INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Global Syllable Vectors for Building TTS Front-End with Deep Learning

Jinfu Ni, Yoshinori Shiga, and Hisashi Kawai

Advanced Speech Technology Laboratory, ASTREC,
National Institute of Information and Communications Technology, Japan
{jinfu.ni, yoshinori.shiga, hisashi.kawai}@nict.go.jp

Abstract

Recent vector space representations of words have succeeded in capturing syntactic and semantic regularities. In the context of text-to-speech (TTS) synthesis, a front-end is a key component for extracting multi-level linguistic features from text, where syllable acts as a link between low- and high-level features. This paper describes the use of global syllable vectors as features to build a front-end, particularly evaluated in Chinese. The global syllable vectors directly capture global statistics of syllable-syllable co-occurrences in a large-scale text corpus. They are learned by a global log-bilinear regression model in an unsupervised manner, whilst the front-end is built using deep bidirectional recurrent neural networks in a supervised fashion. Experiments are conducted on large-scale Chinese speech and treebank text corpora, evaluating grapheme to phoneme (G2P) conversion, word segmentation, part of speech (POS) tagging, phrasal chunking, and pause break prediction. Results show that the proposed method is efficient for building a compact and robust front-end with high performance. The global syllable vectors can be acquired relatively cheaply from plain text resources, therefore, they are vital to develop multilingual speech synthesis, especially for under-resourced language modeling.

Index Terms: Front-end, global syllable vectors, G2P, deep neural networks, text-to-speech, multilingual speech synthesis

1. Introduction

This work explores raw text to produce global syllable vectors as features for building a front-end of text-to-speech (TTS) synthesis, inspired by recent advances in global word vector representation [1] and distributed representation learning [2] as well as their applications in natural language processing (NLP) [3], opinion mining [4], and speech synthesis [5]. TTS synthesis maps input text to speech waveform output [6]. The mapping is bridged in terms of features based on linguistic knowledge, such as phonemes, syllables, prosodic words, intonational phrases, etc. A front-end accepts text as input, extracts such linguistic features from the text, and returns a linguistic specification of the utterance to be synthesized. Although the popular statistical parametric speech synthesis (SPSS) learns speech models from speech data using a well-defined objective function [7], the linguistic specification still is the basis for the context-dependent modeling [8][9]. Therefore, a front-end is important for achieving high-quality speech synthesis. Conventionally, it analyzes the text input, using some established sets of linguistic features. However, obtaining the knowledge and data required to establish appropriate sets of linguistic features is expensive and sometimes is difficult in multilingual speech synthesis [10], especially for under-resourced languages, such as Myanmar [11].

An alternative approach to construction of linguistic specification is to build a vector space model (VSM)-based front-end [12]. VSMs, originally from the field of NLP, are built on the co-occurrence statistics of words which are gathered in matrix form to produce high-dimensional representations of the distributional behavior of words [13]. Lower dimensional representations are obtained by approximately factorizing the matrix of raw co-occurrence counts using slim singular value decomposition (SVD) [13]. The main advantage of the VSM-based model over the traditional decision tree and CRF (conditional random fields)-based methods is that the objective distance between VSM output values directly represents the similarity of two units. Moreover, VSMs are learned in an unsupervised fashion from text; no labeled data is required.

Recently an open source for learning VSMs, called GloVe [1] originally proposed for NLP tasks, is available that is based on global log-bilinear regression. GloVe combines the advantages of two major model families in the literature: global matrix factorization and local context window methods [1]. It efficiently leverages statistical information by training only on the nonzero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or individual context windows.

