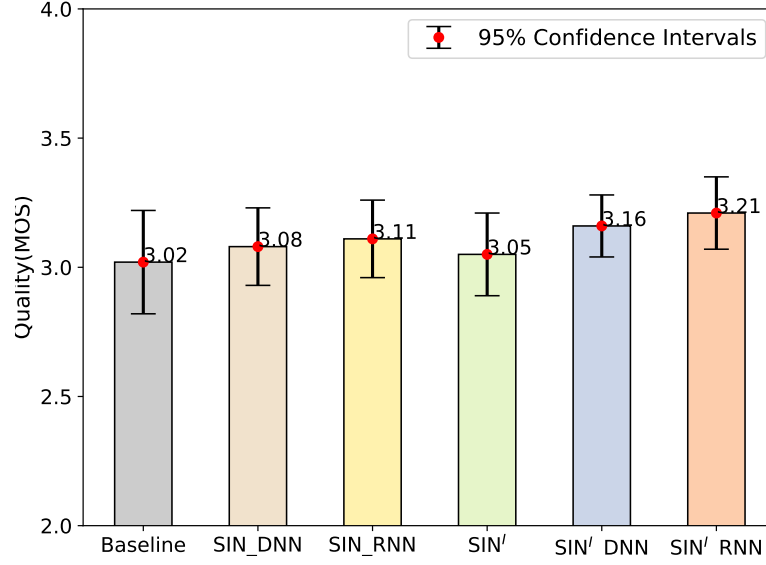


Figure 3.7: Subjective comparison for speaker adaptation in terms of speech quality averaged across all speakers

SIN'	Neural	Baseline
33.2%	37.9%	29.1%
SIN' DNN	Neural	SIN'
48.6%	30.2%	21.2%
SIN' RNN	Neural	SIN' DNN
41.1%	30.6%	28.3%

Figure 3.8: Subjective comparison for speaker adaptation in terms of similarity. The p -value of the three pairs are 0.89, 0.31 and 0.45 respectively.

The MOS score for speaker adaptation using 100 utterances is shown in Fig. 3.7. The synthesized speech quality of baseline system is very close to SIN^I. The speaker dependent network could slightly improve the performance. SIN^I_RNN achieves the highest evaluation. Fig. 3.8 lists out three groups of preference tests' results under 100 adaptation utterances condition. We can find that the speaker dependent network could also help to improve the speaker similarity.

3.6 Conclusions

In this chapter, we have proposed a DBLSTM-RNN based speech synthesis model which performs multi-speaker speech synthesis in three levels. Firstly, augmented input vectors with speaker identity parameters are prepared. Secondly, SIN is used

to perform transformation from linguistic features to general vocoder parameters. At last, SDN is used for accurate mapping between the general vocoder parameters and target speakers. This model can also apply to speaker adaptation. Objective and subjective evaluation results show that our scheme can achieve good speech quality and good speaker identity with reasonably less data.

Chapter 4

Speaker Representations for Multi-speaker Synthesis and Adaptation

Experimental results show that the speaker representations input to the first layer of acoustic model can effectively control speaker identity during speaker adaptive training, thus improving the synthesized speech quality of speakers included in training phase. For speaker adaptation, speaker code estimated from MFCCs can achieve higher preference than other speaker representations.

4.1 Introduction

Compared with unit-selection and concatenative approaches, Statistical Parametric Speech Synthesis (SPSS) is preferred because it can generate natural sounding synthetic speech with rather small corpus and vary speaker identity and speaking styles flexibly. Recently, Deep Neural Network (DNN) has significantly advanced the performance of SPSS. However, it still suffers from necessity of a large recording corpus of one speaker to train a high quality acoustic model [4–6, 27, 39, 49]. Meanwhile, significant efforts have been made to generate a new speaker’s voices with only a few utterances [39, 40, 49, 50].

Speaker adaptive training is one of the most effective approaches to train a high quality acoustic model with a limited database [39, 49, 51, 52]. In speaker adaptive training, acoustic model is jointly trained utilizing multiple speakers’ data. For DNN, it has been experimentally proved that the shared hidden layers can benefit synthesized speech of each speaker from the knowledge of others [39, 49]. Speaker adaptation has been developed for generating an arbitrary speaker’s voice with min-

imum adaptation data, and the adaptation process is usually performed on a well trained acoustic model [40, 51, 53]. However, both speaker adaptive training and speaker adaptation techniques need to control speaker identity precisely.

Generally, there are three ways to control the speaker identity in a DNN-based acoustic model. The first way is to control the speaker identity at the input layer, such as adding speaker information as auxiliary input features [40, 50]. The second one is to control the speaker identity with specially designed hidden layers, such as learning hidden unit contribution (LHUC) [40]. And the last one is to control the speaker identity near the output layer space, such as speaker dependent regression or feature space transformation [39, 40, 49].

In our work, we mainly focus on speaker identity control at the input layer. Augmented speaker identity vectors are prepared independently from linguistic features and these vectors can distinguish different speakers very well even if the speakers share the same linguistic labels. In addition to i-vector, speaker code is another speaker representation which has been widely used in DNN-based speaker adaptation in automatic speech recognition [54–57]. In ASR, a new speaker’s code is usually estimated from the training speakers in a backpropagation manner [56, 57]. In SPSS, the estimation of a new speaker’s code remains to be solved.

In this chapter, we conduct an exploring and comparative study on the controllability of different speaker identity representations when they are used in augmented inputs for speaker adaptive training and speaker adaptation in the same framework. Instead of DNN, Bidirectional Long Short-Term Memory with Recurrent Neural Network (BLSTM-RNN) acoustic model is employed due to its strong capability of learning long-range dynamics of speech as well as variation in speaker identity. We first briefly describe the framework of multiple speaker BLSTM-RNN-based speech synthesis. Then we introduce different speaker identity representations including i-vector and speaker code. We also propose two approaches especially to estimate a new speaker’s code. Analytical experiments are done to examine the performance of each speaker representation in both speaker adaptive training and speaker adaptation.

4.2 Framework for multi-speaker speech synthesis and adaptation

In this section, we mainly introduce our experimental framework used for speaker adaptive training and speaker adaptation. In order to examine the performance of

different speaker representations when they are input to the first layer, only conventional BLSTM-RNN acoustic model is adopted, with no specially designed layer or feature space mapping as post-processing. The schematic diagram of our framework is shown in Fig.4.1.

In this work, we utilize a hybrid network structure which includes both feedforward and BLSTM-RNN layers in acoustic model. The feedforward layer, trained with a back-propagation learning algorithm [58], is widely used in many practical applications. But the assumption of sample independence results in limited ability of modeling context information as well as acoustic signals [5, 59]. Bidirectional RNN can access both the preceding and succeeding input contexts with two separate hidden layers. An LSTM architecture, can overcome the gradient vanishing problem that prevents RNN from modeling long-span relations in both linguistic and acoustic domains. Deep bidirectional LSTM-RNN is able to build up progressively higher level representations of input data, which is a crucial factor of the recent success of hybrid systems [59].

Different from the conventional individual speaker’s synthesis, the network is trained with multiple speakers’ data, and it takes linguistic features as input as well as speaker-specific features, such as speaker code or i-vector. With augmented speaker identity representations, inputs with the same linguistic content but spoken by different speakers can be distinguished. Different from Fan’s work in [39], not only hidden layers but also the output layer is shared across all the speakers.

During the speaker adaptive training phase, it is very crucial to train the network for all the speakers simultaneously, which means that each batch should consider the data from all the speakers during the stochastic gradient decent (SGD) procedure, and training data also needs to be shuffled across all the speakers. Since BLSTM-RNN can take use of both past and future information, it can capture the dynamics in speeches as well as speaker identity.

In synthesis, the speeches of any speaker who had ever joined in speaker adaptive training can be synthesized through the well-trained multi-speaker model with the help of speaker specific representation. In speaker adaptation phase, a new speaker’s representation is firstly estimated, then appended to linguistic features. The well-trained multi-speaker model is updated with the new speaker’s data. The error of new training/adaptation samples are back-propagated to the whole network. The new speaker’s speech can be generated with the well-updated model.

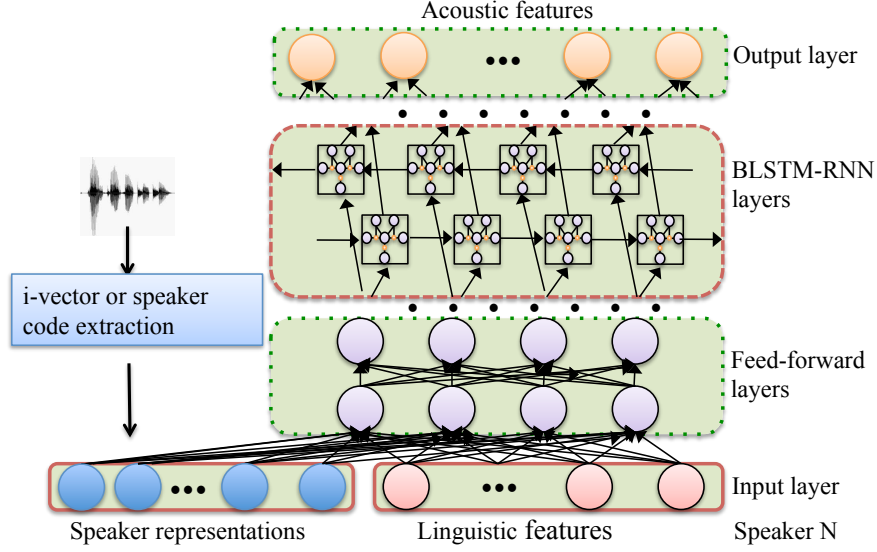


Figure 4.1: BLSTM-RNN based multi-speaker speech synthesis model

4.3 General speaker identity representations

This section mainly introduces different speaker identity representations including i-vector and speaker code. After that, we propose two methods for speaker code estimation.

4.3.1 I-vector

I-vector has been widely used in both speaker and speech recognition. It is a low-dimensional vector, the cosine distance between two different i-vectors represents the difference of two speakers: the smaller the distance is, the closer the speakers are, and vice versa. According to [44], by factor analysis, a speaker's supervector M is approximated as

$$\mathbf{M} \approx \mathbf{m} + \mathbf{T}\mathbf{w}, \quad (\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})), \quad (4.1)$$

where \mathbf{m} denotes the speaker-independent supervector, which can be extracted easily from the universal background model (UBM), often obtained as GMM. \mathbf{T} is a total variability matrix and \mathbf{w} is a weight vector, so called i-vector.

Since the robust estimation of the total variability matrix \mathbf{T} requires a large amount of data, additional data need to be collected while the number of speakers are limited.

4.3.2 Speaker Code

Speaker code has achieved promising results in speaker adaptation [56] of automatic speech recognition task. As described in [54], if there are K speakers' corpora for model training, we can simply use $z_c = (z_{1,c}, z_{2,c}, \dots, z_{k,c}, \dots, z_{K,c})$ to represent the c -th speaker's code. $z_{k,c}$ is defined as follows:

$$z_{k,c} = \begin{cases} 1 & (k = c), \\ 0 & (k \neq c). \end{cases} \quad (4.2)$$

Although there are many works related to estimate a new speaker's code in ASR, but no work has been reported in SPSS area to the best of our knowledge.

4.4 Independently estimated discriminating code

4.4.1 Speaker code estimation from i-vector

Usually, a speaker's i-vector is estimated from his/her utterance-based i-vectors. Those utterance-based i-vectors are able to capture the nuances in speaking style of different utterances, even these utterances are spoken by the same speaker. By collecting utterance-based i-vectors from a single speaker, we can estimate the i-vectors' distribution of that speaker, which is assumed to follow the single Gaussian distribution. If we have K speakers and their corresponding K Gaussian distributions, we can define K -mixture GMM with even weights, assuming that prior probability of each speaker is the same:

$$p(\mathbf{x}) = \sum_{k=1}^K \frac{1}{K} \mathcal{N}(\mathbf{x} | \mu_{\mathbf{k}}, \Sigma_{\mathbf{k}}) \quad (4.3)$$

In equation (4.3), \mathbf{x} represents utterance-level i-vectors, $\mu_{\mathbf{k}}$ and $\Sigma_{\mathbf{k}}$ are the mean and variance of utterance-based i-vectors of k -th speaker. The speaker code of a new speaker is estimated as a vector that composed of per-mixture posterior possibility of this K mixture GMM model. Given his/her utterance-based i-vector \mathbf{x}_c , the k -th component of the speaker code γ_k is calculated as:

$$\gamma_k = p(k | \mathbf{x}_c) = \frac{\frac{1}{K} \mathcal{N}(\mathbf{x}_c | \mu_{\mathbf{k}}, \Sigma_{\mathbf{k}})}{\sum_{j=1}^K \frac{1}{K} \mathcal{N}(\mathbf{x}_c | \mu_{\mathbf{j}}, \Sigma_{\mathbf{j}})} \quad (4.4)$$

The new speaker code can be represented as:

$$\tilde{z}_c = (\gamma_1, \gamma_2, \dots, \gamma_K) \quad (4.5)$$

4.4.2 Speaker code estimation from MFCCs

Mel-frequency cepstral coefficients (MFCCs) include various kinds of information not only linguistic but also non-linguistic such as speaker identity. Since speaker identity was not drastically changed in an utterance, models which are able to capture the information lying on the long time span are preferable. Hence, direct estimation of speaker code from MFCCs sequences utilizing BLSTM-RNN is investigated in this chapter. The input of the network is a gender mark of a speaker and his/her MFCCs for each frame, and the output is always the speaker code of that speaker. This network need to be jointly trained with different speakers' parameters. To get a better classification, a softmax output layer is employed.

4.5 Experimental setup

In our experiments, the data we used is the CMU ARCTIC corpora [14], which contain 7 speakers (5 males, and 2 females) with parallel sentences. For speaker adaptive training, we chose 900 utterances from each of 4 US English speakers (2 males and 2 females) as training dataset, 100 utterances as validation set, and 50 utterances as testing set. The sentences were common among the four speakers. For speaker adaptation, the other three speakers were treated as the new target speakers. For each speaker, we prepared two training sets, one set had 100 utterances and the other had 30 utterances. 10 sentences were used for validation and 50 utterances were used for testing. For both speaker adaptive training and speaker adaptation, transcriptions of testing sets were common and never covered in training or validation set.

All the wav files were converted into 16KHz sampling raw files, windowed by 25ms. The frame shift is 5ms. Linguistic features were extracted and converted to 307 dimensional vectors. State index and frame index were also attached. State-level alignment was done with the help of HTS [45]. In speaker adaptive training with speaker representations, gender mark as well as i-vector or speaker code were appended to each frame of the linguistic features. Acoustic features including 39-dimensional mel-cepstral coefficients, F0 in log-scale, 26-dimensional Band-a-periodicity parameters (BAP), and their delta and delta-delta features were extracted with the help of STRAIGHT [60]. A binary value for voiced/unvoiced decision was also attached. Linear interpolation of F0 was done over unvoiced segments.

The neural network of speaker independent acoustic model had four hidden layers, including 2 feedforward layers and 2 BLSTM-RNN layers, and each of them had 300

nodes. To train the acoustic model, both input and target features were normalized to zero mean and unit variance. The learning rate was set to $1e-5$ and the momentum was set to 0.6. The training was stopped if no improvement was observed within the latest 20 iterations. For speaker adaptation, the adaptation data were used to update the well-trained speaker independent model until no improvement was observed within the latest 20 iterations. Implementation of the network training was done with the help of a machine learning library “CURRENNT” [47].

STRAIGHT was employed to synthesize waveforms from predicted acoustic parameters. Before waveform generation, global variances were used with Maximum Likelihood Parameter Generation (MLPG) algorithm to enhance the dynamic properties of synthetic speech. To calculate the global variances, the variance of each sentence’s acoustic features was built as a single GMM.

To extract i-vectors, a gender independent 2048-mixture UBM and 30-dimension total variability matrices were trained with the EM algorithm, using NIST SRE corpora (2004, 2006, 2008), Switchboard II Phase 1/2/3, Switchboard Cellular I/II and the CMU ARCTIC corpora [14, 61–65]. Then the i-vectors of the input speeches were extracted using the UBM and T matrices.

