# Word Level Prosody Prediction Using Large Audiobook Dataset

Yanfeng Lu, Chenyu Yang and Minghui Dong
Human Language Technology Department
Institute for Infocomm Research, A-Star, Singapore
{luyf, yangc, mhdong}@i2r.a-star.edu.sg

***Abstract*—Prosody modelling is an essential part of the text-to-speech synthesis system. In this paper we propose and investigate a way to leverage public domain audiobook data to do word level prosody modelling. Specifically we base our work on the LibriSpeech project, in which a large quantity of public domain audiobook data from LibriVox were processed, selected and aligned with text. We choose long-short-term-memory recurrent deep neural network as the modelling tool. The input word features spread from phonetic, through syntactic, to semantic layers. The word prosody features include log F0, energy and after-word break. A way of incorporating the word prosody model into the speech synthesis system is also proposed. Experiments show that it is an effective way to leverage large quantity and variety of speech data to do prosody modelling for speech synthesis.**

## I. Introduction

A text-to-speech (TTS) synthesis system converts a piece of free text in a certain language into the corresponding speech in the format of waveform. Prosody is an integral part of any speech. It not only determines the naturalness of the speech, but also influences its intelligibility. Therefore, prosody is a crucial factor that each TTS system has to consider. Prosody is a speech property, but during the synthesis stage only text is given. Hence we need to predict prosodic features on the basis of linguistic features extracted from the text. To achieve that we need to do prosody modelling using existing speech data.

The authors of [1] gave a rather comprehensive survey of prosody modelling techniques in TTS systems. But there have been further developments in recent years. Specifically, the authors didn't include deep neural network modelling in their paper at that time. And this was a major recent development.

Some researchers adopted rule-based approach. In this case the mapping from linguistic to prosodic features was embodied in a set of rules, which were manually created. For instance, in [2], the author predicted intonational phrases based on $\phi$-phrases, which was essentially a syntactic structure, with a set of hand-crafted rules. In contrast, the authors of [3] performed statistical prosody modelling based on rich syntactic context. The context was in turn based on the syntactic tree. In this case the mapping from linguistic to prosodic features was embodied in hidden Markov models (HMMs) and established through statistical training on the basis of existing speech data. Recently the mapping from linguistic to prosodic features has been embodied in a deep neural network, which was trained with existing speech data. In [4], the authors conducted prosody boundary modelling on the basis of word embedding with bidirectional long-short-term-memory (LSTM) recurrent neural network (RNN), and compared it with other methods. The experiments showed the superiority of the deep neural network approach.

Apparently each of the above three approaches has its advantages and disadvantages. The rule-based approach has no dependency on speech data, but it's difficult to capture all the complexity of the mapping with predefined rules. The HMM approach saves the human effort to figure out the rules, but it requires balanced speech data. From a biased speech dataset can only be generated a skewed model. It has been more and more demonstrated recently that deep neural network is a more powerful modelling tool than HMM. But training a deep neural network is trickier. Besides having a much higher demand for researchers' experience and skills, it's really data hungry. A more complex neural network could provide a more accurate modelling, but it needs more data to train.

Due to the exceptional power of deep neural network, it has been the trend to move toward the third approach in prosody modelling. As a result, obtaining speech data becomes more and more crucial. Creating speech database in studio settings is a very costly process. Fortunately collecting large quantity of non-studio speech data has become much easier in this internet age. The major issue turns out to be how to leverage these big chunks of data. In this paper, we propose a way to leverage the data collected from the internet to do prosody modelling for TTS systems.

Our study is based on the LibriSpeech project [5]. In the project 1000 hours of speech from LibriVox[1], the volunteer, public domain audiobook website, were processed with automatic speech recognition (ASR) tools. The speech data were selected and then aligned with text. The authors detailed the processing in their paper. And the resulting corpus was made public. Although the corpus was intended to be used in the ASR projects, it definitely can be used for TTS too. It's used as a base corpus for the study in this paper. However, due to the different nature of TTS than ASR, we have to do further filtering of the speech data.