This paper describes the use of GloVe [1] upon a large-scale text corpus to produce a VSM for global vector representations of syllables, named global syllable vectors (GSVs). GSVs directly capture the global statistics of syllable-syllable co-occurrences in plain text resources. Moreover, motivated by the recent success of deep architectures in recurrent neural networks (RNNs) in particular, we explore an application of deep bidirectional RNNs to build a front-end with global syllable vectors. Here syllable is taken as the unit type for two reasons. First, syllable is a link between low-level linguistic features like phonemes and high-level ones like intonational phrases (viewed as a group of syllables). Second, it can be acquired relatively cheaply from plain text like Chinese directly, or only depending on the availability of syllable breaking of text scripts in some languages like Myanmar [11]. We conduct benchmarking experiments on LDC treebank [14] and in-house speech corpora in Chinese to evaluate the method for building a front-end, including grapheme to phoneme (G2P) conversion, word segmentation, part of speech (POS) tagging, phrasal chunking, and pause break prediction. Also, we evaluate CRF-baselines for comparison. The basic motivation is to build a front-end combining unsupervised GSV, given a large-scale text corpus, and supervised learning methods. Questions arise: if labeled data are available, whether a GSV-based front-end can complete the conventional tasks in high accuracy with compact and robust models. Otherwise whether global syllable vectors themselves are sufficient for capturing the characteristics and dependencies of syllables in text; this is interesting in context of deep neural network based speech synthesis.

In the rest of the paper, Section 2 outlines the method combining global syllable vectors with deep bidirectional RNNs for classification and discusses related work. Section 3 presents experiments and results. Section 4 concludes this paper.

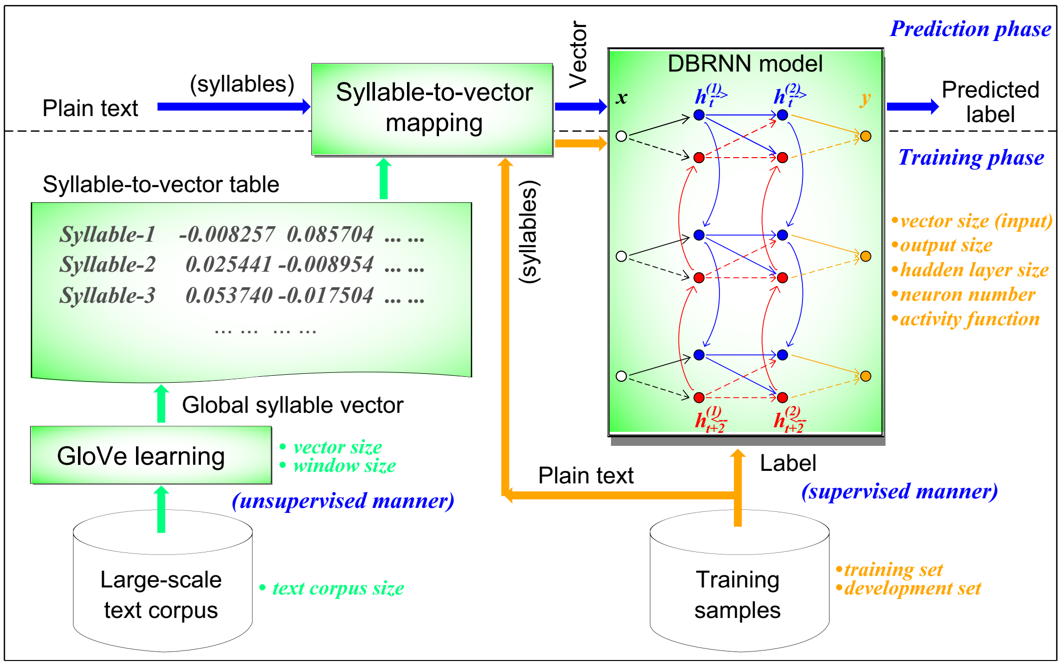

Figure 1: *Schematic diagram of DBRNN-based label prediction with global syllable vectors. Thick arrows indicate information streams during the period of learning global syllable vectors (green color), training a DBRNN model (orange), and evaluation (blue), respectively. In the box of DBRNN model, each node denotes an input (open circle), hidden (blue and red), or output (yellow) layer, respectively. Solid and dashed lines denote the connections of forward and backward layers, respectively.*

2. Deep learning with global syllable vectors

The novel part of the method described here is shown by the bright color blocks in Fig. 1. Generally speaking, two functions are achieved: syllable-to-vector mapping and deep neural network-based prediction. The syllable-to-vector mapping by looking up a syllable-vector table provides the network's input layer with syllable vectors as features. The syllable vectors are constructed offline by GloVe [1] which is used to learn global vector representations of syllables based on a large-scale text corpus. On the other hand, the bulk of label prediction used here is based on deep bidirectional recurrent neural networks (DBRNNs) [4][16].