Speaker codes of the 4 US speakers were prepared according to equation (4.2). To estimate a new speaker’s code using i-vector, 4 mixture GMM was trained with utterance-based i-vectors of the 4 US speakers. For speaker code estimation using MFCCs, 13-dimensional MFCCs parameters were extracted. The neural network had 6 layers, including 3 feedforward layers and 3 BLSTM-RNN layers, the set for the number of nodes in each layer is [50, 200, 400, 300, 200, 100]. A softmax output layer and cross entropy objective function were utilized. The learning rate was set to $1e-4$ and the momentum was set to 0.5.

4.6 Evaluation and discussion

Objective measures used in this chapter are Mel-cepstral Distortion (MCD) [66], F0 distortion in the root mean squared error (RMSE), BAP distortion and voiced/unvoiced (V/UV) swapping errors.

As for subjective evaluation, we conducted AB preference tests to evaluate the preference in naturalness and ABX preference tests to evaluate similarity. Here, A and B stand for synthesized speech samples generated by two different systems. X represents the target speaker’s raw speech. For naturalness test, 20 English speakers were asked to select A or B which is more natural and comfortable. For similarity

test, the listeners were asked to select A or B based on their perceptual similarity to X in terms of speaker identity. If no difference is perceived, the listeners were asked to select the other option, that is neutral. In each experimental group, 20 parallel sentences are selected randomly from testing sets of each system.

In this section, SI represents the speaker independent model which is trained by multiple speakers' data but without speaker identity information. $SC_{(O)}$ stands for the original speaker code as showed in equation (4.2). $SC_{(I)}$ means speaker code estimated from i-vectors and $SC_{(M)}$ is for speaker code estimated from MFCCs.

4.6.1 Evaluations for multi-speaker speech synthesis

We use the 4 US speakers who have joined in speaker adaptive training to evaluate the quality for multi-speaker synthesis. Table 4.1 and Table 4.2 shows the average objective evaluation results over female and male speakers respectively. Individual modeling is trained with only the target speaker's data and used as baseline.

From table 4.1 and table 4.2, we can find that SI presents much higher distortion than baseline in all kinds of objective measures. But when i-vector or speaker code is attached to the input feature, it can outperform the baseline in all aspects. For the female speakers, SI+ $SC_{(O)}$ achieves best performance in F0 RMSE and V/UV error rates while SI+i-vector gives the lowest distortion in MCD and BAP. For the male speaker, SI+ $SC_{(O)}$ gives the lowest distortion in MCD, F0 and BAP while SI+i-vector shows the lowest V/UV error rates.

Subjective evaluation results are showed in Fig.4.2. SI + $SC_{(O)}$ and SI + i-vector gets much higher preference than the individually modeling while individual modeling is preferred than SI. Among all systems, SI + $SC_{(O)}$ achieves the most preference in terms of both naturalness and similarity, and we hope it can achieve good performance in speaker adaptation.

The evaluation results not only suggest that the shared hidden layers can help to improve the quality of synthesized speech, but also demonstrate that the speaker representations input to the first layer of BLSTM-RNN can control speaker identity very effectively during speaker adaptive training.

4.6.2 Evaluations for speaker adaptation

The average objective evaluation results of the new speaker's adaptation are presented in Table 4.3 and Table 4.4. Individual speech synthesis systems are trained with the same adaptation data of the new speaker. To illustrate the effect of adaptation, we

Table 4.1: Objective evaluations for **female speakers** in multi-speaker’s speech synthesis. Individual synthesis is trained by only one female speaker’s data.

Speaker	Female			
Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP (dB)	V/UV err(%)
Individual synthesis	5.45	23.68	6.87	4.37
SI	5.67	31.40	3.89	4.39
SI+i-vector	5.04	20.25	3.70	4.02
SI+SC _(O)	5.06	19.96	3.69	4.01

Table 4.2: Objective evaluations for **male speakers** in multi-speaker’s speech synthesis. Individual synthesis is trained by only one male speaker’s data.

Speaker	Male			
Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP (dB)	V/UV err(%)
Individual synthesis	7.15	15.50	3.52	7.21
SI	5.52	18.36	3.90	7.24
SI+i-vector	5.26	13.80	3.29	6.82
SI+SC _(O)	5.24	13.40	3.28	7.09

Table 4.3: Objective evaluations for new speaker’s adaptation **using 100 adaptation utterances**. Individual synthesis is trained using the same data of the target speaker.

Sentences’ number	100 sentences			
Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP (dB)	V/UV err(%)
Individual synthesis	8.19	18.16	3.86	14.1
SI	5.14	20.90	3.31	7.05
SI+i-vector	5.19	23.02	3.31	7.41
SI+SC _(I)	5.11	19.64	3.30	7.19
SI+SC _(M)	5.16	20.06	3.29	7.32

Table 4.4: Objective evaluations for new speaker’s adaptation **using 30 adaptation utterances**. Individual synthesis is trained using the same data of the target speaker.

Sentences’ number	30 sentences			
Experimental Systems	MCD (dB)	F_0 RMSE(Hz)	BAP (dB)	V/UV err(%)
Individual synthesis	9.06	20.32	4.17	22.6
SI	5.30	22.56	3.35	7.21
SI+i-vector	5.28	21.03	3.32	7.70
SI+SC _(I)	5.28	18.76	3.31	7.35
SI+SC _(M)	5.23	18.17	3.29	7.77

Table 4.5: Objective evaluations for the new speaker’s speech synthesis using 900 sentences

Experimental System	MCD (dB)	F_0 RMSE(Hz)	BAP (dB)	V/UV err(%)
Individual synthesis	7.13	15.06	3.38	7.17

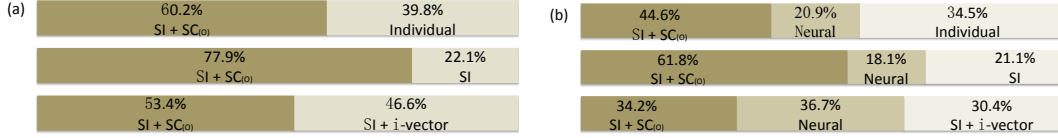


Figure 4.2: Subjective evaluations for multi-speaker synthesis in terms of naturalness (a) and similarity (b). The p -values of the three pairs in (a) are 2.4×10^{-6} , 3.6×10^{-12} and 0.41 respectively. The p -values of the three pairs in (b) are 0.02, 2.6×10^{-6} and 0.33 respectively.

also trained an individual synthesis system with 900 sentences of the target speaker and the distortions are showed in Table 4.5.

According to Table 4.3 and Table 4.4, distortions of adapted speeches are much lower than the distortion of individual synthesis, which suggests the importance of model initialization. For speaker adaptation, the neural network is updated based on a well-trained multiple speakers’ acoustic model, but for individual synthesis it is initialized randomly.

From Table 4.3 and and Table 4.4, we can find a very interesting phenomenon that the supervised adaptation based on the speaker independent model can achieve quite small distortion even without i-vector or speaker code, especially in 100 sentences’ case. This is quite different from the experimental results of multi-speaker synthesis. It may indicate that speaker information which just input to the first layer may only have very limited guidance in a supervised adaptation way.

By comparing Table 4.4, Table 4.4 and Table 4.5, it is easy to find that the adaptation system can obtain less MCD distortions than individual speech synthesis using 900 sentences although F_0 RMSE error is a little higher. Among all the systems, estimated speaker code can achieve the lowest MCD, RMSE of F_0 and BAP distortions, and SI achieves the lowest V/UV error rate.

Another interesting phenomenon is that the speaker code based adaptation always achieves a slight superior to i-vector based adaptation in terms of F_0 RMSE. This trend also happens in multi-speaker synthesis. A possible explanation is that the discrete speaker codes are more robust in F_0 prediction than i-vectors which are more continuous and have strong relationship to spectral parameters.

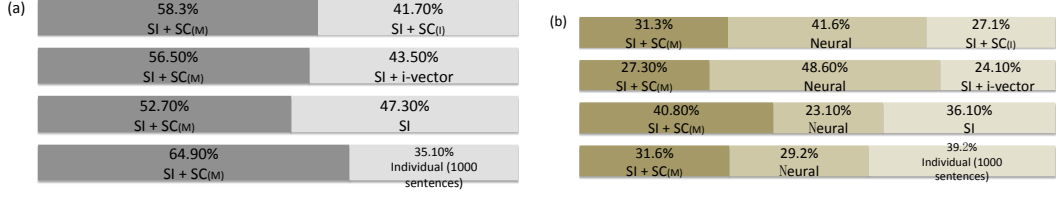


Figure 4.3: Subjective evaluations for speaker adaptation in terms of naturalness (a) and similarity (b). The p -values of the four pairs in (a) are 0.58, 0.81, 0.74 and 0.48 respectively. The p -values of the four pairs in (b) are 0.84, 0.53, 0.28 and 0.47 respectively

Subjective evaluation results are presented in Fig.4.3. SI + SC_(M) obtains better preference than SI+SC_(I) in terms of both naturalness and similarity. SI+SC_(M) also gets higher evaluation than SI+i-vector and SI. Compared with individual synthesis with 900 sentences, SI + SC_(M) achieves more preference in naturalness.

4.7 Conclusions

In this chapter, we mainly investigated the controllability of different speaker representations when they are performed at the input layer of neural networks. Experimental results showed that speaker representations input to the first layer of acoustic model can control speaker identity effectively during speaker adaptive training, but its impact on the supervised adaptation of a new speaker is limited. Further, we will explore to perform speaker identity control at different layers of neural network.

Chapter 5

Wasserstein GAN and Waveform Loss-based Acoustic Model Training Using a WaveNet Vocoder

Recent neural networks such as WaveNet and sampleRNN that learn directly from speech waveform samples have achieved very high-quality synthetic speech in terms of both naturalness and speaker similarity even in multi-speaker text-to-speech synthesis systems. Such neural networks are being used as an alternative to vocoders and hence they are often called neural vocoders. The neural vocoder uses acoustic features as local condition parameters, and these parameters need to be accurately predicted by another acoustic model. However, it is not yet clear how to train this acoustic model, which is problematic because the final quality of synthetic speech is significantly affected by the performance of the acoustic model. Significant degradation happens, especially when predicted acoustic features have mismatched characteristics compared to natural ones. In order to reduce the mismatched characteristics between natural and generated acoustic features, we propose frameworks that incorporate either a conditional generative adversarial network (GAN) or its variant, Wasserstein GAN with gradient penalty (WGAN-GP), into multi-speaker speech synthesis that uses the WaveNet vocoder. We also extend the GAN frameworks and use the discretized mixture logistic loss of a well-trained WaveNet in addition to mean squared error and adversarial losses as parts of objective functions. Experimental results show that acoustic models trained using the WGAN-GP framework using back-propagated discretized-mixture-of-logistics (DML) loss achieves the highest subjective evaluation scores in terms of both quality and speaker similarity.

5.1 Introduction

In recent years, text-to-speech (TTS) synthesis has gained popularity as an artificial intelligence technique and is widely used in many applications with speech interfaces. There are currently two major categories in the machine learning-based speech synthesis field: a) an end-to-end approach that learns the relationship between text and speech directly and b) the conventional pipeline processing approach that divides text-to-speech conversion into sub tasks such as linguistic feature extraction and acoustic feature extraction. In the latter approach, an acoustic model is trained to learn the relationship between separately extracted linguistic and acoustic features [67]. Previously investigated acoustic models include the hidden Markov model (HMM) [4], the deep neural network (DNN) [68], and the recurrent neural network (RNN) [5][6]. These are normally trained with the minimum mean squared error (MSE) criterion, and hence, the generated acoustic parameters tend to be over-smoothed regardless of the architectures. Finally, speech waveforms have been reconstructed using a deterministic vocoder based on the acoustic parameters [69][70][71]. However, the generated signals have artifacts and typically sound buzzy. Due to these two major issues, the resultant quality of generated speech sounds obviously worse compared with natural speech.

Very recently, we see emerging solutions for the two issues. To alleviate the over-smoothing problem, Saito et al. have incorporated adversarial training into acoustic modeling [72][30]. The generative adversarial network (GAN) contains a generator as well as a discriminator [73], where the generator aims at deceiving the discriminator and the discriminator is trained to distinguish the natural and generated feature samples. In the framework proposed by [30], the generator acts as an acoustic model and is optimized by not only the conventional MSE but also an adversarial loss computed using the discriminator. Experimental results show that GAN can effectively alleviate the over-smoothing effect of the generated speech parameters.

To avoid the artifacts and deterioration caused by deterministic vocoders, WaveNet, which directly models the raw waveform of the audio signal in a non-linear autoregressive way, has been proposed and dramatically improves the quality of synthetic speech [7][74]. The original WaveNet model [7] used linguistic features as well as the fundamental frequency (F0) as local conditions. Later, the WaveNet model was used as an alternative to the deterministic vocoders in many studies [75][34] by conditioning it on acoustic features such as cepstrum, F0, or spectrograms only [75],

and results have shown that the sound quality of the WaveNet vocoder outperformed deterministic vocoders and phase recovery algorithms [76].

However, it is also reported that the samples generated from WaveNet occasionally become unstable and generate collapsed speech, especially when less accurately predicted acoustic features are used as the local condition parameters [77]. This would be more critical for the case of multi-speaker acoustic modeling where the same network is used for modeling multiple speakers at the same time, as the prediction accuracy of the multi-speaker model would be worse than well-trained speaker-dependent models.

In this paper, we propose frameworks that incorporate either the conditional GAN [78] or its variant, Wasserstein GAN with gradient penalty (WGAN-GP) [79], into RNN-based speech synthesis systems using the WaveNet vocoder for the purpose of reducing the mismatched characteristics between natural and generated acoustic features and for making the outputs of the WaveNet vocoder better and more stable. We evaluate the proposed frameworks using a multi-speaker modeling task. The generator of GAN is conditioned on both linguistic features and speaker code, and the discriminator aiming at distinguishing the real and predicted mel-spectrograms is also conditioned on speaker information. The WaveNet vocoder is conditioned on both mel-spectrogram and speaker codes, as well.

In addition, we extend the GAN frameworks and define a new objective function using the weighted sum of three kinds of losses: conventional MSE loss, adversarial loss, and discretized mixture logistic loss [80] obtained through the well-trained WaveNet vocoder. Since the third loss will let neural networks consider losses not only in the acoustic feature domain (such as mel-spectrogram) but also in the final waveform, we hypothesize that it will improve the quality of synthetic speech. In our experiment, simple recurrent units (SRUs) [81] are utilized as basic components since they can be trained faster than the LSTM-based RNN architecture while maintaining a performance as good as or even better than LSTM-RNN.

In Section 2 of this paper, we briefly review previously proposed DNN-based multi-speaker speech synthesis, as we evaluate our proposed method in a popular multi-speaker modeling task. In Section 5.3, we present the proposed framework for multi-speaker speech synthesis. Section 5.4 describes the basic elements of the structure of the proposed model including SRU, GAN, and WaveNet, and the details of the training algorithms are given in Section 5.5. Section 5.6 describes experimental conditions and Section 5.7 discusses the results. We conclude in Section 5.8 with a brief summary and mention of future work.

5.2 DNN-based Multi-speaker Speech Synthesis

Although deep learning-based methods have significantly advanced the performance of statistical parametric speech synthesis (SPSS), it still suffers from the necessity of a large amount of speech recordings of one speaker to train a high-quality acoustic model. Ideally, a speech synthesis system should be able to generate an arbitrary speaker’s voice with a minimum of training data. Multi-speaker speech synthesis is one of the most effective approaches to train such a high-quality acoustic model with a limited amount of speech data of each speaker. Using multiple speakers’ data at the same time, we can improve the quality of synthesized speech and can also change the speaker characteristics of synthetic speech flexibly.

