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

Prosody Aware Word-level Encoder Based on BLSTM-RNNs for DNN-based Speech Synthesis

Yusuke Ijima, Nobukatsu Hojo, Ryo Masumura, Taichi Asami

NTT Media Intelligence Laboratories, NTT Corporation.

ijima.yusuke@lab.ntt.co.jp

Abstract

Recent studies have shown the effectiveness of the use of word vectors in DNN-based speech synthesis. However, these word vectors trained from a large amount of text generally carry not prosodic information, which is important information for speech synthesis, but semantic information. Therefore, if word vectors that take prosodic information into account can be obtained, it would be expected to improve the quality of synthesized speech. In this paper, to obtain word-level vectors that take prosodic information into account, we propose a novel prosody aware word-level encoder. A novel point of the proposed technique is to train a word-level encoder by using a large speech corpus constructed for automatic speech recognition. A word-level encoder that estimates the F0 contour for each word from the input word sequence is trained. The outputs of the bottleneck layer in the trained encoder are used as the word-level vector. By training the relationship between words and their prosodic information by using large speech corpus, the outputs of the bottleneck layer would be expected to contain prosodic information. The results of objective and subjective experiments indicate the proposed technique can synthesize speech with improved naturalness.

Index Terms: Speech synthesis, DNN-based speech synthesis, prosodic information, word-level encoder

1. Introduction

Recent research on statistical parametric speech synthesis [1], especially deep neural network (DNN)-based one [2, 3], has achieved a remarkable breakthrough with the great developments of generative models in image generation. For instance, Generative Adversarial Nets (GAN) [4] and PixelRNN/CNN [5] have been applied to modeling of acoustic features and speech samples [6, 7, 8, 9]. Although these techniques can synthesize speech with naturalness similar to that of natural speech by using a large amount of training data, there are still some problems in constructing a text-to-speech (TTS) system that synthesizes high quality speech for new languages. That is, a text analysis is required that can output language-specific prosodic information, i.e., stress (English etc.), accent (Japanese etc.), tone (Chinese etc.), and so on. Manual annotation of such prosodic information to training speech is also required to synthesize high quality speech. However, since achieving text analysis and annotation that take language-specific prosodic information into account requires considerable knowledge of or skills in spoken language processing for the target language, achieving such the text analysis and annotation will generally require significant cost. The goal of this study is synthesizing high quality speech without text analysis and manual annotation that takes language-specific prosodic information into account.

To achieve this goal, we believe that one promising approach is the use of word embeddings [10, 11] for DNN-based speech synthesis. Word embeddings is low dimensional vector representation of words, and it has been widely used in many research areas such as natural language processing (NLP), automatic speech recognition (ASR), and TTS. For TTS tasks, word vectors are used as the input for DNN instead of the language-specific prosodic information [12]. For this method, it has been reported that using word vectors instead of a prosodic information labels improves the quality of synthesized speech as compared with using phoneme information only. However, there is still a difference in quality as compared with using prosodic information labels. One reason for this is prosodic information is not considered in word vectors because word vectors are trained from text only. Although it has been shown that phoneme and prosodic information are important as the input for statistical parametric speech synthesis [13], word vectors trained from only text mainly carry not prosodic information but semantic information. Therefore, if word vectors that take prosodic information into account can be obtained, it would be expected to improve the quality of synthesized speech.

To this end, this paper proposes a novel word vector conversion technique, called prosody aware word-level encoder, to obtain word-level vectors that take prosodic information into account. The key idea of our technique is the use of a large speech corpus constructed for ASR. An ASR corpus generally contains a large amount of data in which speech and text are paired. It is expected that using paired speech and text data adequately will make it possible to obtain the relationship between words and their prosodic information extracted from speech data. In the proposed technique, we train an encoder that estimates the F0 contour for each word from the input word sequence. The outputs of the bottleneck layer in the trained encoder are used as the word-level vectors. It is expected that training the relationship between words and their prosodic information by using large speech corpus will enable the outputs of the bottleneck layer to contain prosodic information. Objective and subjective experiments in DNN-based speech synthesis showed that the proposed technique can synthesize speech with better naturalness than the conventional method.