2.1. Deep bidirectional RNN-based prediction

A RNN [15] is a class of neural network that has certain memory. This makes it suitable for the tasks in question. For a task like pause break prediction, a single input sentence is viewed as analyzing a sequence of tokens (i.e., syllables here). A bidirectional RNN incorporates information from preceding as well as following tokens [16]. At each time step, t, this network maintains two hidden layers, one for the forward propagation and another for the backward propagation. Moreover, deep bidirectional RNNs, having multiple stacked hidden layers, are known to naturally employ a temporal hierarchy with multiple layers operating at different time scales [2]: lower levels capture short term interactions among tokens; higher layers reflect interactions aggregated over longer spans of text [4].

Figure 1 shows a two-hidden layer bidirectional RNN. As shown in this figure, at time-step t each intermediate neuron receives one set of the previous time-step (in the same RNN layer), and two sets of parameters from the previous RNN hidden layer: One input comes from the forward RNN and the other from the backward RNN. More formally [4],

$$\overrightarrow{h}_t^{(1)} = f(\overrightarrow{W}^{(1)} x_t + \overrightarrow{V}^{(1)} \overrightarrow{h}_{t-1}^{(1)} + \overrightarrow{b}^{(1)}) \quad (1)$$

$$\overleftarrow{h}_t^{(1)} = f(\overleftarrow{W}^{(1)} x_t + \overleftarrow{V}^{(1)} \overleftarrow{h}_{t-1}^{(1)} + \overleftarrow{b}^{(1)}) \quad (2)$$

$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}_{\rightarrow}^{(i)} \overrightarrow{h}_t^{(i-1)} + \overrightarrow{W}_{\leftarrow}^{(i)} \overleftarrow{h}_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)}) \quad (3)$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}_{\rightarrow}^{(i)} \overrightarrow{h}_t^{(i-1)} + \overleftarrow{W}_{\leftarrow}^{(i)} \overleftarrow{h}_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t-1}^{(i)} + \overleftarrow{b}^{(i)}) \quad (4)$$

when $i > 1$. $\boldsymbol{h}$ stands for hidden representations, $\boldsymbol{W}$ and $\boldsymbol{V}$ are the weight matrices, $\boldsymbol{b}$ the bias vectors, $\boldsymbol{x}_t$ the input vectors. $f(x)$ is activation function which is chosen according to tasks.

$$f(x) = \begin{cases} \max\{0, x\} & \text{rectified linear unit, ReLU(x)} \\ (e^x - e^{-x})/(e^x + e^{-x}) & \text{tanh function, } \tanh(x) \end{cases}$$

The network output, $\boldsymbol{y}$, at each time-step is

$$\boldsymbol{y}_t = g(U_{\rightarrow} \overrightarrow{h}_t^{(L)} + U_{\leftarrow} \overleftarrow{h}_t^{(L)} + c) \quad (5)$$

$$g(x) = e^{x_i} / \sum_j e^{x_j} \quad (6)$$

where L is the number of layers, $\boldsymbol{U}$ is the weight matrix, and c the bias. Prediction at time-step t is based on $argmax(\boldsymbol{y}_t)$.

2.2. Learning global syllable vectors

The statistics of syllable occurrences in a text corpus is the primary source of information available to unsupervised methods for learning syllable representations.