When we humans read a text aloud, through reflection we learn, the prosody of the speech is to a large extent determined by the words as units. One plausible explanation is that, words

[1] https://librivox.org

are semantic units and meaning is an important prosodic factor. As the first step, in this study we focus on word level features. Different from previous studies, we use a comprehensive set of word linguistic features, spreading over the phonetic, syntactic and semantic layers. Compared with speech data recorded in studio settings, those obtained from the internet, such as LibriVox, have a much bigger variety, in terms of speakers, genres and quality. We hope this variety could help build a more generic model. So a comprehensive feature set and a diverse database are the two unique characteristics of this study. Besides the word prosody modelling itself, we also propose a way to incorporate the model into the TTS systems.

The rest of the paper is organized as follows: In Section II we describe the further processing we performed on the LibriSpeech corpus. Then the details of the word prosody modelling is presented in Section III. Section IV shows how we incorporate the word prosody modelling into existing TTS systems. In Section V we report the experiments we did with our proposals. And in the concluding section we summarize our work and suggest relevant future studies.

## II. Further Processing of the Audiobook Data

The LibriSpeech corpus consists of several subsets. For the sake of quality we only pick the subsets that are labelled "clean". They specifically include dev-clean, test-clean, train-clean-100 and train-clean-360.

### A. Preliminary Processing

Since we need to generate the syntactic tree in computing the linguistic features, it's desirable that for each of the speech pieces we have the corresponding raw text with punctuation. However, the transcriptions that come with the Librispeech corpus are just word sequences. So one part of the preliminary processing is to match the raw text in the original books with the transcriptions provided. The clean subsets in the LibriSpeech corpus have higher quality, with lower recognition WER. In our case we only select the segments that have an exact word match, which means zero WER. With this we actually apply another round of filtering. The resulting speech data are about 200 hours. Another part of the processing is to convert the speech files provided into 16K Hertz 16bits wave format.

### B. Word Alignment

For the alignment we use our internal tool that's based on a neural network model trained with Kaldi ASR toolkit [6]. The wave files and the corresponding raw texts obtained above are used as inputs of the alignment tool. The tool outputs phone level alignment. The word level alignment can be easily computed out of the phone level alignment.

### C. Full Context Label Generation

In the field of speech synthesis, the linguistic features of phones are called full context labels. In our study, the full context labels are generated with the front end of our TTS engine. We generally adopt the features listed in [7] and add the syntactic features based on the syntactic tree as suggested in [8]. In principle, the full context labels are based on the text. But in our engine we also factor in the phone level alignment above. In this way the phone sequence for a certain word will be determined by the ASR result, instead of being arbitrarily picked from the dictionary. Silences are also inserted according to the alignment.

### D. Prosody Feature Extraction

For the acoustic prosody features we choose F0 and energy. They are extracted from the wave files using the Praat tool [9]. The frame size is set to 10ms. The pitch range is set between 50 and 500 Hertz.

## III. Word Prosody Modelling

As we mentioned above, prosody modelling is essential to establishing the mapping from linguistic features computed from the text to prosodic features that characterize the speech. The selection of the features is as important as the modelling itself.

### A. Word Linguistic Features

For our word prosody modelling task we want to select a comprehensive set of word linguistic features. Computational linguistics has involved phonetics, syntax and semantics. For the word features we want to include all the three linguistic layers. In particular, the features we select are listed as follows:

- **Word Phonetic Features**: number of syllables of the previous, current and next word.
- **Word Syntactic Features**: POS of the previous, current and next word; number of words in the utterance; features based on the syntactic tree (phrase type and depth of the father and grandfather phrases of the previous, current and next word; position of the current word in the father phrase).
- **Word Semantic Features**: word embedding.

The phonetic and syntactic features above are all contained in our full context labels. Therefore, they can be easily extracted from the later. For all the categorical features we expand them into one-hot vectors. For word embedding we use the *word2vec* tool [10] to train a corpus and use the training results. The dimension of the word embedding is set to 200. All the component vectors are concatenated to form a general linguistic vector.

### B. Word Prosody Features

Pitch and loudness are two important factors in prosody. They are determined by F0 and energy. Due to the special nature of human auditory perception, log F0 is normally used instead of F0. The F0 and energy are extracted from the wave files frame by frame. A word contains many frames. So to capture the word prosody we select some statistical properties of the frame log F0 and energy sequences that are contained in the word. They are mean, variance, max and min. To capture the pitch dynamics we also include the statistical properties of the velocity and acceleration of log F0. Besides pitch and

loudness, the duration of the after-word break is another word prosody feature. Thus, we have a 17-dimensional prosody vector.