2. DNN-based TTS using word vectors

In this section, we will briefly describe DNN-based speech synthesis using word vectors [12] (Fig. 1). In standard DNN-based speech synthesis [2], the input vector converted from phonemes, parts of speech, prosodic information (accent (Japanese etc.), stress (English etc.), and tone (Chinese etc.)) are fed to the DNN. On the other hand, DNN-based speech synthesis using word vectors does not use prosodic information explicitly. Each word in the training data is first converted to a low-dimensional word vector using a word vector conversion model. Thereafter, a vector obtained from combining the obtained word vector of

http://dx.doi.org/10.21437/Interspeech.2017-521

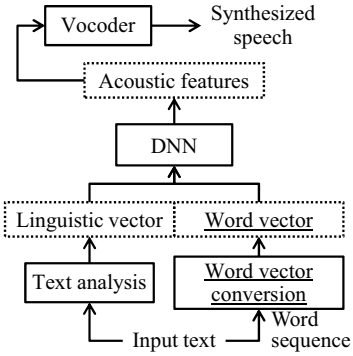

Figure 1: *An overview of DNN-based TTS with word vectors.*

each word and the vector obtained from a phoneme is used as the input of the DNN for speech synthesis. In this method, if a word vector that takes prosodic information characteristics into account can be obtained, it would be expected to generate high quality synthesized speech even though text analysis and manual annotation considering language-specific prosodic information cannot be obtained.

3. Proposed technique

3.1. An overview

Figure 2 shows an overview of the proposed technique. Although the conventional method trains the model for word embeddings using only text source, the proposed method uses a large amount of speech data, which includes paired text and speech data, such as that contained in ASR corpus to obtain a word vector that takes prosodic information into account.

A related approach to obtain word vectors that take prosodic information into account has been proposed, i.e., a postfilter that modifies word vectors trained from only text data [14]. However, for the postfilter training, manual annotation of prosodic information is still necessary. In contrast, our approach does not require such manual annotation but only a large speech corpus. A similar approach using an ASR corpus has also been proposed for word spotting tasks, i.e., acoustic word embeddings [15, 16, 17]. Acoustic word embeddings convert input spectral feature sequence to a low-dimensional word vector. However, since the obtained vector mainly contains phonetic similarity between words, our proposed approach, which focuses on prosodic similarity, is fundamentally different from acoustic word embeddings. In the DNN-based speech synthesis, to improve the quality of synthesized speech, a method which uses ASR corpus has also been proposed [18]. However, since this method has to require prosodic information label, the idea is completely different from the proposed method.

3.2. Model structure of the proposed technique

Figure 2 also shows the model structure of the proposed technique. This model estimates a fixed-length F0 vector (o_i) of each word from the input word sequence (x_i). The model consists of linear layers as the input/output layer and multiple layers of Bidirectional LSTM (BLSTM) as hidden layers. In the BLSTM layers, one layer is a bottleneck layer for word-level vector output. Here, x_i is a one-hot representation of a word, and the value of the vector is 1 for only the dimension corresponding to the input word and 0 for the other dimensions. The number of dimensions of a vector is the number of vocabulary words and o_i is the fixed-length F0 vector of the word. The