Let X denote the matrix of syllable-syllable co-occurrence counts, and X_{ij} the number of times for syllable j occurring in the context of syllable i within a n-syllable symmetric window. GloVe generates two sets of d-dimensional syllable vectors, S and $\tilde{S}$, by minimizing the following cost function [1].

Table 1: *Experimental setup. A token is viewed as a sequence of syllables labeled using SBIE tagging scheme: S indicates a syllable-wise token. B, I, and E label syllable at the beginning of a token, inside the token, and at the end of the token, respectively.*

Task	**Training set**	**Development**	**Evaluation**	**Hidden layer**	**Activation**	**Output size**
G2P	44,122 sens	2,985 sens	3,765 sens	2	tanh	1,390 (pinyins)
Word segmentation	94,862 sens	10,075 sens	11,755 sens	2	ReLU	4 (B, I, E, S)
POS tagging	94,862 sens	10,075 sens	11,755 sens	2	tanh	44 (LDC tags)
Phrase chunk prediction	80,811 sens	10,637 sens	11,764 sens	2	tanh	2 (Phrase, O(ther))
Pause break prediction	29,561 utts	3,182 utts	6,079 utts	2	ReLU	2 (Pause break, O)

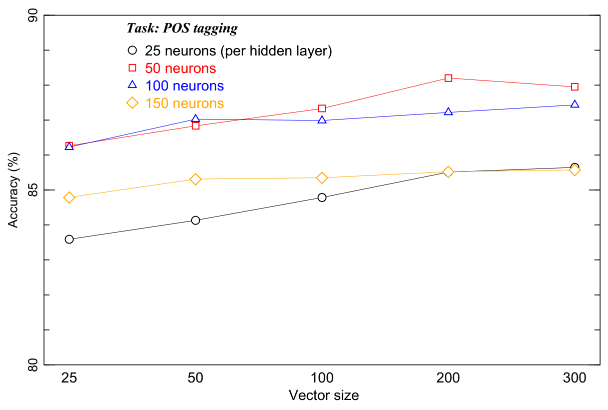

Figure 2: *Accuracy for the POS tagging task as function of vector size and neuron number of a hidden layer. All the global syllable vectors are learned with a 20-syllable window.*

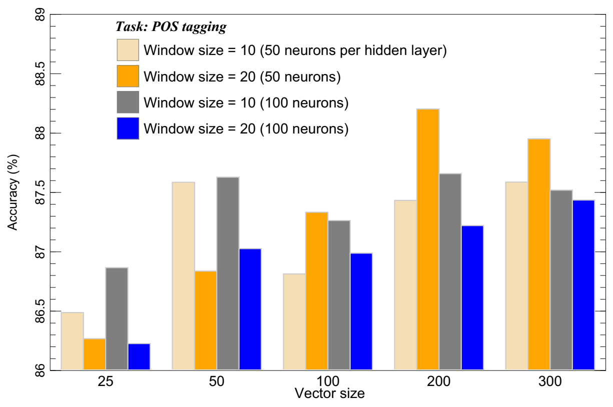

Figure 3: *Accuracy for the POS tagging task as function of vector size, window size, and neuron number of a hidden layer.*

$$J = \sum_{i,j=1}^{V} q(X_{ij})(\boldsymbol{s}_i^T \tilde{\boldsymbol{s}}_j + b_i + \tilde{b}_j - logX_{ij})^2, \qquad (7)$$

where V is the size of corresponding syllable vocabulary, $\boldsymbol{s}_i \in \Re^d$ are syllable vectors and $\tilde{\boldsymbol{s}}_j \in \Re^d$ are separate context syllable vectors, b_i and $\tilde{b}_j$ are bias, respectively. $q(X_{ij})$ is weighting function to avoid frequent co-occurrence overweighted [1].

$$q(x) = \begin{cases} (x/x_{\max})^\alpha & \text{if } x < x_{\max} \\ 1 & \text{otherwise.} \end{cases} \qquad (8)$$

In the work, $x_{\max} = 100$ and $\alpha = 3/4$ as used in [1]. Because X is symmetric here, the two sets of syllable vectors, S and $\tilde{S}$ are equivalent and differ only as a result of their random initializations. Global syllable vectors are $\boldsymbol{s}_i + \tilde{\boldsymbol{s}}_i, i = 1, ..., V$.