Using DNN-based acoustic models as a basis, Fan et al. [39] proposed multi-speaker speech synthesis using shared speaker-independent layers as well as a speaker-dependent output layer. They showed that the speaker-dependent output layer can be estimated from a target speaker’s data only and that the shared hidden layers can improve the quality of synthesized speech of individual speakers. Wu et al. [40] suggested using i-vectors for modeling multiple speakers and controlling the speaker identity of synthetic speech. Hojo et al. [82] proposed using speaker codes based on a one-hot vector for modeling multiple speakers and extending the code and associated weights at an input layer for adapting it to unseen speakers. Luong et al. [83] proposed estimating code vectors for new speakers via back-propagation and experimented with manually manipulating input code vectors to alter the gender and/or age characteristics of the synthesized speech. Similar work has been extended to LSTM-based acoustic models. Zhao et al. [84] examined various speaker identity representations for multi-speaker synthesis and showed that multi-speaker systems trained with less of the target speaker’s data can even outperform single speaker speech synthesis, which uses a larger amount of the target speaker’s data. Li et al. [85] investigated multi-speaker modeling with speech data in different languages.

Multi-speaker speech synthesis has also been investigated in the recent WaveNet-based approaches and in end-to-end approaches. Hayashi [86] attempted WaveNet vocoder-based multi-speaker synthesis using four speakers from the CMU arctic corpus [14]. VoiceLoop [15] involves the data of 109 speakers for acoustic model training, and Deep Voice 3 [11] trained a multi-speaker model using over 2,000 speakers. Wang et al. [16] proposed a bank of style embedding vectors and used it for modeling multiple TED speakers. As we can see, very active research on multi-speaker modeling has been carried out.

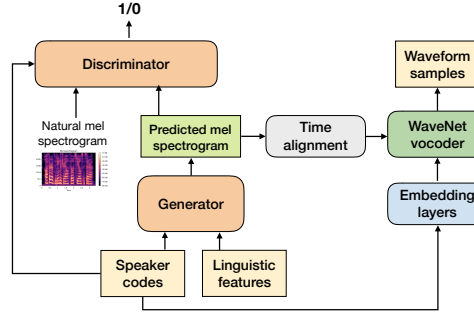


Figure 5.1: Proposed GAN-trained multi-speaker speech synthesis framework using a WaveNet vocoder.

5.3 Multi-speaker Speech Synthesis Incorporating GAN and WaveNet Vocoder

In this section, we introduce the proposed speech synthesis framework for multi-speaker modeling.

In the conventional SPSS structure, acoustic models and vocoders usually work independently: the acoustic models are trained without any consideration of the speech vocoding process, and vice versa. It was the same in the first versions of end-to-end structures such as Deep Voice [87], where vocoders were usually designed or trained on natural acoustic parameters without considering the divergence between predicted and natural acoustic parameters. This may lead to obvious and unpredictable distortion of the synthesized speech. To alleviate this problem, Tacotron 2 [34] utilized predicted mel-spectrograms to train the WaveNet vocoder instead of natural mel-spectrograms. Experimental results showed that such a strategy may outperform those that use natural parameters and may achieve a higher evaluation.

In the present work, we try to minimize the acoustic mismatch of predicted and natural parameters by conducting acoustic model training based on GAN, which also considers vocoder loss. The proposed multi-speaker speech synthesis framework is shown in Fig. 5.1. In this framework, a generator part of GAN is adopted to predict acoustic features from linguistic features, and both the generator and discriminator are conditioned on speaker codes and trained with multiple speakers’ data. Similar to Tacotron 2, the mel-spectrogram, a low-dimensional representation of the linear-frequency spectrogram, which contains both spectral envelop and harmonics information, is selected as the output of the generator and used to bridge the acoustic model and the WaveNet vocoder. Mel-scale acoustic features have overwhelming advantages in terms of emphasizing the details of audio, especially for lower frequencies, since

they are more critical to phonetic information and hence to speech intelligibility in general.

The input of the discriminator is either natural or generated acoustic feature samples. The discriminator is trained to distinguish natural samples from generated ones. Speaker codes are also attached to both the input and hidden layers of the discriminator in order to make a better distinction between different speakers. The discriminator is used to compute the adversarial (ADV) loss, which is expected to alleviate the over-smoothing problem.

In addition to the adversarial (ADV) loss from the discriminator, the average discretized-mixture-of-logistics (DML) loss of a well-trained WaveNet model is also back-propagated to the generator of GAN. This loss corresponds to distortion between natural and generated waveform samples. We hypothesize that this increases the consistency of acoustic features predicted by the acoustic model and utilized in the vocoder since the acoustic model is updated on the basis of gradients directly computed by the pre-trained WaveNet vocoder.

In brief, it is expected that the weighted sum of the conventional MSE loss, the adversarial loss of the discriminator, and the DML from the WaveNet vocoder will improve the accuracy of the predicted acoustic parameters and thus enhance synthesized speech quality. What sets this work apart from other related works is that WaveNet is involved in the process of acoustic modeling training. After extracting acoustic features from a training corpus, the WaveNet vocoder is first trained by utilizing natural mel-spectrograms, and then the trained WaveNet model is fixed and referenced for acoustic model optimization.

5.4 Components of Proposed Model Structure

In this section, we describe the three major components of the proposed framework, namely, the SRU architecture and the GAN and WaveNet models.

5.4.1 SRU

For the sake of modeling accuracy as well as time efficiency, we choose SRU [81] as the basic architecture of the acoustic modeling. The SRU architecture was originally designed to speed up the training process of RNN. By utilizing both skip and highway connections, SRU is capable of outperforming RNN, especially on very deep networks. Compared with other recurrent architectures (e.g., LSTM and GRU), the basic form of SRU includes only a single forget gate f_t to alleviate vanishing and exploding

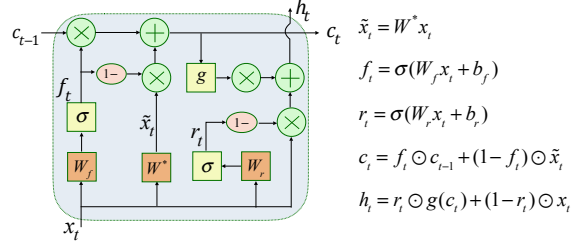


Figure 5.2: Details of the SRU cell. $\sigma(\cdot)$ and $g(\cdot)$ represent sigmoid and ReLU activation functions, respectively.

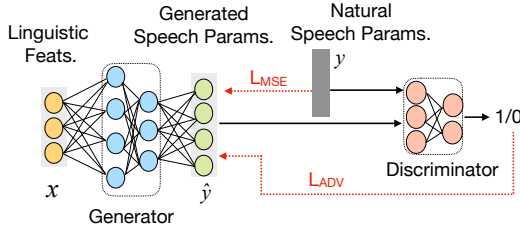


Figure 5.3: GAN-based training of TTS acoustic model. L_{ADV} indicates adversarial loss and L_{MSE} indicates L2 loss.

gradient problems instead of using many different gates to control the information flow. In SRU, the forget gate is used to modulate the internal state c_t , which is then used to compute the output state h_t . Unlike existing RNN architectures that use the previous output state in the recurrence computation, SRU completely drops the connection between the gating computations and the previous states, and this makes SRU computationally efficient and allows us to use parallelization. The complete architecture of SRU is shown in Fig. 5.2. The reset gate r_t is computed similar to the forget gate f_t and is used to compute the output state h_t , which performs as a combination of the internal state $g(c_t)$ and the input x_t . $g(\cdot)$ represents a ReLU activation function and $\sigma(\cdot)$ is a sigmoid function.

5.4.2 Generative Adversarial Network

GANs have achieved great success in modeling the distributions of complex data and the predictions of realistic data in many applications. They have also proven beneficial for speaker-dependent speech synthesis [30].

Fig. 5.3 shows the GAN-based training of acoustic models for TTS systems. The

GAN training involves a pair of networks: a generator G aims to produce vivid feature samples that deceive a discriminator D , and the discriminator aims to estimate the probability that a sample y came from the real data set distribution \mathbb{P}_r rather than a generator distribution \mathbb{P}_g . For speech synthesis from text, the generator is conditioned on linguistic vectors $x \sim \mathbb{P}_x$. The generator and discriminator are trained like a two-player min-max game objective function, as

$$\min_G \max_D \mathbb{E}_{y \sim \mathbb{P}_r} [\log D(y)] + \mathbb{E}_{x \sim \mathbb{P}_x} [\log(1 - D(G(x)))] \quad (5.1)$$

This objective function is not easy to optimize. To improve the stability of model training, Wasserstein GAN (WGAN), which minimizes a different distribution divergence called *Earth - Mover* (EM) or Wasserstein-1 distance, has been proposed and achieved a better performance than original GAN in terms of convergence, especially in image processing [88]. The optimization criteria for WGAN is equal to

$$\min_G \max_D \mathbb{E}_{y \sim \mathbb{P}_r} [D(y)] - \mathbb{E}_{x \sim \mathbb{P}_x} [D(G(x))] \quad (5.2)$$

During the training of WGAN, the updated model parameters of discriminator are clipped into a compact space $[-c, c]$ to enforce a Lipschitz constraint on D . However, the weight clipping may lead to either vanishing or exploding gradients if the clipping threshold c is not carefully tuned, and the resulting discriminator may have a pathological value surface even when optimization performs smoothly [79]. To address this problem, Gulrajani et al. [79] proposed penalizing the norm of the gradient deduced from a discriminator with respect to its input. The new objective for WGAN with gradient penalty (WGAN-GP) is shown as follows:

$$\begin{aligned} \min_G \max_D \mathbb{E}_{y \sim \mathbb{P}_r} [D(y)] - \mathbb{E}_{x \sim \mathbb{P}_x} [D(G(x))] \\ + \lambda \mathbb{E}_{\tilde{y} \sim \mathbb{P}_{\tilde{y}}} [(\|\nabla_{\tilde{y}} D(\tilde{y})\|_2 - 1)^2] \end{aligned} \quad (5.3)$$

where \tilde{y} represents samples that are linearly interpolated by the real data y and the fake data generated from the generator $G(x)$:

$$\tilde{y} = \epsilon y + (1 - \epsilon)G(x) \quad (5.4)$$

where ϵ is a random number that obeys distribution $U[0, 1]$.

The loss function of the generator is also expanded on the basis of the least square errors of y as:

$$L_G(y, \hat{y}) = L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV}(\hat{y}) \quad (5.5)$$

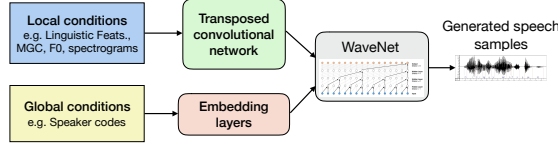


Figure 5.4: Local condition and global condition used in a WaveNet model.

where $L_{ADV}(\hat{y})$ is the adversarial loss and γ_D controls the weight of the adversarial loss. When $\gamma_D = 0$, the loss function is equivalent to the conventional MSE criteria. In original GAN, $L_{ADV}(\hat{y})$ equals $\mathbb{E}[\log(1 - D(G(x)))]$. In WGAN-GP, $L_{ADV}(\hat{y})$ can be regarded as $-\mathbb{E}[D(G(x))]$.

5.4.3 WaveNet

WaveNet is a deep auto-regressive and generative model that models a joint distribution of sequential data as a product of conditional distributions, as

$$p(s) = \prod_t p(s_t | s_{<t}, \theta) \quad (5.6)$$

where s_t is a variable of s at time t and θ denotes model parameters. The conditional distributions are usually modelled with a neural network that receives all past variables $s_{<t}$ as input and outputs a distribution over possible s_t . The neural network consists of stacked dilated causal convolution layers [7], and each causal convolutional layer can process its input in parallel, making these architectures very fast to train compared to RNNs. It typically uses gated activation functions [89] along with two conditions, global and local, which is another important concept in WaveNet.

The difference between the two conditions is shown in Fig. 5.4. The global condition focuses on conditional vectors irrelevant to time, e.g. a speaker embedding in a TTS model, while the local condition deals with time-series input conditions, such as linguistic and acoustic features. The basic activation function with global conditioning is

$$h_i = \sigma(W_{g,i} * s_i + V_{g,i}^T c) \odot \tanh(W_{f,i} * s_i + V_{f,i}^T c) \quad (5.7)$$

where $*$ denotes a convolution operator and \odot denotes an element-wise multiplication operator. σ is a logistic sigmoid function. c represents a global condition. i is the layer index. f and g denote filter and gate, respectively. W and V are learnable weights. For a case where c denotes the local condition (such as mel-spectrogram),

the matrix products $V_{g,i}^T c$ and $V_{f,i}^T c$ are replaced by convolutions $V_{g,i}^T * c$ and $V_{f,i}^T * c$, respectively.

Oord et al. [7] take both linguistic and acoustic features such as F0 as the local conditions. In other studies [11, 34, 75], only acoustic features are used as the local conditions, and the WaveNet model tends to perform as a neural vocoder. In the proposed framework, WaveNet is used as a multi-speaker neural vocoder. It is locally conditioned on mel-spectrograms and globally conditioned on speaker embeddings.

5.4.4 DML loss

In [7], speech waveform samples were quantized and the cross entropy loss was used for modeling categorical distribution, but if we use additional quantization bits (to reduce the quantization noise), the cost of computations may be exponentially increased. Using discretized mixture of logistics (DML) distribution loss [80] could save memory and improve training efficiency because it just needs to predict parameters for each mixture component instead of all bits. For example, modeling 16-bit quantized bits always requires the training of a 65,536-way categorical distribution, while only ten mixtures of logistic distributions are sufficient to model 16-bit audio samples empirically.

DML distribution assumes that each sample point s is composed of a mixture of continuous uni-variate distributions v , and each component v_i obeys logistic distribution, as

$$v = \sum_{i=1}^K \pi_i v_i, \quad \text{where } v_i \sim \text{logistic}(\mu_i, \phi_i) \quad (5.8)$$

where π_i is the mixture weight of component i that satisfies $\sum_{i=1}^K \pi_i = 1$. μ is the mean and ϕ is a scale parameter proportional to the standard deviation. The probability on the observed discretized audio sample s excepting the edge cases (e.g., 0 and 65,535 for 16-bit sampling) would be

$$P(s|\pi, \mu, \phi) = \sum_{i=1}^K \pi_i \left[\sigma\left(\frac{s+1-\mu_i}{\phi_i \zeta}\right) - \sigma\left(\frac{s-1-\mu_i}{\phi_i \zeta}\right) \right] \quad (5.9)$$

$\sigma(\cdot)$ is the logistic sigmoid function. ζ denotes the number of sampling classes and $\zeta = 256$ for 8-bit and 65536 for 16-bit sampling. For the edge case of 0, replace $s-1$ with $-\infty$, and for 255 or 65535, replace $s+1$ with $+\infty$. Finally, the WaveNet model aims at maximizing the average log likelihood of P :

$$L_{DML} = \max_W \mathbb{E}[\log(P(s|\hat{\pi}, \hat{\mu}, \hat{\phi}))] \quad (5.10)$$

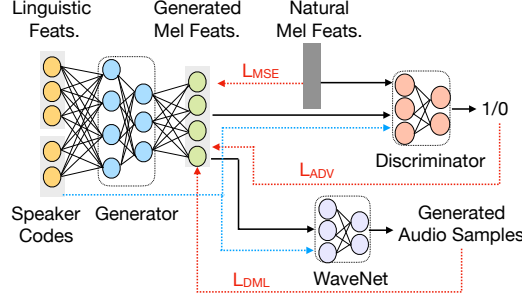


Figure 5.5: Loss functions and gradients for updating acoustic models in the proposed method. Note that neither the model parameters of WaveNet nor the discriminator are updated in this step.

where $\hat{\pi}, \hat{\mu}, \hat{\phi}$ are predicted mixture component parameters.

5.5 Training Algorithm

5.5.1 Training algorithm for the proposed acoustic model

The overall loss function for training the proposed acoustic model that predicts mel-spectrogram can be written as

$$L_G(y, \hat{y}) = L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV}(\hat{y}) + \gamma_W L_{DML}(y, \hat{y}) \quad (5.11)$$

In addition to the general MSE loss L_{MSE} and adversarial loss L_{ADV} , the DML loss L_{DML} generated by a well-trained WaveNet model is utilized for updating the model parameters of the generator. Utilizing the DML loss with the generator would integrate the divergence of synthesized speech samples into the acoustic parametric training process. Therefore, the proposed loss function minimizes not only the parametric error of the mel-spectrogram but also the fidelity disparity between predicted and natural audios. γ_W is a hyper-parameter that denotes the weight of L_{DML} . When $\gamma_W = 0$, the loss function is equivalent to the conventional GAN training. Model parameters of the generator θ_G are updated by using the stochastic gradient calculated from $L_G(y, \hat{y})$. Fig. 5.5 shows the procedure for computing the proposed loss function.