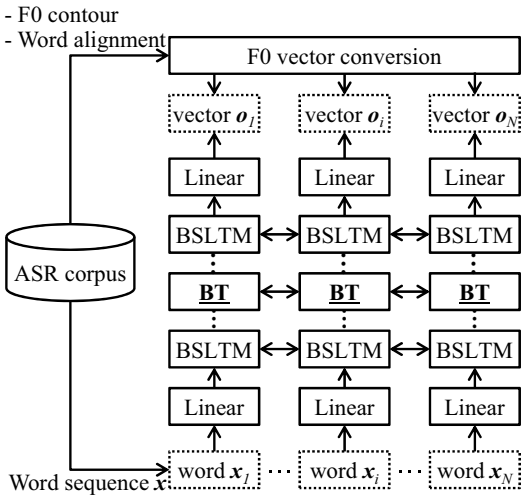

Figure 2: *Model structure of the proposed technique. This model structure estimates F0 vector for each word from input word sequence. BT indicates bottleneck layers for word-level vector output.*

details of the F0 vector are described in Sect. **3.3**. After the model training, outputs of the bottleneck layer are obtained as the word-level vector.

With the proposed model structure, it is expected that estimating a fixed-length F0 vector for each word from the word sequence will enable the hidden layers to contain the relationship between words and prosodic information. Therefore, if the F0 vectors between two words are similar, outputs of the bottleneck layer (word-level vector) between the words would be expected to be similar. Furthermore, since the F0 contour for each word strongly depends on the word sequence, an F0 vector that takes a whole word sequence into account is estimated by using BLSTMs.

3.3. Constructing fixed-length F0 vector

Since an ASR corpus contains speech data uttered by a large number of speakers, the proposed encoder will not be trained accurately by the average and variance of F0 for each speaker. Additionally, in the model structure, the DNN output o_i needs to be a fixed-length vector. However, since the frame length of each word is not fixed, the F0 for each word can not be used directly for model training. In order to avoid these problems, the F0 for each word is transformed into a fixed-length F0 vector by the following procedure.

Step 1 Calculate average and variance of F0 for each speaker in ASR corpus

Step 2 F0 of each utterance is normalized to z-score using calculated average and variance of F0 for each speaker

Step 3 Extract F0 contour of each word using word alignment information

Step 4 Resample the fixed number of F0 values from extracted contour

In the word we report in this paper, we performed Discrete Cosine Transform (DCT) to the obtained fixed-length F0, and used DCT values as DNN outputs o_i.

3.4. Fine-tuning using TTS corpus

With the proposed technique, the word-level encoder is trained from an ASR corpus. However, since the F0 contour is different

Table 1: *Details of training data.*

	No. of speakers	No. of sentences	No. of words (includes "PAUSE")	No. of words	No. of vocabulary words
ASR corpus	5,372	488,256	7,356,364	5,043,904	40,163
TTS corpus	1	4,583	52,954	41,424	6,104

Table 2: *Cosine similarities between each word (Sentence 1).*

	PropVec			TxtVec		
	date	late	door	date	late	door
gate	0.947	0.926	0.768	0.756	0.295	0.800
date		0.943	0.882		0.327	0.754
late			0.758			0.337

Table 3: *Cosine similarities between each word (Sentence 2).*

	PropVec			TxtVec		
	peace	portion	patch	peace	portion	patch
piece	0.955	0.914	0.812	0.371	0.689	0.783
peace		0.850	0.747		0.314	0.438
portion			0.938			0.705

for each speaker, the obtained word-level vector is not necessarily optimal to a speaker for a TTS corpus. To overcome this problem, the encoder trained from an ASR corpus is fine-tuned by using a TTS corpus. During the fine-tuning process, parameters of BLSTM layers are updated, and the other parameters are fixed.

4. Experiments

4.1. Experimental conditions

For the word-level encoder training, we used an ASR corpus that includes about 700 hours of speech uttered by 5,372 North American English native speakers. For each utterance, word alignment information was set by forced alignment. We also used about five hours of speech uttered by one professional speaker as the TTS corpus. Table 1 shows the details of the ASR and TTS corpus.

The word-level encoder has three hidden layers and the second hidden layer is set as a bottleneck layer. The number of units for each layer was 256 except for the bottleneck layer. We used Rectified Linear Unit (ReLU) [19] as the activation function. To investigate the performance obtained in changing the number of word vector dimensions, we trained five models whose sizes of the bottleneck layer were 16, 32, 64, 128, and 256.