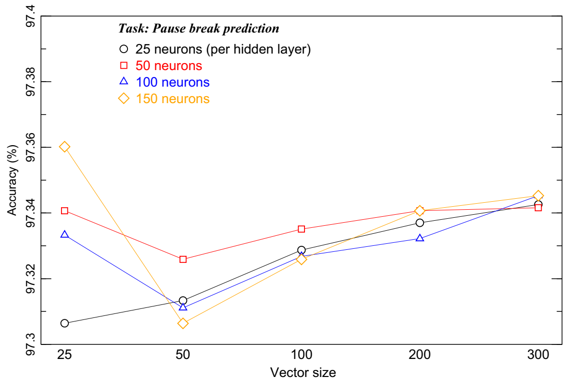

Figure 4: *Accuracy for the pause break prediction task as function of vector size and neuron number of a hidden layer. All the global syllable vectors are learned with a 20-syllable window.*

2.3. Related work

The use of VSMs to represent word and letter types was proposed in [13][12]. Their methods were based on matrix factorization methods for generating a low-dimensional word (or letter) representation with SVD. They used CART (classification and regression tree [17]) and DNN (deep neural networks) to implement a front-end with VSMs. On the other hand, the use of deep learning in TTS has become widespread in acoustic modeling [18][9] as well as being applied to individual tasks related to a front-end (e.g. letter-to-sound conversion [19] and phrase-break prediction [5]).

As mentioned previously, our method bears some resemblance to those [13][12] such as the use of VSMs for building a front-end in that all these methods allow to learn VSMs from plain text in an unsupervised fashion. In contrast to them, we employ GloVe [1] to learn VSMs that is based on the global log-bilinear regression model. Because GloVe combines the advantages of both global matrix factorization and local context window methods, global syllable vectors (GSVs) are trained only on the nonzero elements in a syllable-syllable co-occurrence matrix, rather than on the entire sparse matrix. Consequently, GloVe-based VSMs, or GSVs, directly capture the global corpus statistics.

3. Experiments and results

3.1. Experimental setup

To evaluate the method, we use GloVe [1] to learn global syllable vectors and then conduct benchmarking experiments on LDC treebank [14] and in-house speech corpora in Chinese. Text is split into a sequence of syllables; a Chinese char is a syllable. The basic unit for prediction is syllable when training models for the following tasks in a supervised manner.
(1) G2P (converting text to standard pronunciation in pinyin).

Table 2: *Comparison of compact DBRNN models with global syllable vectors to CRF baselines for five tasks of a front-end.*

Task	**Syllable vector size**	**Hidden layer**	**Units per hidden layer**	**Accuracy**	**CRF baseline**
G2P	50	2	100	**99.05**	98.15
Word segmentation	50	2	100	94.31	**95.05**
POS tagging	50	2	100	87.33	**90.30**
Phrase chunk prediction	50	2	100	**94.61**	93.96
Pause break prediction	50	2	100	**97.34**	97.09