### C. Word Prosody Feature Normalization

Considering the big variety of the LibriSpeech data in terms of speakers and genres, we have to do certain normalization of the prosody features, in order to obtain a consistent model. To do this, we compute the mean of F0 and energy by chapters, which belong to the same audiobook read by the same speaker, and then factor them in as extra inputs. In this way the variance in F0 and energy could be offset.

In a studio database, normally only one speaker and one genre are involved. So when the model is applied in this case, the mean F0 and energy can be computed across the whole database and then used as part of the inputs.

### D. The Modelling Tool: LSTM-RNN

LSTM-RNN was first introduced in [11]. It has been proven to be an effective modelling tool of sequential signal in many recent studies, [12] and [4] for instance. The core of the architecture is a long-short-term memory which can selectively store contextual information. The memory is encapsulated in an LSTM cell. An LSTM-RNN normally consists of multiple layers, which contain an array of LSTM cells. The structure of the neural network we use is depicted in Figure 1. The word linguistic features are treated as input to a feed forward neural network. The output of the feed forward neural network is then passed to an LSTM-RNN. The LSTM-RNN outputs the word prosody features directly.

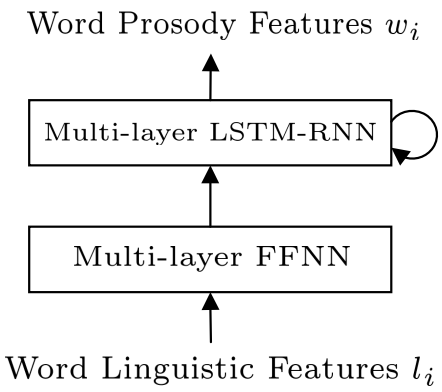


Fig. 1. Structure of the Neural Network Used for Word Prosody Modelling

Certain processing is performed on the linguistic and prosody features before they are used in the training. Categorical linguistic features are expanded into one-hot vectors. All the input and output vectors are normalized. After the processing, the word linguistic feature input has 720 dimensions and the word prosody feature output keeps 17 dimensions.

We use the Computational Network Toolkit (CNTK) [13] to train our network. Root squared error is the loss function. We start the training with a relatively bigger learning rate and decrease it each time there is no improvement. We also start the next epoch with the best model. The training ends either when the learning rate falls below a threshold or when 200 epochs have completed.

## IV. Incorporating the Word Prosody Model into the TTS Systems

In this study we consider both parametric and unit selection-concatenation TTS systems. But both systems are based on frame level LSTM-RNN acoustic modelling. The inputs to the LSTM-RNN are full context label vectors. The output acoustic features include spectrum, log F0 and aperiodicity. The phone duration is modelled separately. This constitutes the training phase. In the synthesis phase, full context label vectors are computed out of the target text first. Then the phone durations are predicted. Based on the phone durations, acoustic features are predicted frame by frame. The two types of TTS systems diverge at this stage. In the parametric system the predicted acoustic features are passed to the vocoder to generate the target waveform. But in the unit selection-concatenation system, the predicted phone durations and frame acoustic features are used to compute the target cost. We use the Merlin Toolkit [14] for the acoustic modelling and the parametric waveform generation. But the unit selection and concatenation are built by ourselves. It follows the general ideas of unit selection based on target and concatenation costs [15] and trajectory tiling [16]. Figure 2 shows the general architecture of the two types of TTS systems.

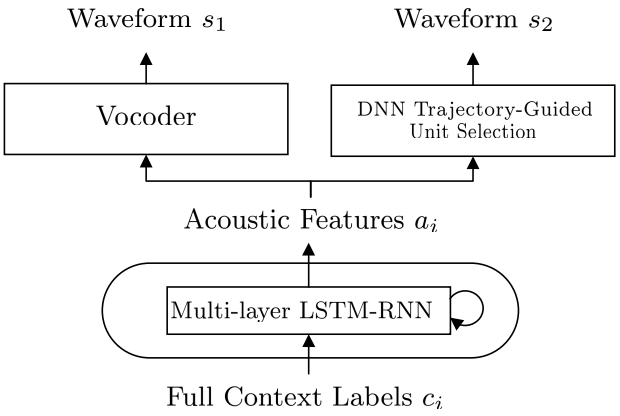


Fig. 2. Architecture of the Two Types of TTS Systems

With the architecture of the TTS systems clarified, the method to incorporate the word prosody model is easy to discuss. A straightforward way is to concatenate the predicted word prosody features with the full context label features and then use the concatenated features to train the acoustic models. In this way we combine the word prosody modelling on the basis of the large audiobook database and the acoustic modelling on the basis of the studio database. This method is demonstrated in Figure 3. Note the word prosody features are predicted by the model trained with the audiobook database.