The details of the acoustic model training algorithm are given in Algorithm 1. In the first step, the generator is trained with the MSE criterion for a few epochs. Then, the generator and discriminator are optimized in an iterative way, where one module is being updated while the model parameters of another are fixed. In the final step, the loss of WaveNet $L_{DML}(y, \hat{y})$ is enrolled in the training criterion of the

generator. Before this step, the WaveNet vocoder needs to be trained in advance and the optimum model parameters θ_W should be saved. Note that the DML loss does not join the optimization process of the discriminator, and the parameters of the WaveNet model are always kept fixed. In other words, although θ_D and θ_W are included in calculating $L_G(y, \hat{y})$, θ_D is not updated by the back-propagation of L_G in the final step, and neither is θ_W . The WaveNet model is used as a measurement that reflects the divergence between speech samples. In the WGAN-GP-based case, θ_D is first optimized according to Eq. (5.3) and then θ_G is optimized according to Eq. (5.11).

5.5.2 Time resolution adjustment

During the training of the multi-speaker acoustic model, there are two instances where we need to pay attention to *time resolution* problems. The first is when the mel-spectrograms are input to the WaveNet vocoder. The other is when DML loss is applied for generator optimization.

When acoustic features are transformed into speech samples, conventional parametric vocoders always use interpolation inside frames to recover the audio sampling points. Since different sampling points may share the same acoustic features, in existing studies related to WaveNet, several approaches have been proposed to align the input conditional features with the speech samples.

When acoustic features are transformed into speech samples in the WaveNet vocoder, Oord et al. used a trainable transposed convolutional network to upsample the time resolution of the conditional acoustic features. Deep Voice 2 applied a stack of bidirectional quasi-recurrent neural networks and Tamamori et al. simply duplicated the conditional acoustic feature vector of each frame. In our work, we use trainable transposed convolutional layers to align mel-spectrograms and speech samples for the WaveNet vocoder as in [7].

When the well-trained WaveNet vocoder is used for the proposed generator optimization, it would be time-consuming to calculate the DML loss along all the waveform audio samples within the same frame. As shown in Fig. 5.6, in order to improve computational efficiency, we randomly select a part of the waveform audio points within each frame and back-propagate their averaged DML loss to the generator for acoustic model optimization.

Algorithm 1 Training algorithm for acoustic modeling.

Require:

- 1: $x :=$ linguistic features; $c :=$ speaker code; $y :=$ mel-spectrogram;
- 2: Initial generator parameter θ_G and initial discriminator parameter θ_D ;
- 3: A well-trained WaveNet model W and θ_W is fixed;
- 4: batch size m , learning rate η , the gradient penalty coefficient λ , weight for adversarial loss r_D , weight for DML loss r_W , generator warming up iterations $n1$, basic adversarial training iterations $n2$, number of total iterations $n3$.

Begin step 1: warming up generator

- 1: **for** epoch = $1, \dots, n1$ **do**
- 2: **for** training data in (x, c, y) **do**
- 3: generate \hat{y} from the generator

$$\hat{y} = G(x, c)$$

- 4: update θ_G using MSE criterion:

$$\theta_G \leftarrow \theta_G - \eta_G \nabla_{\theta_G} L_{MSE}(y, \hat{y})$$

- 5: **end for**

- 6: **end for**

End**Begin** step2: adversarial training

- 1: **for** epoch = $n1, \dots, n2$ **do**
- 2: **for** training data in (x, c, y) **do**
- 3: **for** $i = 1, \dots, m$ **do**

$$\hat{y} = G(x, c)$$

$$\tilde{y} = \epsilon y + (1 - \epsilon \hat{y}), \quad \epsilon \in U[0, 1]$$

$$L_D^{(i)} = D(y) - D(\hat{y}) + \lambda (\|\nabla_{\tilde{y}} D(\tilde{y})\|_2 - 1)^2$$

- 4: **end for**

- 5: update θ_D while fixing θ_G :

$$\theta_D \leftarrow \theta_D - \eta_D \nabla_{\theta_D} \frac{1}{m} \sum_{i=1}^m L_D^{(i)}$$

- 6: update θ_G using both MSE and adversarial criterion:

$$L_{ADV} = \frac{1}{m} \sum_{i=1}^m D(G(x, c))$$

$$\theta_G \leftarrow \theta_G - \eta_G \nabla_{\theta_G} (L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV})$$

- 7: **end for**

- 8: **end for**

End**Begin** step 3: fine tuning the generator by utilizing WaveNet loss.

- 1: **for** epoch = $n2, \dots, n3$ **do**
- 2: **for** training data in (x, c, y) **do**
- 3: generate \hat{y} and update θ_D following step 2.
- 4: upsampling \hat{y} .
- 5: generate \hat{s} from the well-trained WaveNet model:

$$\hat{s} = W(\hat{y}, c)$$

- 6: update θ_G with DML loss from WaveNet:

$$\theta_G \leftarrow \theta_G - \eta_G \nabla_{\theta_G} (L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV} + \gamma_W L_{DML}(s, \hat{s}))$$

- 7: **end for**

- 8: **end for**

End

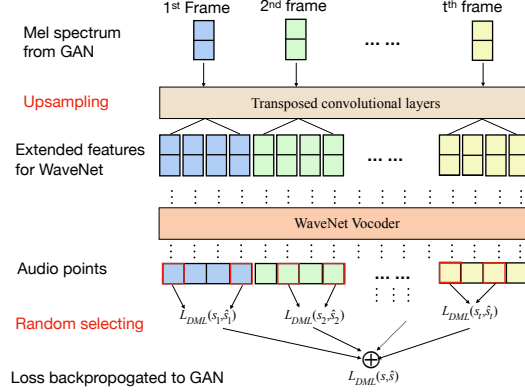


Figure 5.6: Time resolution adjustment of conditional acoustic features. One frame includes four waveform audio points. Transposed convolutional layers are used to upsample the conditional acoustic features. The DML loss was computed using randomly selected waveform audio points within each frame.

5.6 Experimental Setup

We used six speakers (awb, bdl, clb, ksp, rms, and slt) from the CMU-ARCTIC database for multi-speaker training. Two speakers (clb and slt) are female and the others are male. For each speaker, 1000 utterances were used for training. Their speech waveforms have a sampling frequency of 16 kHz and a 16-bit PCM format. The six speakers read out the same set of utterances. Linguistic labels were generated by Festival TTS and consist of 376-dimensional binary vectors and 5-dimensional duration information. The linguistic features are normalized by the min-max rule. Speaker codes consist of seven dimensions, where six dimensions represent speaker identity difference in one-hot format and the other dimension denotes gender. The speaker codes are input to the first layer of both the generator and discriminator as auxiliary features. For the WaveNet vocoder, the speaker codes are first input to a fixed-size embedding layer and then converted to an input format compatible with WaveNet. None of the utterances in the testing set appear in either the training or development sets.

As acoustic features, 80-dimensional static mel-spectrograms are adopted in our experiment. To compute mel-spectrograms, we first perform a short-time Fourier transform (STFT) on audios using a 15-ms frame size, 5-ms frame shift, and a Hann window function. Then we transform the STFT magnitude spectrum to the mel scale using an 80-channel mel-filterbank that ranges from 125 Hz to 7.6 kHz, followed by log dynamic range compression. Prior to the log compression, the filterbank output

magnitudes are clipped to a minimum value of 0.01 in order to limit the dynamic range in the logarithmic domain. The mel-spectrograms are then normalized to have zero-mean unit variance.

We used six bidirectional SRU layers for acoustic modeling and three feed-forward layers for the discriminator. In the generator, each layer has 512 hidden nodes, and in the discriminator, each layer has 128 hidden nodes. The ReLU activation function is utilized in the SRU cell. A stochastic gradient descent (SGD) optimizer was used as the optimizer for both the generator and discriminator. Learning rate was initialized to 0.01 for the generator and 0.001 for discriminator along with exponential decays corresponding to the number of training epochs.

To implement the WaveNet model, we referenced [90] and adopted a modified version of the WaveNet architecture. Instead of predicting discretized buckets with a softmax layer, we followed Tacotron 2 and Parallel WaveNet and used a 10-component mixture of logistic distributions to generate 16-bit samples at 16 kHz. To compute the logistic mixture distribution, the WaveNet stack output was passed through a ReLU activation, followed by a linear projection to predict parameters (mean, log scale, mixture weight) for each mixture component. We adopted 24 dilated convolution layers grouped into four dilation cycles. The dilation rate of the k -th layer was set to $2^{k \pmod 6}$, where $k \in [0, 1, 2 \dots 23]$. Finally, 24 residual blocks were connected. The number of channels of (dilated) causal convolution and 1×1 convolution in the residual block were set to 512. The number of 1×1 convolution channel between skip-connection and output layer was set to 256. We used three transposed convolutional layers for up-sampling. The Adam algorithm [91] was used for the optimization, and its learning rate was initialized to 0.001 and scheduled carefully with a scheme similar to [92]. Other parameters in the Adam optimizer were set as $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1.0e^{-8}$. We also maintained an exponentially weighted moving average of the network parameters over update steps with a decay of 0.9999. A GeForce GTX 1080 was used for training. It took about a week to train a high-quality multi-speaker WaveNet vocoder and eight minutes to synthesize ten seconds of speech. When updating the generator using the DML loss back-propagated from the trained WaveNet Vocoder, we randomly chose half of the sampling points in each frame to efficiently calculate the DML loss. γ_D was set equal to $E(L_{MGE})/E(L_{ADV})$, and $E(\cdot)$ represented expectation value. γ_W was fixed as 0.0001.

5.7 Experimental Evaluation

We compared the performance of the following configurations based on a listening test:

1. Baseline: Acoustic model trained using $L_{MSE}(y, \hat{y})$ as a criterion.
2. GAN: Acoustic model trained using $L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV}(\hat{y})$ as a criterion.
3. GAN^W: Acoustic model trained using $L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV}(\hat{y}) + \gamma_W L_{DML}(y, \hat{y})$ as a criterion.
4. WGAN-GP: Acoustic modeling trained using $L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV}(\hat{y})$ as a criterion. WGAN-GP was also used.
5. WGAN-GP^W: Acoustic model trained using $L_{MSE}(y, \hat{y}) + \gamma_D L_{ADV}(\hat{y}) + \gamma_W L_{DML}(y, \hat{y})$ as a criterion. WGAN-GP was also used.
6. Analysis by synthesis (AbS): Synthetic speech generated by a WaveNet vocoder using ground-truth mel-spectrograms.
7. Natural: Natural speech.

Note that systems 1 to 5 are TTS systems and use SRU as basic architectures for acoustic models, as described earlier. Also note that all the above TTS systems and analysis by synthesis use the same WaveNet vocoder. The differences are how the local condition parameters of the WaveNet vocoder, that is, mel-spectrogram, are predicted.

5.7.1 Evaluation methodology

For the listening test, we selected 20 utterances from the testing set of each speaker and generated sets of synthetic speech corresponding to the above experimental systems. Each experimental system had 20 utterances, so $20 \text{ utterances} \times 6 \text{ speakers} \times 7 = 840$ samples that needed to be evaluated in total. Crowdsourced perceptual evaluation was carried out to evaluate naturalness as well as speaker similarity of generated speech. In the crowdsourcing test, we evaluated each sample ten times to alleviate personal bias. The testing samples were divided into different evaluation sets. Each set consisted of three utterances generated by seven different systems. Therefore, there were 42 utterances to be evaluated in each set: 21 for naturalness and 21 for similarity. We then collected 400 sets to cover all 840 samples ($400 =$

840 \times 10/21). This guarantees at least 40 unique listeners, since we limited the maximum number of sets per crowdsourced participant to ten. The actual number of listeners who participated in our test was 42.

To evaluate naturalness, listeners were asked to ignore the meaning of the sentence and concentrate only on rating how natural the speech sounded on a five-point scale:

1. completely unnatural
2. mostly unnatural
3. equally natural and unnatural
4. mostly natural
5. completely natural

For speaker similarity, listeners were asked to ignore the meaning of the sentence and concentrate only on rating the speaker identity. Synthetic speech samples and the corresponding natural sound were presented in pairs at every turn and listeners were asked to judge whether the two samples were from the same or different speaker(s). The scale for speaker similarity was judged on a four-point scale:

1. same speaker, absolutely sure
2. same speaker, not sure
3. different speaker, not sure
4. different speaker, absolutely sure

5.7.2 Evaluation results and analysis

Fig. 5.7 shows the box plots for the naturalness evaluation results averaged across all speakers. Table 5.1 shows statistical significance. From these, we can see that four GAN-based experimental groups (GAN, GAN^W, WGAN-GP, WGAN-GP^W) outperform the baseline significantly. Upper quartiles and mean opinion scores of the four GAN-based groups are much higher than those of the baseline, although their lower quartiles are quite similar to the baseline. Note that all the systems (apart from natural speech) use the same WaveNet vocoder. Hence, this also indicates that the quality of WaveNet synthetic speech is affected by the local condition parameters and that the ones predicted by the GAN-based acoustic models sound more natural than

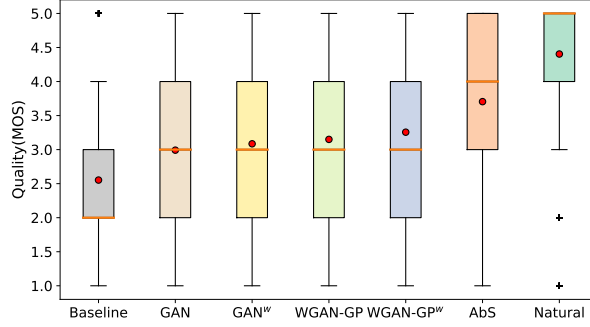


Figure 5.7: Box plots on naturalness evaluation results. Red dots represent the mean of each group averaged across all speakers.

Table 5.1: Statistical significance analysis using t -tests with Holm-Bonferroni correction in terms of quality judgment.

System	Baseline	GAN	GAN ^W	WGAN-GP	WGAN-GP ^W	AbS
GAN	$\leq 2e-16$	-	-	-	-	-
GAN ^W	$\leq 2e-16$	0.05206	-	-	-	-
WGAN-GP	$\leq 2e-16$	0.00028	0.19850	-	-	-
WGAN-GP ^W	$\leq 2e-16$	1.2e-06	0.01916	0.24092	-	-
AbS	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$	-
Natural	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$	$\leq 2e-16$

those by the baseline. We also see that WGAN-GP systems (WGAN-GP, WGAN-GP^W) are better than the original GAN system. The use of DML loss alone did not bring statistically significant improvements, but it obviously reduced p -values (see Table 5.1), and hence a combination of WGAN-GP and the DML loss resulted in the highest scores among the TTS systems and was significantly better than GAN and GAN^W ($p < 0.05$).

Compared with the natural speech and AbS, all TTS methods have obvious gaps. There is also a gap between the AbS samples and natural speech. This indicates that our multi-speaker TTS systems do not sound as good as natural speech yet, and the multi-speaker WaveNet vocoder itself does not sound as good as natural speech either, even if it uses the ground-truth mel-spectrogram. In other words, both the neural vocoder and the acoustic model have room for further improvement.

Through our experiments, we found that the quality of our synthetic speech varied speaker by speaker. Fig.5.8 shows box plots of the MOS scores of the best WGAN-GP^W system and the AbS system of the six speakers. The left box plot shows the

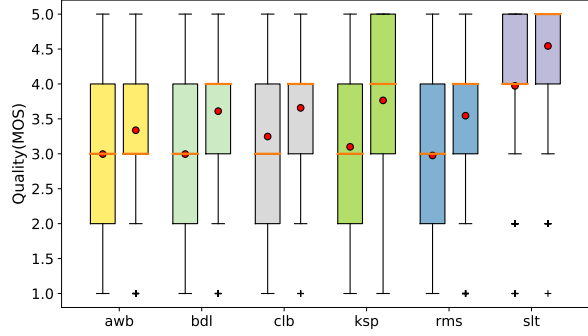


Figure 5.8: Box plots of the MOS scores of six speakers. Left: WGAN-GP^W system. Right: AbS system.