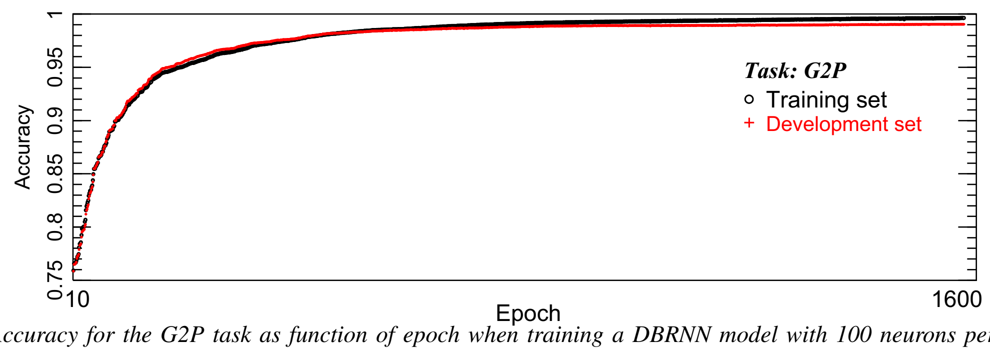

Figure 5: *Accuracy for the G2P task as function of epoch when training a DBRNN model with 100 neurons per hidden layer and 50-dimensional vectors for syllable representation as features.*

(2) Word segmentation (segmenting text into words).
(3) Part of speech (POS) tagging (associating syllables with their surface words' POS).
(4) Phrase chunking (predicting main syntactic phrases here).
(5) Pause break prediction (inserting pause breaks into text).

Task (1) is conducted on a set of sentences with pinyin labels checked manually by natives. Tasks (2) to (4) are conducted on labeled data extracted from LDC Chinese treebank 9.0 [14]. Task (5) is conducted on reading speech corpora amounting to 38 hour speech by four native narrators. Table 1 tabulates the experimental setup in more details. The sample sets for training, development and evaluation are disjoint.

GSV learning A large-scale text corpus used here for learning global syllable vectors (GSVs) comprises news and other text resources amounting to 35 million syllables. The size of vocabulary V is 5,610 (unique syllables). We train GSVs with varying vector sizes (25, 50, 100, 200, and 300) and window sizes (10 and 20 syllables) in the experiments.

Network training We use the standard multiclass cross-entropy as the objective function when training the neural networks. We use stochastic gradient descent with momentum with a fixed learning rate (0.005) and a fixed momentum (0.9). We update weights after mini-batches of 20 sentences. Weights are initialized from small random uniform noise. We train networks of various sizes (25, 50, 100, and 150 neurons per hidden layer) with maximum 2000 epochs each network, however we have the same number of hidden units across multiple forward and backward hidden layers of a single RNN. Additionally, we employ early stopping for the neural networks: out of all iterations, the model with the best performance for the development set is selected as the final model to be evaluated.

CRF baseline As a baseline, we use CRF [20]. Features used in CRF input are plain Chinese chars (within a [-2, +2] context window); a Chinese char is a syllable.

3.2. Results

The results are shown in Table 2 and Figs. 2–5. A few observations can be made from the results.

- The method of global syllable vectors with deep bidirectional RNNs achieved high performance almost for all the tasks (see Table 2) and the resulting models are compact and robust. In Table 2, the results were achieved by 50-dimensional GSVs and 2-hidden layer BRNN with 100 neurons for each forward and backward layer.
- Vector size affected the resulting performances slightly. There was a trend for most tasks that the larger the vector size, the better performance the prediction task (see Figs. 2 and 4). Generally speaking, 50 to 100 dimensional vectors are good enough for the purpose of building a front-end.
- The effects of different window size (10 or 20 syllables here) for producing GSVs on the resulting performances were not significant (see Fig. 3).
- The global syllable vectors are efficient for capturing the meaning of syllables in context. This is clearly demonstrated in Fig. 5. Among 1,390 candidates a pinyin was chosen with 99% accuracy for a syllable, which was represented by a 50-dimensional vector.

It is concluded from the benchmarking experiments that GloVe-based global syllable vectors are sufficient for representing the characteristics and dependencies of syllables in text.

4. Conclusion

We have shown that GloVe-based global syllable vectors as features completed the main tasks of a front-end well with deep learning. The global syllable vectors are learned from plain text resources in an unsupervised fashion. This is motivated by a desire to relatively easily build a multilingual front-end for multilingual speech synthesis including under-resourced languages.

Future work will investigate the use of global syllable vectors as features for a direct specification of utterances to be synthesized with deep learning, instead of generating a conventional linguistic specification for the utterances.