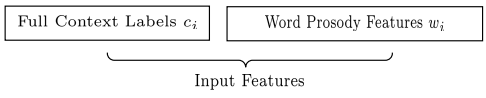


Fig. 3. Combine Word Prosody Modelling with Acoustic Modelling

Adding the word prosody model enhancement, we have 4 types of TTS systems. For easy reference, they are listed in Table I.

TABLE I
FOUR TYPES OF TTS SYSTEMS

| | |
|---|---|
| TTS-P | Parametric TTS system based on LSTM-RNN acoustic modelling |
| TTS-P-E | System TTS-P enhanced with word prosody model |
| TTS-C | Concatenated TTS system based on LSTM-RNN acoustic modelling |
| TTS-C-E | System TTS-C enhanced with word prosody model |

## V. EXPERIMENTS

We do some experiments to investigate the effectiveness of our proposal. The first part is to find out whether volunteer, public domain audiobook data taken from the internet are useful for modelling word prosody. The second part is to find out whether the prosody model obtained is helpful for the TTS systems.

### A. Experiment Settings

Two speech datasets are used in the experiments. One is the LibriSpeech dataset (Set-LS) as we pre-processed in Section II. It contains around 45K speech segments, many of which include multiple sentences. Dozens of speakers, both female and male, are involved. With so many speakers the accent also vary a lot. The other dataset (Set-SLIC[2]) was created in our own lab. It only involves a female speaker. The texts selected are mostly neutral. Most of the pieces contain a single sentence.

We divide the LibriSpeech dataset into two parts. 10% of the segments are randomly selected to make the testing subset (Set-LS-Test). All the rest are used as the training subset (Set-LS-Train). We also use our own dataset for several purposes. 17K pieces are used as the training subset (Set-SLIC-Train), both for the word prosody modelling and the TTS system building. 971 pieces are used as the testing subset (Set-SLIC-Test1) for objective evaluation. Other 500 pieces are used as the testing subset (Set-SLIC-Test2) for subjective evaluation of the 4 TTS systems.

The 5 data subsets are listed in Table II, so that they can be easily referred below.

[2]"SLIC" is our project code name.

TABLE II
DATASETS USED IN THE EXPERIMENTS

| Dataset | Data Subsets | Description |
|---|---|---|
| LibriSpeech (Set-LS) | Set-LS-Train | 90% of around 45K segments, for prosody model training |
| | Set-LS-Test | 10% of around 45K segments, for prosody model testing |
| SLIC (Set-SLIC) | Set-SLIC-Train | 17K pieces, for prosody model training and TTS system building |
| | Set-SLIC-Test1 | 971 pieces, for prosody model testing |
| | Set-SLIC-Test2 | 500 pieces, for TTS system testing |

### B. Experiment Steps and Results

In designing the experiment steps and analyzing the experiment results we focus on investigating the effectiveness and usefulness of word prosody modelling with large audiobook data. Generally we compare the word prosody model trained with the audiobook dataset and that trained with a normal studio recorded dataset, and also the baseline TTS systems and word prosody enhanced TTS systems. In comparing the word prosody we use the averaged Euclidean Distance (AED) between the predicted feature vector and the actual feature vector as the measure. In comparing the wave files generated with various TTS systems we use the Mean Opinion Score (MOS).

As data preparation, the text and speech data in both datasets, Set-LS and Set-SLIC, are processed as described in Sections II & III. These include generating phone and word alignments, full context labels, word linguistic feature vectors and word prosody feature vectors. Then the experiments are conducted in the follow steps:

*1) Comparing Word Prosody Models on the Same Type of Data:* We train the word prosody models with the LibriSpeech dataset and our own dataset, and test them on separate data taken from the same dataset. The experiment is conducted in the following specific steps:

- **Word prosody training with Set-LS-Train**: The data in Set-LS-Train are first used to train the network depicted in Figure 1. From this we obtain a model Model-LS.
- **Word prosody testing with Set-LS-Test**: The model Model-LS is tested with LibriSpeech data Set-LS-Test.
- **Word prosody training with Set-SLIC-Train**: The data in Set-SLIC-Train are then used to train the same network. We obtain another model Model-SLIC.
- **Word prosody testing with Set-SLIC-Test1**: The model Model-SLIC is tested with SLIC data Set-SLIC-Test1.