To deal with unknown words, all words that occurred less than three times in the training data were regarded as unknown words ("UNK"). Additionally, unlike text data, since silent pauses are inserted in the beginning, in the middle, and at the end of the speech, we treated a silent pause as one word ("PAUSE"). As a result, a total number of 26,663 dimensional vectors, which included "UNK" and "PAUSE" words, were fed to the word-level encoder. For the encoder output, we used 1-5 dimensional DCT values of a resampled F0 of each word. Before performing DCT of F0, we resampled the F0 of each word to 32 samples. We used 99% of all data as the training data, and the remaining 1% as the validation data for early stopping. In the case of fine-tuning using the TTS corpus, we respectively used 4,400 and 100 sentences for training and validation.

In order to evaluate the effectiveness of the proposed approach, we trained baseline word vectors from text only. Text data in the ASR corpus was used for the training. As the training algorithm, we used skip-gram [11] and set the context windows size as 5. We set the number of word vector dimension as 64.

4.2. Analysis on word vectors

To confirm the effectiveness of the proposed word-level encoder, we firstly compared the proposed word-level vector (PropVec) to a word vector trained from text (TxtVec). In the comparison, we used words whose prosodic information (number of syllables, stress position) is similar but whose meanings are different, and words whose prosody information is different but whose meanings are similar. As the proposed word-level vector, we used a 64 dimensional vector trained from ASR corpus only. Since we used BLSTMs in the proposed technique, the obtained word-level vector was changed by the input word sequence. Therefore, we made the following two sentences, and compared the word vector for each word in {}.

1. I closed the {gate | date | late | door}.
2. It's a {piece | peace | portion | patch} of cake.

Tables 2 and 3 show cosine similarities between each word. It can be seen that with the proposed technique quite high cosine similarities are obtained between words having similar prosodic information such as "piece" and "peace". In contrast, in the case of words having similar meaning such as "piece" and "patch", we can see that the cosine similarities obtained are lower than those of words having similar prosodic information. On the other hand, in the case of word vectors trained from text data, the cosine similarity between words does not necessarily coincide with the similarity of prosodic information. These results lead us to consider the proposed technique is able to obtain word-level vectors with prosodic information.

4.3. Performance evaluation in DNN-based TTS

Next, we evaluated the performance of the proposed technique in DNN-based speech synthesis.

4.3.1. Experimental conditions

As the DNN for speech synthesis, we used two linear layers and two unidirectional LSTM layers [20]. The number of units per layer was 256. We used ReLU as the activation function. We respectively used sets of 4400, 100, and 83 sentences for training, validation, and evaluation. The sampling frequency of the speech was 22.05 kHz. We used STRAIGHT analysis [21] for speech feature extraction, and extracted spectral envelope, F0, and aperiodic components. The analysis frame shift was 5 ms. The spectral envelope was then converted to mel-cepstral coefficients using a recursion formula. The aperiodicity feature was also converted to average values for five frequency sub-bands. As a result, the feature vector consisted of 40 mel-cepstral coefficients including the zeroth coefficient, log F0, voiced/unvoiced flag, and five-band aperiodicity values. The total dimensionality was 47.

To compare the performance of the conventional technique, we used the following six types of input vector for DNN.

1. Phoneme (*Quinphone*)
2. **1** + Prosodic information (*Prosodic*)
3. **1** + Word vector trained from text data (*TxtVec*)
4. **1** + Proposed word vector w/o fine-tuning (*PropVec*)