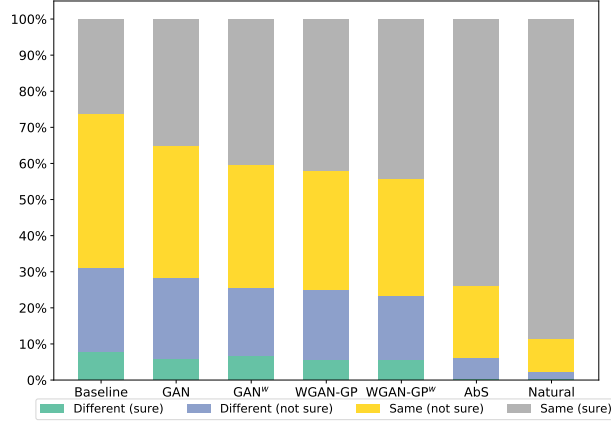


Figure 5.9: Similarity results averaged across all speakers.

results of the WGAN-GP^W system and the right box plot shows those of the AbS system for each speaker. Interestingly, the quality of synthetic speech varied speaker by speaker, and there is a very large gap between speaker SLT and the other speakers. This implies that we need a more generalized model that can handle multiple speakers better and can reproduce the differences between speakers more precisely.

The similarity evaluation results are shown in Fig.5.9 and the t-test results for similarity are shown in Table 5.2. The trend of the similarity tests is very similar to that of the naturalness. The WGAN-based systems outperform the baseline, and we can clearly see that the portions of "Same" (yellow and gray) have been increased. The proposed systems using a combination of WGAN-GP and DML loss achieved more apparent preference in terms of "Same, absolutely sure". Likewise in the quality evaluation, we can see a gap between TTS systems and WaveNet analysis-by-synthesis systems as well as between WaveNet analysis-by-synthesis systems and natural speech.

Table 5.2: Statistical significance analysis using t -tests with Holm-Bonferroni correction in terms of speaker similarity judgment.

Systems	Baseline	GAN	GAN ^W	WGAN-GP	WGAN-GP ^W	AbS
GAN	0.43810	-	-	-	-	-
GAN ^W	0.28401	1.00000	-	-	-	-
WGAN-GP	0.00426	0.31593	0.47565	-	-	-
WGAN-GP ^W	0.00044	0.11438	0.28401	1.00000	-	-
AbS	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 2e-16$	-
Natural	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 2e-16$	$\downarrow 4.2e-06$

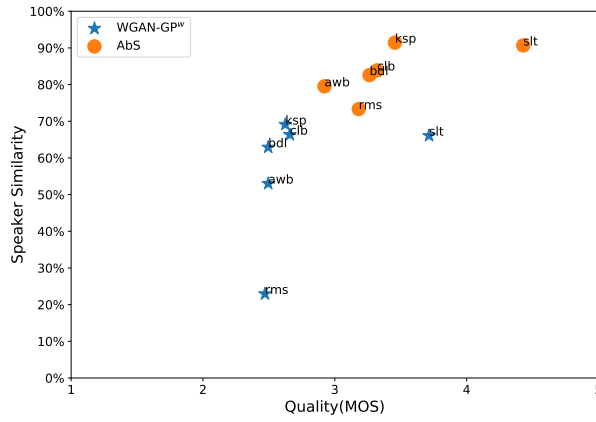


Figure 5.10: Scatter plot matching naturalness and similarity scores for each speaker in system WGAN-GP^W and AbS. The similarity score is defined as the added percentage of ‘same (not sure)’ and ‘same (sure)’ scores.

Fig. 5.10 shows a scatter plot matching naturalness and similarity scores of the best WGAN-GP^W system and AbS system of six speakers. Interestingly, the speaker similarity scores also significantly varied speaker by speaker, and speaker RMS had a very low speaker similarity score. Our next step is to investigate why a few speakers had lower speaker similarity.

5.8 Conclusion

This paper investigated how we should train the acoustic model that predicts the local condition parameters to be used by neural vocoders. Specifically, we looked into conditional GANs or WGAN-GP to reduce the mismatched characteristics between natural and generated acoustic features. We also extended the GAN frameworks and used the discretized mixture logistic loss of a well-trained WaveNet along with

mean squared error and adversarial losses as parts of the objective functions. These new objective functions were evaluated in multi-speaker speech synthesis that uses the WaveNet vocoder. Experimental results show that acoustic models trained with the WGAN-GP framework using back-propagated DML loss achieved the highest subjective evaluation scores in terms of both quality and speaker similarity.

Our future work will investigate why some speakers have lower quality of synthetic speech or lower similarity. We will also perform larger scale experiments using more speakers.

Chapter 6

Prosody Prediction in Text to Speech Synthesis

In Text-to-Speech system, prosodic attributes have to be predicted only from input text. The accuracy of prosody prediction has a significant effect on the naturalness of synthesized speech of Chinese. In this paper, we explore using neural networks to predict prosodic boundaries from Chinese text without task specific knowledge or sophisticated feature engineering. We examine sequence character-level features and word-level features, and compare their performance with one-hot and embedding representations. Instead of traditional cascaded prediction, we propose a unified framework which can be considered to be a multi-task learning process. Experimental results show that character-level features can obtain approximate F-scores compared to those with word-level features, and embedding features learned from large unlabeled texts can help to enhance the performance. The unified framework can achieve similar performance to cascaded framework, while using less training time and without the necessary of preparing task-specific features.

6.1 Introduction

In linguistics, prosody is concerned with properties of syllables and larger units of speech, which contribute to linguistic functions such as phrase-based chunking by intonation, rhythmical organization of an utterance using lexical stress, and so on. Prosody can also reflect various extra- or para-linguistic aspects of various characteristics of speaker or utterance: the emotional state of speaker, the form of utterance, the presence of irony or sarcasm, emphasis, contrast, and focus [93]. These clearly indicate that the accuracy of prosody prediction has a significant effect on the naturalness of synthesized speech.

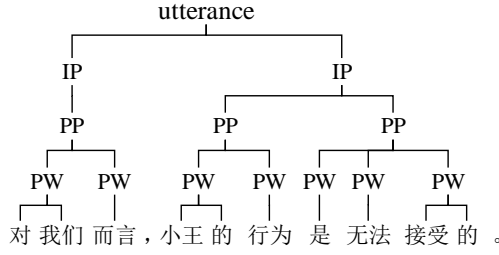


Figure 6.1: The hierarchical prosodic structure in Chinese.

In Chinese TTS systems, to specify the prosodic structure of a given text, the following hierarchical features have to be predicted automatically [94, 95]: 1) prosodic word boundaries (PW), prosodic phrase boundaries (PP), and intonational phrase boundaries (IP), as is shown in Fig 6.1. The leaf nodes of the tree structure are lexical words which are deduced from a word segmentation module. A great number of linguistic features and various prosody modeling methods have been investigated in previous research. Some syntactic cues like part-of-speech (POS), syllable identity, syllable stress and their contextual counterparts are commonly used for prosodic structure prediction [96–98]. Many statistical methods have been investigated to predict prosodic structure, including classification and regression tree (CART) [99], hidden Markov model (HMM) [100], maximum entropy model (MEM) [101] and conditional random fields (CRF) [102]. Due to its ability of relaxing assumption of strong model independence and solving label bias problem, CRF has achieved superior performance in prosodic structure prediction [103–105]. However, CRF still suffers two major drawbacks. First, its performance can be easily affected by Chinese Word Segmentation (CWS) and Part-of-Speech (POS) tagging. Second, it heavily relies on manually feature engineering [106].

To overcome the disadvantages of traditional prediction based on CRF, a new architecture based on deep neural networks (DNN) and embedding features has been proposed in previous work [106]. It has proved that stacking feed-forward and bidirectional long short-term memory (BLSTM) recurrent network layers achieves superior performance over the CRF-based method. The embedding features learned from raw text further enhance the performance. Similar conclusion can be drawn from other studies [107]. However, in [106], although character-level features have been investigated while word-level features are always considered indispensable in prosodic structure prediction. Moreover, three separate neural networks were trained inde-

pendently for PW, PP and IP in a cascaded framework, which is blamed for error accumulation.

Word level-features are usually considered helpful for prosodic structure prediction because word boundaries are always prosodic boundaries. However, manual word segmentation is quite laborious and automatic word segmentation will inevitably cause some errors. The particle size of CWS is difficult to choose. Further, in Chinese dialect synthesis, high-accuracy word segmentation is difficult to realize because it is difficult to prepare a large enough corpus. Even in that case, character-level features can be always correctly extracted. This is because character-level features do not contain boundary information, but it can avoid the negative effect of particle size and inevitable segmentation errors in CWS.

In this paper, we continue one of the authors' research on prosodic structure prediction with the neural network architecture. This work has two main contributions: (1) N-gram sequence character-level and word-level features are investigated in both one-hot and embedding form to seek the most suitable features for prosodic structure prediction. (2) Instead of traditional cascaded prediction, a unified framework which can be considered as a multi-task learning process is proposed, with expect to eliminate the propagation errors and benefit each target from others' information.

6.2 Neural network based prediction

To solve the problems in traditional prediction based on CRF, a variant of the neural network architecture [108] for probabilistic language model is proposed in [106]. As shown in Fig. 6.2, the architecture takes raw text as input and maps each Chinese character into a basic feature vector. The following layers are two types of neural networks, feed-forward neural network (FFNN) and BLSTM recurrent neural network, used to discover multiple levels of feature representations from the basic feature vectors. After network prediction, tag inference is performed to find an optimum tag transformation path globally.

6.2.1 Network Structures and Training

A hybrid network structure which includes both feed forward and BLSTM-RNN layers is investigated in work [106]. FFNN, trained with a back-propagation learning algorithm [58], is widely used in many practical applications. In a typical FFNN, every unit in a layer, which is connected to all the units in the previous layer, takes in the output of the previous layer and computes a new set of non-linear activations

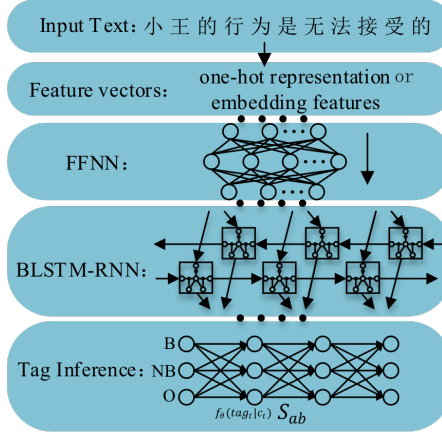


Figure 6.2: The neural network architecture for prosodic boundary prediction. In tag inference, B, NB and O denote boundary, non-boundary and others (e.g., punctuation), respectively.

for next layer. However, the assumption of sample independence results in limited ability of modeling context.

Researchers have proposed RNN to solve the limitation of FFNN. Conventional RNN is only able to make use of previous context information. This is not accurate in modeling prosody that is highly related with both past and future contexts. Instead, bidirectional RNN can access both the preceding and succeeding input contexts with two separate hidden layers, which are then fed to the same output layer. The activation function \mathcal{H} of RNN is usually a sigmoid or hyperbolic tangent function, which often causes the gradient vanishing problem that prevents RNN from modeling the long-span relations in sequence features. An LSTM architecture, which uses purpose-built memory cells to store information, can overcome this problem and model longer contexts. BLSTM-RNN is a combination of LSTM and BRNN.

Deep bidirectional LSTM-RNN can be established by stacking multiple BLSTM-RNN hidden layers on top of each other. The output sequence of one layer is used as input sequence to the next layer. The neural networks can be trained effectively in a layer-wised training manner, which makes it convenient to stack different types of neural network layers on top of each other to form a deep architecture. The deep architecture is able to build up progressively higher level representations of the input data, which is a crucial factor of the recent success of hybrid systems [59].

6.2.2 Tag inference

To find an optimum tag transformation path globally, tag inference is used to model tag dependency. For input character sequence of a sentence $x_{[1:T]}$ with a tag sequence

$tag_{[1:T]}$, a sentence-level score is given by the sum of transition and network scores [109, 110]:

$$l(x_{[1:T]}, tag_{[1:T]}, \theta) = \sum_{t=1}^T (S(tag_{t-1}, tag_t) + f_{\theta}(tag_t|x_t)) \quad (6.1)$$

where $S(a, b)$ is the transition score from tag a to tag b . a and b belong to a set of tags $G = \{B, NB, O\}$. $f_{\theta}(tag_t|x_t)$ indicates the score output for tag_t at the t -th character by the network θ . The best tag path $tag_{[1:T]}^*$ can be found by maximizing the sentence score:

$$tag_{[1:T]}^* = \arg \max_{\forall l_{[1:T]}} l(x_{[1:T]}, tag_{[1:T]}, \theta). \quad (6.2)$$

6.3 Proposed Approach

6.3.1 Word-level features vs. character-level features

Word level-features are usually considered helpful for prosodic structure prediction because word boundaries are always prosodic boundaries. However, manual word segmentation is quite laborious and automatic word segmentation will inevitably cause some errors. The particle size of CWS is difficult to choose. In Chinese dialect synthesis, high-accuracy word segmentation is difficult to realize because it is difficult to prepare a large enough corpus. In that case, character-level features can perform better than word-level features. Character-level features do not contain obvious boundary information, but it can avoid the negative effect of particle size and inevitable segmentation errors in CWS. In our work, we investigate both word-level and character-level features. In addition to single Chinese character, a sequence with N characters is also examined, and we call it N -gram sequence. For current character x_t , its N -gram sequence is defined as follows:

$$X = \{x_{t-\frac{N-1}{2}}, \dots, x_{t-1}, x_t, x_{t+1}, \dots, x_{t+\frac{N-1}{2}}\} \quad (6.3)$$

6.3.2 One-hot features vs. embedding features

Before being fed into network, word-level features or character-level features are transformed into vectors by mapping operation. One-hot and embedding representations are two possible methods. In this work, we would like to compare the results under both kinds of representation. Typically, a character dictionary D of size $|M|$ is extracted from the training set and unknown characters are mapped to a special

symbol that is not used elsewhere. The character set in D are represented as $D = (d_1, d_2, \dots, d_M)$. For n -gram sequence X , we can simply use $H(X) = (h_1, h_2, \dots, h_M)$ as its one-hot representation. h_i is defined as follows:

$$h_i = \begin{cases} 1 & (d_i \in X), \\ 0 & (d_i \notin X). \end{cases} \quad (6.4)$$

For a word $W = (c_1, c_2, \dots, c_N)$ which include N characters, its one-hot form representation $H(W) = (h_1, h_2, \dots, h_M)$ is defined as equation 6.4.

However, one-hot representation is blamed for high dimensions, and it fails to model the semantic similarity between the ideographic characters. In contrast, the distributed representation or embedding feature, the form of a low dimensional continuous-valued vector learned from raw text in a fully unsupervised manner using neural networks, has been experimentally proved to carry important syntactic and semantic information [111]. In [106], in order to prepare embedding features, a skip-gram model *word2vec* which is proposed by Mikolov et al. [112] is chosen. As preliminary experiments did not show much difference of performance among various embedding features, we use *word2vec* in this study.

6.3.3 Unified framework vs. Cascaded framework

In traditional prosody prediction task, prosodic structure is predicted in a cascaded form. PW boundary is firstly predicted according to automatically detected word boundaries. Then, the predicted boundary of PW is used as given information for PP boundary prediction. The result of PP boundary prediction is used for IP boundary prediction. As is shown in Fig. 6.3(a), such kind of cascading structure can ensure prosodic boundaries consistence, where prosodic hierarchy is guaranteed. That is to say, the predicted boundaries of IP would always be the predicted boundaries of PP, and the predicted boundaries of PP would always be the predicted boundaries of PW. However, the errors that occurred in PW prediction could affect the accuracy of PP prediction, and even IP prediction. Moreover, different input and output vectors should be prepared separately, training three deep network is time-consuming.