5. References

[1] J. Pennington, R. Socher, and C. D. Manning, 2014, "GloVe: Global Vectors for Word Representation," *http://nlp.stanford.edu/projects/glove/.*

[2] M. Hermans and B. Schrauwen, "Training and Analysing Deep Recurrent Neural Networks," in *Advances in Neural Information Processing Systems*, pp. 190–198, 2013.

[3] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. P. Kuksa, "Natural Language Processing almost from Scratch," *CoRR*, vol. abs/1103.0398, 2011.

[4] O. Irsory and C. Cardie, "Opinion Mining with Deep Recurrent Neural Networks," in *Proc. the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 720-728, 2014.

[5] O. Watts, S. Gangireddy, J. Yamagishi, S. King, S. Renals, A. Stan, and M. Giurgiu, "Neural Net Word Representations for Phrase-break Prediction Without a Part of Speech Tagger," in *Proc. ICASSP*, pp. 2618–2622, 2014.

[6] H. Kawai, *et al.*, "XIMERA: A Concatenative Speech Synthesis System with Large Scale Corpora," *IEICE Trans. Inf. & Syst.,*, vol. J89-D, no. 12, pp. 2688–2698, 2006.

[7] H. Zen, K. Tokuda, and A. Black, "Statistical Parametric Speech Synthesis," *Speech Communication*, **51** (11), pp. 1039–1064, 2009.

[8] H. Zen, A. Senior, and M. Schuster, "Statistical Parametric Speech Synthesis Using Deep Neural Networks," in *Proc. ICASSP*, pp. 7962–7966, 2013.

[9] Z.-H. Ling, S.-Y. Kang, H. Zen, A. Senior, M. Schuster, X.-J. Qian, H. M. Meng, and L. Deng, "Deep Learning for Acoustic Modeling in Parametric Speech Generation: A Systematic Review of Existing Techniques and Future trends," *IEEE Signal Processing Magazine*, vol. 32, no. 3m pp. 35-52, 2015.

[10] Y. Shiga and H. Kawai, "Multilingual Speech Synthesis System," *Journal of the National Institute of Information and Communications Technology*, vol. 59, nos 3/4, 2012.

[11] Y. K. Thu, W. P. Pa, J. Ni, Y. Shiga, A. Finch, C. Hori, H. Kawai, and E. Sumita. "HMM-based Myanmar Text-to-Speech System," in *Proc. INTERSPEECH2015*, Germany, 2015.

[12] H. Lu, S. King, and O. Watts, "Combining a Vector Space Representation of Linguistic Context with a Deep Neural Network for Text-to-Speech," in *Proc. the 8th ISCA Speech Synthesis Workshop (SSW)*, pp. 281-285, 2013.

[13] O. Watts, "Unsupervised Learning for Text-to-Speech synthesis," *Ph.D. dissertation*, University of Edinburgh, 2012.

[14] N. Xue, X. Zhang, Z. Jiang, M. Palmer, F. Xia, F.-D. Chiou, M. Chang, "Chinese Treebank 9.0," *https://catalog.ldc.upenn.edu/ldc2016t13*.

[15] J. L. Elman, "Finding Structure in Time," *Cognitive Science*, 14(2), pp. 179–211, 1990.

[16] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Transactions on Signal Processing*, 45(11), pp. 2673–2681, 1997.

[17] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and Regression Trees," *Chapman and Hall*, 1993.

[18] Y. Fan, Y. Qian, F. Xie, and F. K. Soong, "TTS Synthesis with Bidirectional LSTM based Recurrent Neural Networks," in *Proc. INTERSPECH2014*, pp. 1964-1968, 2014.

[19] K. J. Jensen and S. Riis, "Self-organizing Letter Code-book for Text-to-phoneme Neural Network Model," in *Proc. INTERSPEECH2000*, pp. 318-321, 2000.

[20] CRF++: Yet Another CRF Toolkit, *https://taku910.github.io/crfpp/*