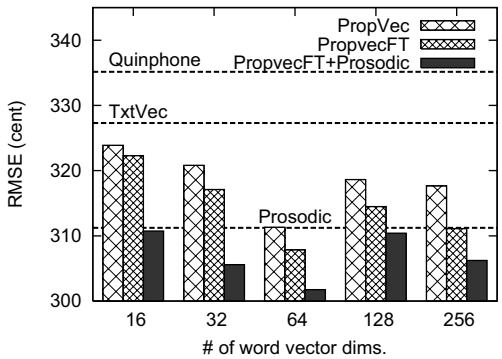

Figure 3: *RMS errors of log F0.*

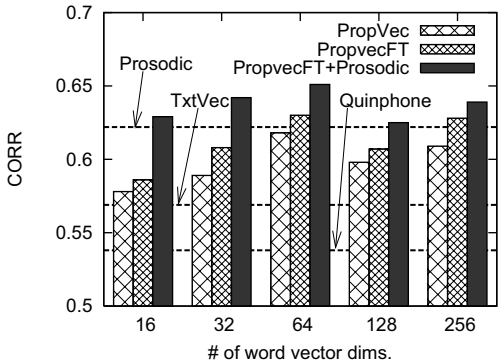

Figure 4: *Correlation coefficients of log F0.*

5. **1** + Proposed word vector w/ fine-tuning (*PropVecFT*)
6. **5** + Prosodic information (*PropVecFT* + *Prosodic*)

As the prosodic information label, we used the stress for each syllable, the ToBI endtone, and the position of syllables, word,s and phrases. In addition, since Unidirectional LSTM was used as the DNN architecture, the word vector of succeeding words cannot be considered. To avoid this problem, a concatenated word vector of current and succeeding words was fed to the DNN in methods 3 to 6.

4.3.2. Objective evaluation

Figures 3 and 4 respectively show RMS errors and correlation coefficients of log F0. Firstly, we compared the performance of the three conventional techniques, i.e., *Quinphone*, *TxtVec*, and *Prosodic*. Although RMSE of *TxtVec* was improved compared to that of *Quinphone*, it was not improved compared to that of *Prosodic*. Similar tendencies to the conventional report [12] were obtained.

Next, we compared the performance of the proposed technique to those of the conventional techniques. We can see that the proposed technique (*PropVec*) can reduce RMSE compared to *TxtVec* regardless of the number of word vector dimensions. Furthermore, RMSE of *PropVec* was comparable to that of *Prosodic* when the number of word vector dimensions was 64. In addition, fine-tuning using the TTS corpus (*PropVecFT*) can further reduce RMSE. In particular, *PropVecFT* has better performance than *Prosodic* when the number of word vector dimensions was 64. These results indicate that the proposed word vector is effective for DNN-based speech synthesis.

Finally, we evaluated the performance when using prosodic information and proposed word vector simultaneously (*PropVecFT+Prosodic*). Compared to *PropVecFT*, *PropVecFT+Prosodic* can reduce RMSE in all word vector dimensions. This demonstrates the proposed word vector is also effective when it is used with prosodic information simultaneously.

4.3.3. Subjective evaluations

We conducted an AB test to evaluate the naturalness of the speech generated by each method. All permutations of synthetic speech pairs were created and presented in both orders (AB and BA), to eliminate bias in the order of stimuli. The subjects were 16 native North American English speakers, and each was presented synthesized speech samples and then asked which of them were natural. For the evaluation, we randomly chose 20 of the 83 sentences in the evaluation set referred to in Sect. **4.3.1** above. To exclude the effects of features other than

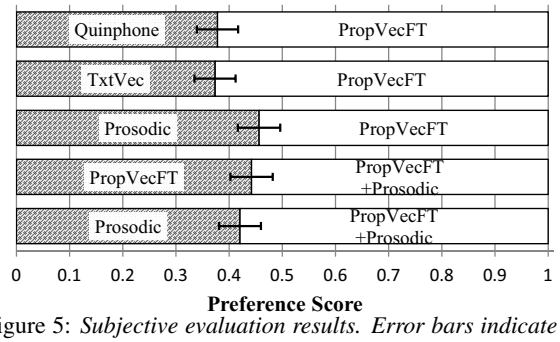

Figure 5: *Subjective evaluation results. Error bars indicate 95 % confidence interval.*

F0 (i.e., mel-cepstral coefficients, aperiodic features, phoneme durations), for these features we used these features extracted from natural speech.