In order to avoid or propagation error, we test a unified prediction framework, which is showed in Fig. 6.3. The network used for unified prediction shares the same input vectors but generate different output forms compared with cascaded structure. In unified framework, PW, PP and IP are transformed into a one vector and predicted at the same time by only one network, which can avoid exploiting specific

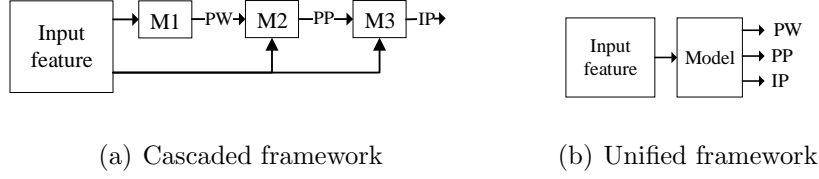


Figure 6.3: Framework of prosodic structure prediction. M represents for neural network model

features carefully optimized for each task and eliminate error propagation. This unified training process can also be considered as multi-task learning. Learning multiple related tasks simultaneously has been empirically as well as theoretically shown to often improve performance significantly compared with learning each task independently [113]. We hope the knowledge of different tasks can benefit each other.

6.4 Experimental Setup

Our database includes 48,210 Chinese sentences. 90% utterances are used for training, 5% are used for validation, the others are used for testing. Prosodic boundaries of all the sentences are manually labelled by experts. Word segmentation and POS tagging is performed by a software provided by Language Technology Platform (LTP) [114]. The accuracy of word segmentation is 97% and the accuracy of POS tagging is 96%. A dictionary with 4,030 characters is extracted from the training dataset. To prepare embedding features, embedding models are trained with a large set of raw texts which are collected from People’s Daily. All texts are preprocessed with text normalization. To perform tag inference, statistical jumping scores between three different boundary tags are estimated from training data.

We have performed prosodic prediction in a cascaded framework and in our unified framework respectively. For each framework, we prepared four groups of features to test. The four groups of features are listed as follows:

- (1) n-gram character-level one-hot features
- (2) word-level one-hot features
- (3) n-gram character-level embedding features
- (4) word-level embedding features

For cascaded framework, three separate neural network models were trained independently for PW, PP and IP. For one-hot features, the network inputs for PW prediction are 4300-dimensional feature vectors, and the inputs for PP or IP prediction have 4031 dimensions. For embedding features, different sizes (256M, 512M, 1024M and 2048M text) of data for unsupervised training and different dimensions (128, 256, 512, 1024) of features are tested. The network outputs have three dimensions and each corresponds to one of three boundary tags: B, BN and O. B for a boundary, NB for non-boundary, and O for other symbols such as punctuation.

For unified prediction, PW, PPH and IPH are predicted by the same and integrated network. Input vectors are the same with the network used for PW prediction in the cascaded framework. Output vectors have nine dimensions which include boundary tags for PW, PP and IP.

Implementation of the network training is done with the help of a machine learning library “CURRENNT” [47]. To find out the best network structure for prediction, we have tested several different distributions of nodes’ size and layer components. The momentum of training is set to 0.9. Learning rate for PW prediction is set to 1e-3, for PP and IP prediction it is set to 1e-4 in the cascaded framework. And the learning rate used in unified framework is set as 1e-5. The maximum number of iteration is set to 300 and training process is stopped if no improvement observed within the latest 20 iterations. A softmax output layer is used in the network and statistical scores of boundary tags can be predicted by the trained network. Then tags inference is performed and an optimum transformation path of all tags in one sequence is searched globally by using Viterbi algorithm.

CRF approach is also used for comparison. It is performed with the help of CRF++ toolkit [115].

6.5 Evaluations and Analysis

6.5.1 Objective evaluation

F-score is used as evaluation criteria. In this paper, we only reports the evaluation results with the optimum network structure that was experimentally determined. For embedding features, only results with optimum trained embedding models and feature dimensions are listed.

Table 6.1 shows the evaluation results of CRF prediction. Table 6.3 shows the evaluation results of cascaded prediction with four different groups of features. From Table 6.1 and Table 6.3 we can see that for all methods, PW owns rather higher

Table 6.1: F-score (%) of CRF-based prosody prediction

Boundary	PW	PP	IP
F-score	95.72	80.60	76.15

Table 6.2: F-score (%) of N-gram character-level one-hot features

N-gram	N=1	N=3	N=5	N=7	N=9	N=11
PW	95.40	94.90	94.65	94.32	94.15	94.11
PP	83.71	83.71	82.79	81.93	81.34	81.21
IP	81.21	80.90	80.37	79.95	79.66	79.22

F-score than PP and IP. But neural network based approaches can predict prosodic boundaries more accurately than CRF, especially in terms of PP and IP. For n-gram character-level one-hot features, prediction accuracy decreases while the length of sequence increases, and 1-gram achieves highest evaluations. Due to this reason, in other experiments, we only use 1-gram character-level features. Compared with 1-gram character-level one-hot features, word-level features with one-hot representation has higher F-score in terms of PW prediction, but a little worse in terms of PP and IP prediction. According to Table 6.3, we can find similar phenomenon. Word-level embedding features has higher F-score than character-level embedding features for PW prediction, but lower F-score for PP and IP prediction. Embedding representation outperform one-hot representation in terms of both character-level and word-level features. Among all these experiment mentioned above, word-level embedding features show best performance for PW prediction, and character-level embedding features obtain highest F-score in terms of PP and IP prediction.

We choose character-level and word-level embedding features, which have shown best performance in cascaded experiments, to perform prosodic structure prediction in our unified framework. The results is showed in Table 6.4. For both character-level and word-level embedding features, unified framework perform better in PW prediction, and F-score of PP and IP prediction is a little lower but quite approximate to cascaded framework.

Table 6.3: F-score (%) of cascaded prediction. C represents for character-level features, W represents for word-level features

	PW	PP	IP
C one-hot	95.40	83.71	81.21
W one-hot	95.73	83.28	80.99
C embedding	96.04	84.60	81.78
W embedding	96.34	84.31	81.32

Table 6.4: F-score (%) of unified prediction. C represents for character-level features, W represents for word-level features

	PW	PP	IP
C embedding	96.20	84.38	81.43
W embedding	96.54	84.18	81.04

6.5.2 Subjective evaluation

We further conducted an A/B preference test on the naturalness of the synthesized speech. A set of 50 sentences were randomly selected from the test set and the prosodic boundary labels were achieved by:

1. Character-level one-hot in Table 6.3;
2. Word-level one-hot in Table 6.3;
3. Character-level embedding in Table 6.3;
4. Word-level embedding in Table 6.3;
5. Character-level embedding in Table 6.4;
6. Word-level embedding in Table 6.4;

We carried out two sessions of comparative evaluations: (3) vs (1), (4) vs (3), (5) vs (3), and (5) vs (6). A set of 10 sentence pairs of each session was randomly selected from the 50 pairs with different prosody prediction results and speech was generated through a typical DNN-based TTS system. A group of 10 subjects were asked to choose which one was better in terms of the naturalness of synthesis speech. The percentage preference is shown in Fig 6.4. We can see that unified framework can achieve higher preference than cascaded ones. And character level embedding features is slightly preferred than word-level features in unified framework, which is opposite to the cascaded framework.

C embedding (cascaded) 40.9%	Neural 19.9%	C one-hot (cascaded) 39.2%
W embedding (cascaded) 45.6%	Neural 22.9%	C embedding (cascaded) 31.5%
C embedding (unified) 37.7%	Neural 27.6%	C embedding (cascaded) 34.7%
C embedding (unified) 43.5%	Neural 25.2%	W embedding (unified) 31.3%

Figure 6.4: Subjective comparison for synthesized speech naturalness using different features. The p -value of the four pairs are 0.31, 1.0 , 0.97, and 0.42 respectively.

6.6 Conclusions

In this paper, we have investigated BLSTM-RNN-based prosodic prediction with different groups of features in both cascaded and unified framework. Experimental results show that embedding features outperform one-hot features in all cases. Word-level features achieve similar evaluations to character-level features. Although unified prediction has achieved similar evaluation to cascaded framework, its performance is not as excellent as we expected. However, unified framework requires less computational work and without the necessity of preparing task specific features. And multi-task learning has already been both experimentally and theoretically proved to be capable of outperforming separate task learning. We will go on our study and try to improve performance.

Chapter 7

Conclusions

In this paper, we mainly focus on the deep neural network based flexible and reliable text to speech synthesis. We first proposed a multi-speaker speech synthesis framework based on BLSTM-RNN, and proved its efficiency in improving speech quality as well as speaker similarity. We also successfully applied this framework on new speaker’s adaptation. To control speaker identity in the proposed framework, we examined various kinds of speaker representations. Finally, we propose to incorporate generative adversarial network as well as WaveNet vocoder into multi-speaker speech synthesis. Further, we propose a unified framework to improve the accuracy for prosody prediction. In future, it is necessary for us to focus on real-time and end-to-end text-to-speech research.

Appendix A

Speaker code based stimulated training for network visualization

Controlling the characteristics and generating arbitrary speaker’s voice are important topics in DNN-based speech synthesis. However, the parameters of the network are hard to analyze, making speaker identity control challenging. Stimulated training has been recently proposed for network interpretation as well as generalization. In order to explore the impact of speaker identity information during the multi-speaker acoustic model training process, we stimulated the speech synthesis network with speaker identity vector to visualize the network performance. It is also expected that the stimulated process can help to improve the synthesized speech quality and speaker similarity.

A.1 Introduction

We have investigated speaker representations, such as i-vector and speaker code, to control speaker identity during speaker adaptive training. In some cases, they are used as augmented features to the input layer [40, 50, 84]. In other cases, they are input to hidden layers with connection weights [82, 116]. These speaker representations can be either independently estimated to the acoustic model [84], or jointly estimated by back propagation [116].

However, due to the complexity of neural networks, the parameters of acoustic model are usually hard to analyze. The role of these speaker representations during speaker adaptive training is still in a blackbox, which makes it a challenge to effectively understand and control speaker identity.

Simulated training was proposed and it has provided an effective approach for network visualization [117]. By introducing a stimuli of phonemic priori factor during

training, the nodes in different regions are regularized to their corresponding but different phonemes. In this way, DNN activations can then be interpreted and visualized directly according to the underlying stimulated priori factor. Compared with other visualization approaches such as examining the weights or output activations in each layer [118–120], stimulated training makes it possible to modify trained hidden layers in order to yield different behavior. In addition to network visualization, stimulated training has also been studied in speaker adaptation and generalization for speech recognition tasks [116, 121].

In the following contents, we mainly investigate stimulated training for speaker identity interpretation in DNN-based speech synthesis. To interpret the network activation based on speaker identity during speaker adaptive training, we mapped utterance-level i-vectors of training speakers into different regions of neurons and performed stimulated training with specially designed regularization functions.

A.2 Stimulated Deep Learning

In stimulated training, nodes with similar activation functions are grouped together in a spatial order, instead of forming an arbitrarily ordered set of activations [117]. For stimulated training in ASR, the nodes in the same layer are organized into a two dimensional grid, to which individual phonemes are mapped into according to their acoustic features. The normalized distances between different nodes are used as prior distribution of activation values in a layer. This factor encourages those activation functions in the same region to have the same normalized output.

The hidden layer activation of feedforward DNNs is defined as:

$$\mathbf{h}^l = \sigma(\mathbf{W}^l \mathbf{h}^{l-1} + \mathbf{b}^l), \quad 1 \leq l \leq L \quad (\text{A.1})$$

where $\sigma(\cdot)$ represents the sigmoid function; L is the total number of layers in neural network; \mathbf{W} and \mathbf{b} are the transformation parameters that belong to the l -th hidden layer.

The standard training scheme of DNN tries to minimize criterion $\mathcal{L}(\theta)$ over a training set. Where $\mathcal{L}(\cdot)$ is the loss function and $\theta = \{\mathbf{W}^1, \mathbf{b}^1, \dots, \mathbf{W}^L, \mathbf{b}^L\}$ are network parameters.

To implement stimulation, a regularization term $\mathcal{R}_{st}(\theta)$ is added to the training criterion:

$$\mathcal{F}(\theta) = \mathcal{L}(\theta) + \eta_{st} \mathcal{R}_{st}(\theta). \quad (\text{A.2})$$

$\mathcal{F}(\theta)$ is a new training criterion. $\mathcal{R}_{st}(\theta)$ is the stimulated priori factor and η_{st} determines its contribution. $\mathcal{R}_{st}(\theta)$ is defined based on the KL-divergence between the prior distribution over the current distribution $g(\mathbf{s}_i, \hat{\mathbf{s}}_{pt})$ and the normalized activation \bar{h}_i^l :

$$\mathcal{R}_{st}(\theta) = \frac{1}{T} \sum_t \sum_l \sum_i g(\mathbf{s}_i, \hat{\mathbf{s}}_{pt}) \log \left(\frac{g(\mathbf{s}_i, \hat{\mathbf{s}}_{pt})}{\bar{h}_i^l} \right). \quad (\text{A.3})$$

The phone-specific activation distribution priori factor $g(\mathbf{s}_i, \hat{\mathbf{s}}_{pt})$ is defined as the normalized distance between a node and the current active-phone position:

$$g(\mathbf{s}_i, \hat{\mathbf{s}}_{pt}) = \frac{\exp \left(-\frac{1}{2\sigma_{st}^2} \|\mathbf{s}_i - \hat{\mathbf{s}}_{pt}\|_2^2 \right)}{\sum_j \exp \left(-\frac{1}{2\sigma_{st}^2} \|\mathbf{s}_j - \hat{\mathbf{s}}_{pt}\|_2^2 \right)}, \quad (\text{A.4})$$

where \mathbf{s}_i is the position of the i -th node in the network-grid space; $\hat{\mathbf{s}}_{pt}$ is the position in the network-grid space that is mapped by the correct phoneme at time t ; σ_{st} controls the sharpness of the prior surface.

\bar{h}_i^l is the normalized activation output for the i -th node on the l -th layer:

$$\bar{h}_i^l = \frac{h_i^l \beta_i^l}{\sum_j h_j^l \beta_j^l}, \quad (\text{A.5})$$

$$\beta_i^l = \sqrt{\sum_k w_{ik}^{(l+1)^2}}, \quad (\text{A.6})$$

where β_i^l is used to reflect the impact which the activation function has on the following layer $l + 1$.

The difference between the activation output of normal and stimulated DNNs can be illustrated in Fig A.1. In this figure, the output activation of DNN have been arranged in a 2-D grid plane. The bright region corresponds to high activation and scatters all over the plane as one would expect from a distributed representation. However, this may cause problem for regularization and adaptation as it is generally hard to relate one weight to another in the same layer [121]. For stimulated DNN, the network activation which is showed in the grids appears to be smooth on each layer. The nearby nodes in spatial order are likely to perform in a similar way. Through enhancing the activation corresponding to the phonemes at the top and weakening at the bottom, such an approach not only improved interpretability of the network but also encouraged better discrimination [116].

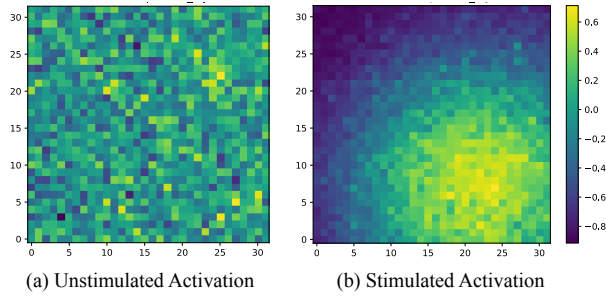


Figure A.1: Activation of normal DNN vs. stimulated DNN

A.3 Network Interpretation in Terms of Speaker Identity

The stimulated training strategy in ASR is performed at a phoneme level. By enhancing the activation of a specific phoneme in its vicinity, network can be interpreted in terms of a variety of phonemes [117] and these phonemes can be better discriminated [121]. In speaker adaptive training of speech synthesis, voice characteristics related to speaker identity tend to be exploited especially at top layer while the phoneme-level information are more inclined to be transformed by lower layers [39]. To better understand and control speaker identity during speaker adaptive training, a stimulated training strategy in terms of speaker identity is introduced. This strategy also builds a relationship between neurons in the same layer, and this kind of relationship are decided by speaker identity.