The results are shown in Table III. Since the word prosody features are computed in the same way and the distance measure is common, the prediction results are still comparable, even though different test sets are involved.

The table tells us that the model trained with LibriSpeech data (Model-LS) is much better than the model trained with normal TTS data (Model-SLIC) when tested on the same type of data. The major difference between the two types of data

TABLE III
COMPARISON OF WORD PROSODY MODELS ON THE SAME TYPE OF DATA

| Testing Configuration | AED |
|---|---|
| Model-LS tested on Set-LS-Test | 1.724 |
| Model-SLIC tested on Set-SLIC-Test1 | 3.933 |

lies in diversity. The LibriSpeech data is much more diverse than our data. This may partly explain the superiority of the former. More diverse data may better represent the various speech cases. So it helps to obtain a more accurate model. Another factor is the size of the dataset. The size of the LibriSpeech dataset we selected is about 10 times the size of our dataset.

*2) Comparing Word Prosody Models on the SLIC Dataset:* Next we examine how the LibriSpeech model fares on SLIC data. This will tell us to how much extent Model-LS is a "generic" model. So we also test Model-LS on Set-SLIC-Test1. The result is compared with Model-SLIC in Table IV.

TABLE IV
COMPARISON OF WORD PROSODY MODELS ON SLIC DATA

| Testing Configuration | AED |
|---|---|
| Model-SLIC tested on Set-SLIC-Test1 | 3.933 |
| Model-LS tested on Set-SLIC-Test1 | 3.414 |

From these results we can see, Model-LS is better than Model-SLIC even when tested on SLIC data. This indicates that the model trained with LibriSpeech data is a reliable generic model.

*3) Comparing Adapted and Baseline Models:* Now we want to see whether the adaptation training could make the word prosody model even better. Specifically we conduct the following steps:

- **Word prosody model adaptation**: In this case we initialize the network with Model-LS and then use Set-SLIC-Train to further train it. The resulting model is labelled Model-LS-SLIC.
- **Adaptation model testing**: At this step Model-LS-SLIC is also tested with Set-SLIC-Test1.

The testing result of the adaptation model is shown in Table V.

TABLE V
COMPARISON OF ADAPTED AND BASELINE MODELS

| Testing Configuration | AED |
|---|---|
| Model-LS tested on Set-SLIC-Test1 | 3.414 |
| Model-LS-SLIC tested on Set-SLIC-Test1 | 3.322 |

With the adaptation we do get some improvement in the prosody model. This means, when further trained with a particular dataset, the generic model is adapted to capture some particular characteristic of the speaker of the dataset.

*4) TTS Enhancement with Word Prosody Model:*

- **TTS system building**: A parametric and a unit selection-concatenation TTS system are built with Set-SLIC-Train. In building the parametric TTS system we use the Merlin Toolkit. The unit selection-concatenation system is based on the predicted acoustic features, but the core mechanism is built by ourselves. We label the parametric TTS system TTS-P and the other system TTS-C.
- **Test wave generation**: Wave files are generated from both TTS systems with the test set Set-SLIC-Test2.
- **Word prosody model enhanced TTS systems**: We enhance the above TTS systems with the word prosody model Model-LS, trained with the LibriSpeech data. In doing this, word prosody in Set-SLIC-Train is predicted with Model-LS. Although Table V shows some improvement in the adapted model Model-LS-SLIC, it requires further word prosody processing in the studio database and also further training. In contrast, Model-LS could be trained once and used for all the other databases. Therefore, we use Model-LS directly to do enhancement. The predicted word prosody vectors are then concatenated to the full context label vectors to do the acoustic modelling. This constitutes the only difference between the enhanced systems and the base systems. We label the enhanced systems TTS-P-E and TTS-C-E, respectively.
- **Listening test**: Finally we conduct a listening test to evaluate the generated wave files from various TTS systems subjectively. MOS is adopted as the measure in the listening test. 20 sentences are randomly selected from Set-SLIC-Test2. For each of the sentence we include 5 wave files. 4 of them are generated from the 4 TTS systems under evaluation: TTS-P, TTS-C, TTS-P-E and TTS-C-E. For a good reference we also include the natural wave files recorded in our studio by the same speaker. The 20 sentences in turn are presented to several listeners. For each sentence the wave files are presented randomly. The listeners are requested to give a MOS score for each of the wave files.