Figure 5 shows preference scores obtained from each pair. The results confirm that the proposed technique (*PropVecFT*) has better performance than *Quinphone* and the conventional word vector (*TxtVec*). It also has slightly better performance than *Prosodic* even though prosodic information was not used. These results indicate that the proposed word-level encoder makes it possible to synthesize speech with higher naturalness without prosodic information. Furthermore, using the proposed word-level vector and the prosodic information simultaneously enabled further improvement to be achieved. This indicates that the proposed word vector is also effective when it is used with prosodic information.

5. Conclusions

This paper presented a novel word-level encoder to obtain word-level vectors that take prosodic information into account for DNN-based speech synthesis. In the proposed technique, we used a large speech corpus constructed for automatic speech recognition to learn relationships between words and their prosodic information. Comparisons of obtained word-level vectors confirmed that the proposed technique can obtain word vectors capturing prosodic information. Moreover, objective and subjective experiments in DNN-based speech synthesis showed that the proposed technique can synthesize speech with better naturalness than the conventional method. In future work, we will explore the performance of the proposed technique by changing the amount of training data for ASR and TTS corpus. We will also apply the proposed technique to other languages includes pitch-accent languages (Japanese etc.), tone languages (Chinese etc.), and so on.

6. References

[1] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[2] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *ICASSP 2013*, 2013, pp. 7962–7966.

[3] Y. Fan, Y. Qian, F. Xie, and F. K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *INTERSPEECH 2014*, 2014, pp. 1964–1968.

[4] 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*, 2014, pp. 2672–2680.

[5] A. V. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 1747–1756.

[6] Y. Saito, S. Takamichi, and H. Saruwatari, "Training algorithm to deceive anti-spoofing verification for DNN-based speech synthesis," in *ICASSP 2017*, 2017, pp. 4900–4904.

[7] T. Kaneko, H. Kameoka, N. Hojo, Y. Ijima, K. Hiramatsu, and K. Kashino, "Generative adversarial network-based postfilter for statistical parametric speech synthesis," in *ICASSP 2017*, 2017, pp. 4910–4914.

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

[9] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "Samplernn: An unconditional end-to-end neural audio generation model," *arXiv preprint arXiv:1612.07837*, 2016.

[10] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *INTERSPEECH 2010*, 2010, pp. 1045–1048.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS 2013*, 2013, pp. 3111–3119.

[12] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Word embedding for recurrent neural network based TTS synthesis," in *ICASSP 2015*, 2015, pp. 4879–4883.

[13] S. Yokomizo, T. Nose, and T. Kobayashi, "Evaluation of prosodic contextual factors for HMM-based speech synthesis," in *INTERSPEECH 2010*, Sept. 2010, pp. 430–433.

[14] X. Wang, S. Takaki, and J. Yamagishi, "Enhance the word vector with prosodic information for the recurrent neural network based TTS system," in *INTERSPEECH 2016*, 2016, pp. 2856–2860.

[15] A. L. Maas, S. D. Miller, T. M. Ońeil, A. Y. Ng, and P. Nguyen, "Word-level acoustic modeling with convolutional vector regression," in *ICML Workshop on Representation Learning, Edinburgh, Scotland*, 2012.

[16] S. Bengio and G. Heigold, "Word embeddings for speech recognition." in *INTERSPEECH*, 2014, pp. 1053–1057.

[17] H. Kamper, W. Wang, and K. Livescu, "Deep convolutional acoustic word embeddings using word-pair side information," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4950–4954.

[18] Z. Wu and S. King, "Improving trajectory modelling for DNN-based speech synthesis by using stacked bottleneck features and minimum generation error training," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 7, pp. 1255–1265, 2016.

[19] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML 2010)*, 2010, pp. 807–814.

[20] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *ICASSP 2015*, 2015, pp. 4470–4474.

[21] H. Kawahara, I. Masuda-Katsuse, and A. 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, pp. 187–207, 1999.