As for speaker identity, i-vector is one of the most effective representations. The cosine distance between two different i-vectors is often used to represent the difference of two speakers. A speaker’s i-vector is the normalized mean value of one’s utterance-level i-vectors. The dimension of i-vector depends on the size of the total variety matrix and varies from tens to hundreds.

Since arbitrarily ordered DNN nodes cannot provide any insights when they are visualized directly [117], it is necessary to arrange the nodes into specific positions, and i-vector can be mapped to such positions. For example, 1024 nodes in one layer can be arranged in a 32*32 grids. The new location of the i -th node of the network in these grids will be (x, y) , where $x = i/32$ and $y = (i\%32)$. I-vectors can also be mapped in such a grid plane by using special dimension reduction algorithm. For example, t-SNE [122] or LargeVis [123], which maps high dimensional points to a rather lower dimension while keeping the spatial distance. Fig. A.2 shows the application of t-SNE algorithm, where the utterance-level i-vectors of seven speakers are mapped in a 32 * 32 grids.

The reason we chose utterance-level i-vectors is straightforward. The mapped positions of utterance-level i-vectors for each speaker can be modeled with a single Gaussian, and a brief definition of speaker-specific activation distribution priori factor is shown as follows:

$$g_t(s_i^l, \hat{s}_c) = \frac{1}{\sqrt{2 \left(\frac{\sigma_{\hat{s}_c}}{\gamma_{\hat{s}_c}}\right)^2 \pi}} \exp \left(-\frac{1}{2 \left(\frac{\sigma_{\hat{s}_c}}{\gamma_{\hat{s}_c}}\right)^2} \|s_i^l - \mu_{\hat{s}_c}^l\|^2 \right), \quad (\text{A.7})$$

where l is the index of layer, s_i is the position of the i -th node in the network grid space. c represents for the c -th speaker. \hat{s}_c is a collection of the positions which are mapped by the utterance-level i-vectors of the c -th speaker. $\mu_{\hat{s}_c}$ is the mean position of utterance-level i-vectors and $\sigma_{\hat{s}_c}$ is the variance. $\gamma_{\hat{s}_c}$ is a variable controlling the sharpness of the prior surface. $g_t(\cdot)$ is a time-varying function because the speaker identity might vary frame by frame.

In the acoustic modeling task of speech synthesis, tanh activation function has been proved to be more efficient than sigmoid function since the targets of neural network, acoustic features, can be either positive or negative. In order to estimate KL-divergence between the speaker-specific priori factor $g(\cdot)$ and activation, a linear transformation is performed on the activation output $h_i^l(\cdot)$:

$$\hat{h}_i^l = \frac{h_i^l + 1}{2}. \quad (\text{A.8})$$

And the corresponding normalized network activation output becomes:

$$\bar{h}_i^l = \frac{\hat{h}_i^l \beta_i^l}{\sum_j \hat{h}_j^l \beta_j^l}. \quad (\text{A.9})$$

In traditional DNN-based speech synthesis, the loss function is:

$$\mathcal{L}(\theta) = \frac{1}{T} \sum_t (y_t - \hat{y}_t)^2. \quad (\text{A.10})$$

T is the total number of frames. y_t is the target acoustic feature at time t while \hat{y}_t is the predicted one. In stimulated DNN, the new training criterion becomes:

$$\mathcal{F}(\theta) = \mathcal{L}(\theta) + \eta_{s_c} \mathcal{R}_{s_c}(\theta), \quad (\text{A.11})$$

where the regularization function $\mathcal{R}_{s_c}(\theta)$ is defined as:

$$\mathcal{R}_{s_c}(\theta) = \frac{1}{T} \sum_t \sum_l \sum_i g(\mathbf{s}_i, \hat{\mathbf{s}}_c) \log \left(\frac{g(\mathbf{s}_i, \hat{\mathbf{s}}_c)}{\bar{h}_i^l} \right). \quad (\text{A.12})$$

A.3.1 Experiments on network visualization

CMU ARCTIC corpus was used for experiment. There are seven speakers in the corpus, including five male speakers (awb, bdl, imk, ksp, rms) and two female speakers (clb, slt). Six speakers (bdl, imk, ksp, rms, clb, slt) were selected to train the speaker independent model. To train the speaker independent model, 900 utterances from each of the six speakers were selected as training set, 100 utterances for validation, and 50 utterances were used as testing data for multi-speaker synthesis. Transcriptions were common among all speakers and the testing transcriptions were not covered in training or validation set.

The input to neural network includes linguistic features and speaker code and gender code. Speaker code is one-hot vector [84]. Gender code is a one-dimensional mark for male or female. To train the network, linguistic features were extracted and converted to 307 dimensional vectors. State index and frame index were also attached. State-level alignment was done with the help of HTS [45]. The outputs of network were acoustic features including 39-dimensional mel-cepstral coefficients, F0 in log-scale, 26-dimensional Band-aperiodicity parameters (BAP), and their delta and delta-delta features. A binary value for voiced/unvoiced decision was also attached. Linear interpolation of F0 was done over unvoiced segments. Min-max normalization were performed over both input and output features. MLPG and GV algorithms [124] were used for post-processing.

The baseline multi-speaker synthesis system was trained with normal criterion as showed in equation A.10. Network consists of four layers and each layer has 1024 nodes. The learning rate is initialized as 1e-3, and automatically increased according to the variation of validation error. The momentum is set as 0.3 at beginning and improved to 0.9 after 15 epochs. Stimulated DNN is setup using similar network structure except for the training criterion.

To setup the stimulated training criterion, utterance level i-vectors of each speaker are firstly mapped to the 2-D grid via t-SNE and the results are shown in Fig. A.2. To extract i-vectors, a gender independent 2048-mixture UBM and 30-dimension total variability matrices are trained with EM algorithm, using NIST SRE corpora (2004, 2006, 2008), Switchboard II Phase 1/2/3, Switchboard Cellular I/II and the CMU ARCTIC corpora [14, 61–65]. The mapped utterance-level i-vectors are shown in A.2.

The activation grids of the first hidden layer of stimulated DNNs was compared on speaker “jmk” and “slt”. All the nodes are trained with stimuli and Fig. A.3 shows that the activation grids corresponded to the stimulating pattern well: the nodes around the mapped positions of each speaker’ utterance-level i-vectors echoed

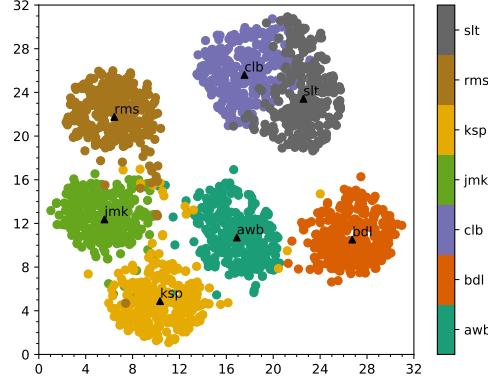


Figure A.2: 2-dimensional mapping of utterance level i-vectors

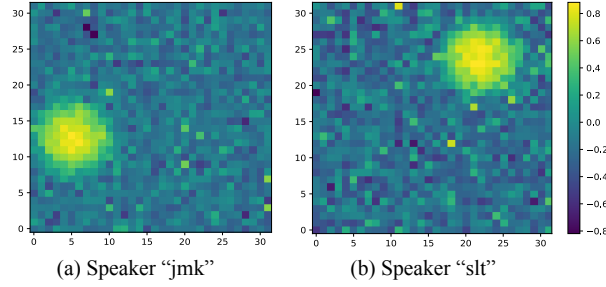


Figure A.3: Stimulated activations for speaker 'jmk' and 'slt'

higher activation values. The distribution of activation values for different speakers can be easily distinguished from each other.

Another phenomenon is observed from Fig. A.4, which shows the stimulated activation output in different layers. Layer 1 is the bottom layer which near to the input and layer 4 is the top layer which near to the output. The bright region is enhanced from the first to last layer, which is consistent with our expectation: stimulated training can help to enhance the difference of speaker identity in top layers.

In the regularization function described in equation A.12, the larger the η_{sc} is, the smaller the effect of regularization. Different η_{sc} values for multi-speaker synthesis

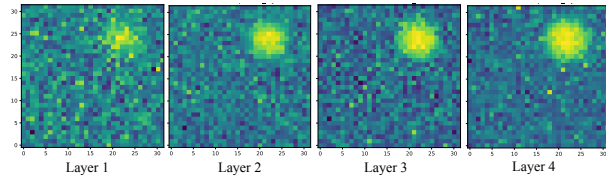


Figure A.4: Stimulated output activation in different layers

Table A.1: Objective evaluations for multi-speaker synthesis on the testing data of speaker "slt".

Systems	η_{s_c}	MCD (dB)	F_0 RMSE(Hz)	BAP (dB)	V/UV err(%)
Baseline	-	6.38	23.41	4.28	7.09
Stimulated	0.05	6.32	23.20	4.27	6.90
	0.10	6.32	23.28	4.29	6.93
	0.15	6.36	23.21	4.35	7.02
	0.2	6.37	23.48	4.25	7.35

task have been tested. Table A.1 reports the distortion for speaker "slt". The stimulated system outperforms the normal system in most cases. The lowest distortion of MGC and F0 appears when $\eta_{s_c} = 0.05$.

A.4 Conclusion

This work has tried to solve to interpret network activation in terms of speaker identity and improve the multi-speaker modeling with stimulated training. Preliminary experimental results demonstrate that the proposed strategy can yield visible speaker dependent activation regions, and the performance of multi-speaker modeling can be slightly improved by stimulated training.

Bibliography

- [1] H. Zen, “Deep learning in speech synthesis,” *Keynote speech given at ISCA SSW8*, 2013.
- [2] A. J. Hunt and A. W. Black, “Unit selection in a concatenative speech synthesis system using a large speech database,” in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 1, pp. 373–376, IEEE, 1996.
- [3] H. Zen, K. Tokuda, and A. W. Black, “Statistical parametric speech synthesis,” *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.
- [4] H. Zen, A. Senior, and M. Schuster, “Statistical parametric speech synthesis using deep neural networks,” in *ICASSP*, pp. 7962–7966, IEEE, 2013.
- [5] Y. Fan, Y. Qian, F. Xie, and F. K. Soong, “Tts synthesis with bidirectional lstm based recurrent neural networks,” in *Interspeech*, pp. 1964–1968, 2014.
- [6] H. Zen and H. Sak, “Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis,” in *ICASSP*, pp. 4470–4474, IEEE, 2015.
- [7] 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.
- [8] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, and Y. Bengio, “Char2wav: End-to-end speech synthesis,” 2017.
- [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] S. O. Arik, G. Diamos, A. Gibiansky, J. Miller, K. Peng, W. Ping, J. Raiman, and Y. Zhou, “Deep voice 2: Multi-speaker neural text-to-speech,” *arXiv preprint arXiv:1705.08947*, 2017.
- [11] W. Ping, K. Peng, A. Gibiansky, S. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, “Deep voice 3: Scaling text-to-speech with convolutional sequence learning,” in *Proc. 6th International Conference on Learning Representations*, 2018.
- [12] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, *et al.*, “Tacotron: Towards end-to-end speech synthesis,” *arXiv preprint arXiv:1703.10135*, 2017.
- [13] R. Skerry-Ryan, E. Battenberg, Y. Xiao, Y. Wang, D. Stanton, J. Shor, R. J. Weiss, R. Clark, and R. A. Saurous, “Towards end-to-end prosody transfer for expressive speech synthesis with tacotron,” *arXiv preprint arXiv:1803.09047*, 2018.
- [14] J. Kominek and A. W. Black, “The cmu arctic speech databases,” in *Fifth ISCA Workshop on Speech Synthesis*, 2004.
- [15] Y. Taigman, L. Wolf, A. Polyak, and E. Nachmani, “Voiceloop: Voice fitting and synthesis via a phonological loop,” in *Proc. 6th International Conference on Learning Representations*, 2018.
- [16] Y. Wang, D. Stanton, Y. Zhang, R. Skerry-Ryan, E. Battenberg, J. Shor, Y. Xiao, F. Ren, Y. Jia, and R. A. Saurous, “Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis,” *arXiv preprint arXiv:1803.09017*, 2018.
- [17] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, p. 436, 2015.
- [18] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.
- [19] R. Maia, H. Zen, and M. J. Gales, “Statistical parametric speech synthesis with joint estimation of acoustic and excitation model parameters,” in *SSW*, pp. 88–93, 2010.

- [20] K. Nakamura, K. Hashimoto, Y. Nankaku, and K. Tokuda, “Integration of acoustic modeling and mel-cepstral analysis for hmm-based speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 7883–7887, IEEE, 2013.
- [21] H. Zen, M. J. Gales, Y. Nankaku, and K. Tokuda, “Product of experts for statistical parametric speech synthesis,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 3, pp. 794–805, 2012.
- [22] L.-H. Chen, T. Raitio, C. Valentini-Botinhao, J. Yamagishi, and Z.-H. Ling, “Dnn-based stochastic postfilter for hmm-based speech synthesis,” in *INTER-SPEECH*, pp. 1954–1958, 2014.
- [23] Z.-H. Ling, L. Deng, and D. Yu, “Modeling spectral envelopes using restricted boltzmann machines for statistical parametric speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 7825–7829, IEEE, 2013.
- [24] Z.-H. Ling, L. Deng, and D. Yu, “Modeling spectral envelopes using restricted boltzmann machines and deep belief networks for statistical parametric speech synthesis,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 10, pp. 2129–2139, 2013.
- [25] S. Kang, X. Qian, and H. Meng, “Multi-distribution deep belief network for speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 8012–8016, IEEE, 2013.
- [26] R. Fernandez, A. Rendel, B. Ramabhadran, and R. Hoory, “F0 contour prediction with a deep belief network-gaussian process hybrid model,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 6885–6889, IEEE, 2013.
- [27] Y. Qian, Y. Fan, W. Hu, and F. K. Soong, “On the training aspects of deep neural network (dnn) for parametric tts synthesis,” in *ICASSP*, pp. 3829–3833, IEEE, 2014.
- [28] H. Kei, O. Keiichiro, N. Yoshihiko, and T. Keiichi, “The effect of neural networks in statistical parametric speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, IEEE, 2015.

- [29] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, “Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, IEEE, 2015.
- [30] Y. Saito, S. Takamichi, and H. Saruwatari, “Statistical parametric speech synthesis incorporating generative adversarial networks,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 1, pp. 84–96, 2018.
- [31] H. Kawahara, “Straight, exploitation of the other aspect of vocoder: Perceptually isomorphic decomposition of speech sounds,” *Acoustical science and technology*, vol. 27, no. 6, pp. 349–353, 2006.
- [32] 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.
- [33] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, “Saplernn: An unconditional end-to-end neural audio generation model,” *arXiv preprint arXiv:1612.07837*, 2016.
- [34] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, *et al.*, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” *arXiv preprint arXiv:1712.05884*, 2017.
- [35] R. Van Bezooijen and L. C. Pols, “Evaluating text-to-speech systems: Some methodological aspects,” *Speech Communication*, vol. 9, no. 4, pp. 263–270, 1990.
- [36] N. Campbell, “Evaluation of speech synthesis,” in *Evaluation of text and speech systems*, pp. 29–64, Springer, 2007.
- [37] V. Grancharov and W. B. Kleijn, “Speech quality assessment,” in *Springer Handbook of Speech Processing*, pp. 83–100, Springer, 2008.
- [38] E. Vincent, “Mushram: A matlab interface for mushra listening tests,” *Online* <http://www.elec.qmul.ac.uk/people/emmanuelv/mushram>, 2005.