The results of the listening test are displayed in Figure 4. The figure shows the boxplot of MOS data for different TTS systems. For each of the systems we combine the MOS data of all the sentences given by all the listeners together. According to the figure, with the word prosody enhancement the TTS systems show some improvement, not as much as the objective evaluation.

## VI. CONCLUSIONS

In this study we try to leverage a large audiobook dataset to do word prosody modelling. By comparing it with a normal studio dataset, the model trained with the audiobook dataset can make better prediction. Based on the objective evaluation, it not only performs better on similar data, but also on studio recorded speech data. Besides, it performs even better when adapted to the new data. We also propose a way to incorporate the word prosody model into TTS systems. Although this straightforward way doesn't show significant effect in the

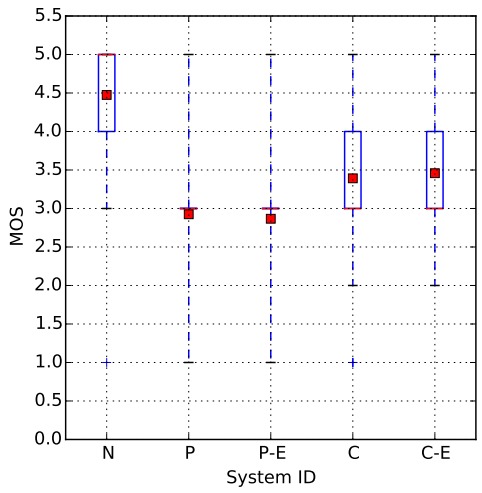


Fig. 4. MOS Scores of Wave Files Generated from Different TTS Systems N: Natural speech; P: TTS-P; C: TTS-C; P-E: TTS-P-E; C-E: TTS-C-E.

subjective testing, the prosody modelling itself is still effective as demonstrated in the objective evaluation. The prosody features themselves and the way of incorporating the prosody model into the TTS system could be improved later. And with the audiobook data other types of modelling could also be experimented in the future.

## REFERENCES

[1] K. Rajeswari and M. Uma, "Prosody modeling techniques for text-to-speech synthesis systems–a survey," *International Journal of Computer Applications*, vol. 39, no. 16, pp. 8–11, 2012.

[2] M. Atterer, "Assigning prosodic structure for speech synthesis: a rule-based approach," in *Speech Prosody 2002, International Conference*, 2002.

[3] Y. Yu, D. Li, and X. Wu, "Prosodic modeling with rich syntactic context in hmm-based mandarin speech synthesis," in *Signal and Information Processing (ChinaSIP), 2013 IEEE China Summit & International Conference on*. IEEE, 2013, pp. 132–136.

[4] 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 *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 98–102.

[5] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5206–5210.

[6] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[7] H. Zen, "An example of context-dependent label format for hmm-based speech synthesis in english," *The HTS CMUARCTIC demo*, vol. 133, 2006.

[8] Y. Yu, F. Zhu, X. Li, Y. Liu, J. Zou, Y. Yang, G. Yang, Z. Fan, and X. Wu, "Overview of shrc-ginkgo speech synthesis system for blizzard challenge 2013," in *Blizzard Challenge Workshop*, vol. 2013, 2013.

[9] P. P. G. Boersma *et al.*, "Praat, a system for doing phonetics by computer," *Glot international*, vol. 5, 2002.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "word2vec," 2014.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[13] D. Yu, A. Eversole, M. Seltzer, K. Yao, Z. Huang, B. Guenter, O. Kuchaiev, Y. Zhang, F. Seide, H. Wang *et al.*, "An introduction to computational networks and the computational network toolkit," *Microsoft Technical Report MSR-TR-2014–112*, 2014.

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

[15] 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. IEEE, 1996, pp. 373–376.

[16] Y. Qian, F. K. Soong, and Z.-J. Yan, "A unified trajectory tiling approach to high quality speech rendering," *IEEE transactions on audio, speech, and language processing*, vol. 21, no. 2, pp. 280–290, 2013.