- [39] Y. Fan, Y. Qian, F. K. Soong, and L. He, “Multi-speaker modeling and speaker adaptation for dnn-based tts synthesis,” in *ICASSP*, pp. 4475–4479, IEEE, 2015.
- [40] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, “A study of speaker adaptation for dnn-based speech synthesis,” in *Interspeech*, 2015.
- [41] L. Sun, S. Kang, K. Li, and H. Meng, “Voice conversion using deep bidirectional long short-term memory based recurrent neural networks,” 2014.
- [42] Y. Miao and F. Metze, “On speaker adaptation of long short-term memory recurrent neural networks,” in *Interspeech*, 2015.
- [43] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
- [44] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Front-end factor analysis for speaker verification,” *Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.
- [45] K. Oura, “List of modifications made in hts (for version 2.2 beta),” 2011.
- [46] S. Imai, T. Kobayashi, K. Tokuda, T. Masuko, K. Koishida, S. Sako, and H. Zen, “Speech signal processing toolkit (sptk), version 3.3,” 2009.
- [47] F. Weninger, “Introducing currennt: The munich open-source cuda recurrent neural network toolkit,” *Journal of Machine Learning Research*, vol. 16, pp. 547–551, 2015.
- [48] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, “Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis,” in *ICASSP*, pp. 4460–4464, IEEE, 2015.
- [49] Y. Q. Yuchen Fan, F. K. Soong, and L. He, “Unsupervised speaker adaptation for dnn-based tts synthesis,” in *ICASSP*, pp. 5135–5139, IEEE, 2016.
- [50] B. Potard, P. Motlicek, and D. Imseng, “Preliminary work on speaker adaptation for dnn-based speech synthesis,” tech. rep., Idiap, 2015.
- [51] J. Yamagishi and T. Kobayashi, “Average-voice-based speech synthesis using hsmm-based speaker adaptation and adaptive training,” *IEICE Transactions on Information and Systems*, vol. 90, no. 2, pp. 533–543, 2007.

- [52] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, “A compact model for speaker-adaptive training,” in *Spoken Language*, vol. 2, pp. 1137–1140, IEEE, 1996.
- [53] C. Leggetter and P. C. Woodland, “Speaker adaptation of continuous density hmms using multivariate linear regression,” in *ICSLP*, vol. 94, pp. 451–454, Citeseer, 1994.
- [54] O. Abdel-Hamid and H. Jiang, “Fast speaker adaptation of hybrid nn/hmm model for speech recognition based on discriminative learning of speaker code,” in *ICASSP*, pp. 7942–7946, IEEE, 2013.
- [55] J. T. Zhiying Huang, S. Xue, and L.-R. Dai, “Speaker adaptation of rnn-blstm for speech recognition based on speaker code,” in *ICASSP*, pp. 7942–7946, IEEE, 2016.
- [56] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, “Direct adaptation of hybrid dnn/hmm model for fast speaker adaptation in lvcsr based on speaker code,” in *ICASSP*, pp. 6339–6343, IEEE, 2014.
- [57] S. Xue, H. Jiang, L. Dai, and Q. Liu, “Unsupervised speaker adaptation of deep neural network based on the combination of speaker codes and singular value decomposition for speech recognition,” in *ICASSP*, pp. 4555–4559, IEEE, 2015.
- [58] S.-i. Horikawa, T. Furuhashi, and Y. Uchikawa, “On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm,” *IEEE transactions on Neural Networks*, vol. 3, no. 5, pp. 801–806, 1992.
- [59] A. Graves, N. Jaitly, and A.-R. Mohamed, “Hybrid speech recognition with deep bidirectional lstm,” in *Proceedings of ASRU*, pp. 273–278, 2013.
- [60] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne, “Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds,” *Speech communication*, vol. 27, no. 3, pp. 187–207, 1999.
- [61] N. Brümmer, “The spescom data voice nist sre 2004 system,” in *NIST SRE 2004 Workshop, Toledo, Spain*, 2004.

- [62] M. A. Przybocki, A. F. Martin, and A. N. Le, “Nist speaker recognition evaluation chronicles-part 2,” in *Speaker and Language Recognition Workshop*, pp. 1–6, IEEE, 2006.
- [63] A. Strasheim and N. Brümmer, “Sunsdv system description: Nist sre 2008,” in *NIST Speaker Recognition Evaluation Workshop Booklet*, 2008.
- [64] D. Graff, K. Walker, and A. Canavan, “Switchboard-2 phase ii,” *LDC 99S79*–<http://www.ldc.upenn.edu/Catalog>, 1999.
- [65] D. Graff, K. Walker, and D. Miller, “Switchboard cellular part 2,” *LDC 2004S07*–<https://catalog.ldc.upenn.edu/LDC2004S07>, 2004.
- [66] R. F. Kubichek, “Mel-cepstral distance measure for objective speech quality assessment,” in *Communications, Computers and Signal Processing*, vol. 1, pp. 125–128, IEEE, 1993.
- [67] 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, pp. IV–1229, IEEE, 2007.
- [68] H. Zen and A. Senior, “Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 3844–3848, IEEE, 2014.
- [69] D. Erro, I. Sainz, E. Navas, and I. Hernaez, “Harmonics plus noise model based vocoder for statistical parametric speech synthesis,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 2, pp. 184–194, 2014.
- [70] Y. Agiomyrgiannakis, “Vocaine the vocoder and applications in speech synthesis,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 4230–4234, IEEE, 2015.
- [71] F. Espic, C. Valentini-Botinhao, and S. King, “Direct modelling of magnitude and phase spectra for statistical parametric speech synthesis,” *Proc. Interspeech, Stochohlm, Sweden*, 2017.
- [72] Y. Saito, S. Takamichi, and H. Saruwatari, “Training algorithm to deceive anti-spoofing verification for dnn-based speech synthesis,” in *Acoustics, Speech*

- and *Signal Processing (ICASSP)*, *2017 IEEE International Conference on*, pp. 4900–4904, IEEE, 2017.
- [73] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, pp. 2672–2680, 2014.
 - [74] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, *et al.*, “Parallel wavenet: Fast high-fidelity speech synthesis,” *arXiv preprint arXiv:1711.10433*, 2017.
 - [75] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, “Speaker-dependent wavenet vocoder,” in *Proceedings of Interspeech*, pp. 1118–1122, 2017.
 - [76] X. Wang, J. Lorenzo-Trueba, S. Takaki, L. Juvela, and J. Yamagishi, “A comparison of recent waveform generation and acoustic modeling methods for neural-network-based speech synthesis,” *arXiv preprint arXiv:1804.02549*, 2018.
 - [77] Y.-C. Wu, K. Kobayashi, T. Hayashi, P. L. Tobing, and T. Toda, “Collapsed speech segment detection and suppression for wavenet vocoder,” *arXiv preprint arXiv:1804.11055*, 2018.
 - [78] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.
 - [79] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, “Improved training of wasserstein gans,” in *Advances in Neural Information Processing Systems*, pp. 5769–5779, 2017.
 - [80] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, “Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications,” *arXiv preprint arXiv:1701.05517*, 2017.
 - [81] T. Lei and Y. Zhang, “Training rnns as fast as cnns,” *arXiv preprint arXiv:1709.02755*, 2017.
 - [82] N. Hojo, Y. Ijima, and H. Mizuno, “An investigation of dnn-based speech synthesis using speaker codes,” in *Interspeech*, pp. 2278–2282, 2016.

- [83] H.-T. Luong, S. Takaki, G. E. Henter, and J. Yamagishi, “Adapting and controlling dnn-based speech synthesis using input codes,” in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pp. 4905–4909, IEEE, 2017.
- [84] Y. Zhao, D. Saito, and N. Minematsu, “Speaker representations for speaker adaptation in multiple speakers’ blstm-rnn-based speech synthesis,” in *Inter-speech*, pp. 2268–2272, 2016.
- [85] B. Li and H. Zen, “Multi-language multi-speaker acoustic modeling for lstm-rnn based statistical parametric speech synthesis,” in *INTERSPEECH*, pp. 2468–2472, 2016.
- [86] T. Hayashi, A. Tamamori, K. Kobayashi, K. Takeda, and T. Toda, “An investigation of multi-speaker training for wavenet vocoder,” in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*, pp. 712–718, IEEE, 2017.
- [87] 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.
- [88] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan,” *arXiv preprint arXiv:1701.07875*, 2017.
- [89] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.*, “Conditional image generation with pixelcnn decoders,” in *Advances in Neural Information Processing Systems*, pp. 4790–4798, 2016.
- [90] T. L. Paine, P. Khorrami, S. Chang, Y. Zhang, P. Ramachandran, M. A. Hasegawa-Johnson, and T. S. Huang, “Fast wavenet generation algorithm,” *arXiv preprint arXiv:1611.09482*, 2016.
- [91] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [92] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, pp. 6000–6010, 2017.

- [93] Wikipedia, “Prosody (linguistics).” [https://en.wikipedia.org/wiki/Prosody_\(linguistics\)](https://en.wikipedia.org/wiki/Prosody_(linguistics)).
- [94] J. Sun, J. Yang, J. Zhang, and Y. Yan, “Chinese prosody structure prediction based on conditional random fields,” in *Proceedings of ICNC*, vol. 3, pp. 602–606, 2009.
- [95] F. chiang Chou, C. yu Tseng, and L. shan Lee, “Automatic generation of prosodic structure for high quality mandarin speech synthesis,” in *Proceedings of ICSLP*, vol. 3, pp. 1624–1627, IEEE, 1996.
- [96] J. H. Jeon and Y. Liu, “Automatic prosodic events detection using syllable-based acoustic and syntactic features,” in *Proceedings of ICASSP*, pp. 4565–4568, 2009.
- [97] V. Rangarajan, S. Narayanan, and S. Bangalore, “Exploiting acoustic and syntactic features for prosody labeling in a maximum entropy framework,” in *Proceedings of NAACL HLT*, pp. 1–8, 2007.
- [98] P. Koehn, S. Abney, J. Hirschberg, and M. Collins, “Improving intonational phrasing with syntactic information,” in *Proceedings of ICASSP*, pp. 1289–1290, 2000.
- [99] M. Chu and Y. Qian, “Locating boundaries for prosodic constituents in unrestricted mandarin texts,” *Computational linguistics and Chinese language processing*, pp. 61–82, 2001.
- [100] X. Nie and Z.-y. Wang, “Automatic phrase break prediction in chinese sentences,” *Journal of Chinese information Processing*, pp. 39–44, 2003.
- [101] J.-F. Li, G. Hu, and R.-h. Wang, “Chinese prosody phrase break prediction based on maximum entropy model,” in *Proceedings of INTERSPEECH*, pp. 729–732, 2004.
- [102] G.-A. Levow, “Automatic prosodic labeling with conditional random fields and rich acoustic features,” in *Proceedings of IJCNLP*, pp. 217–224, 2008.
- [103] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proceedings of 18th ICML*, pp. 282–289, 2001.

- [104] M. Nobuaki, K. Shumpei, S. Shinya, and H. Keikichi, “Improved prediction of japanese word accent sandhi using crf,” in *Proceedings of INTERSPEECH*, vol. 10, pp. 114–783, 2012.
- [105] Y. Qian, Z. Wu, X. Ma, and F. Soong, “Automatic prosody prediction and detection with conditional random field (crf) models,” in *Proceedings of ISCSLP*, pp. 135–138, 2010.
- [106] C. Ding, L. Xie, J. Yan, W. Zhang, and Y. Liu, “Automatic prosody prediction for chinese speech synthesis using blstm-rnn and embedding features,” in *Proceedings of ASRU*, IEEE, 2015.
- [107] A. Rosenberg, R. Fernandez, and B. Ramabhadran, “Modeling phrasing and prominence using deep recurrent learning,” in *Proceedings of INTERSPEECH*, 2015.
- [108] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, “A neural probabilistic language model,” *The Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.
- [109] X. Zheng, H. Chen, and T. Xu, “Deep learning for chinese word segmentation and pos tagging,” in *Proceedings of EMNLP*, pp. 647–657, 2013.
- [110] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, “Natural language processing (almost) from scratch,” *The Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.
- [111] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, “Learning sentiment-specific word embedding for twitter sentiment classification,” in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 1555–1565, 2014.
- [112] Y. Goldberg and O. Levy, “word2vec explained: deriving mikolov et al.’s negative-sampling word-embedding method,” *arXiv preprint arXiv:1402.3722*, 2014.
- [113] A. Evgeniou and M. Pontil, “Multi-task feature learning,” *Advances in neural information processing systems*, vol. 19, p. 41, 2007.
- [114] W. Che, Z. Li, and T. Liu, “Ltp: A chinese language technology platform,” in *Proceedings of Coling 2010:Demonstrations*, pp. 13–16, ACL, 2010.

- [115] T. Kudo, “Crf++: Yet another crf toolkit,” *Software available at <http://crfpp.sourceforge.net>*, 2005.
- [116] A. Ragni, C. Wu, M. J. F. Gales, J. Vasilakes, and K. M. Knill, “Stimulated training for automatic speech recognition and keyword search in limited resource conditions,” in *ICASSP*, pp. 4830–4834, IEEE, 2017.
- [117] S. Tan, K. C. Sim, and M. Gales, “Improving the interpretability of deep neural networks with stimulated learning,” in *ASRU*, pp. 617–623, IEEE, 2015.
- [118] D. G. Garson, “Interpreting neural network connection weights,” 1991.
- [119] A. Goh, “Back-propagation neural networks for modeling complex systems,” *Artificial Intelligence in Engineering*, vol. 9, no. 3, pp. 143–151, 1995.
- [120] A. Nguyen, J. Yosinski, and J. Clune, “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,” in *CVPR*, pp. 427–436, 2015.
- [121] C. Wu, P. Karanasou, M. J. Gales, and K. C. Sim, “Stimulated deep neural network for speech recognition,” in *Interspeech*, pp. 400–404, 2016.
- [122] L. v. d. Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [123] J. Tang, J. Liu, M. Zhang, and Q. Mei, “Visualization large-scale and high-dimensional data,” *arXiv preprint [arXiv:1602.00370](https://arxiv.org/abs/1602.00370)*, 2016.
- [124] T. Tomoki and K. Tokuda, “A speech parameter generation algorithm considering global variance for hmm-based speech synthesis,” *IEICE TRANSACTIONS on Information and Systems*, vol. 90, no. 5, pp. 816–824, 2007.

Dedication

There are a number of people that I want to express my gratitude. They guided me, placed opportunities in front of me, and showed me the doors that might be useful to open. Without them, this thesis might not have been written.

Firstly, I would like to express my sincere gratitude to my advisor: Professor NOBUAKI MINEMATSU for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I am also very grateful to his kindness and support in my study career.

My sincere thanks also goes to Prof. Saito for the stimulating discussions, for his insightful comments and encouragement, but also for the hard question which encouraged me to widen my research from various perspectives.

I would like to thank Prof. Yamagishi, who provided me an opportunity to join their team as intern, and who gave access to the laboratory and research facilities. Without his precious support it would not be possible to conduct this research.

I thank my fellow lab mates for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years. Also I thank my friends for enlightening me the first glance of research.

Last but not the least, I would like to thank my parents for supporting me spiritually throughout writing this thesis and my my life in general.

List of Publications

Journal Paper

1. Yi Zhao, Shinji Takaki, Hieu-Thi LUONG, Yamagishi Junichi, Daisuke Saito, Minematsu Nobuaki. Wasserstein GAN and Waveform Loss-based Acoustic Model Training for Multi-speaker Text-to-Speech Synthesis Systems Using a WaveNet Neural Vocoder. (Submitted to IEEE Access on 06/2018)

International Conference Paper

1. Zhao Y, Saito D, Minematsu N. Speaker Representations for Speaker Adaptation in Multiple Speakers' BLSTM-RNN-based Speech Synthesis[J]. Interspeech, 2016, 5(6): 7.
2. Zhao Y, Ding C, Minematsu N, et al. A Study on BLSTM-RNN-based Chinese Prosodic Structure Prediction in a Unified Framework with Character-level Features[J]. Speech Prosody 2016: 64-68.
3. Zhao Y, You X, Saito D, et al. The UTokyo System for Blizzard Challenge 2016[J].

National Conference Paper

1. Zhao, Y., Minematsu, N., Saito, D. Integration of Multi-Speaker Training and Speaker Adaptation for DBLSTM-RNN-based Text-To-Speech Synthesis. 2016 Spring Meeting Acoustical Society of Japan. pp.339-340
2. Zhao, Y., Ding, C., Minematsu, N., Saito, D. A Study on BLSTM-RNN-based Chinese Prosodic Structure Prediction. 2016 Spring Meeting Acoustical Society of Japan. pp. 217-218
3. Zhao, Y., Minematsu, N., Saito, D. Multi-speaker speech synthesis and speaker adaptation based on deep bidirectional long short-term memory recurrent neural network. IEICE technique report 2015, 2015(19): 1-6.