# Investigation of Using Continuous Representation of Various Linguistic Units in Neural Network Based Text-to-Speech Synthesis

PAPER

# Investigation of Using Continuous Representation of Various Linguistic Units in Neural Network based Text-to-Speech Synthesis

**Xin WANG**[†,††], **Shinji TAKAKI**[†], ***Nonmembers*, *and*** **Junichi YAMAGISHI**[†,††,†††], ***Member***

**SUMMARY** Building high-quality text-to-speech (TTS) systems without expert knowledge of the target language and/or time-consuming manual annotation of speech and text data is an important yet challenging research topic. In this kind of TTS system, it is vital to find representation of the input text that is both effective and easy to acquire. Recently, the continuous representation of raw word inputs, called "word embedding", has been successfully used in various natural language processing tasks. It has also been used as the additional or alternative linguistic input features to a neural-network-based acoustic model for TTS systems. In this paper, we further investigate the use of this embedding technique to represent phonemes, syllables and phrases for the acoustic model based on the recurrent and feed-forward neural network. Results of the experiments show that most of these continuous representations cannot significantly improve the system's performance when they are fed into the acoustic model either as additional component or as a replacement of the conventional prosodic context. However, subjective evaluation shows that the continuous representation of phrases can achieve significant improvement when it is combined with the prosodic context as input to the acoustic model based on the feed-forward neural network.

***key words:*** *Text-to-speech, Speech synthesis, Recurrent neural network, Contexts, Word embedding*

## 1. Introduction

Text-to-speech (TTS) synthesis converts text strings into speech waveforms. Due to the non-linear relationships between text and speech, TTS is normally decomposed into front-end and back-end. The front-end performs linguistic analysis and symbolic prosody prediction to obtain intermediate linguistic representations between speech and text. Based on the intermediate representations, the back-end performs acoustic feature predictions and synthesizes the speech waveform.

The front-end can be further divided into smaller yet specific sub-modules. For English TTS, they include a) grapheme-to-phoneme conversion (G2P), b) syllabification, c) part-of-speech (POS) tagging, d) syntactic parsing, e) symbolic prosody prediction, and so on [1]. These sub-modules are usually statistic models trained using databases in which correct labels are carefully and manually annotated. This results in very accurate and high-quality synthetic speech. However, it is laborious to collect such the databases. This becomes the major barrier, especially when we need to build a TTS system in a new language in which speech and linguistic resources are lacking. Even for the major languages such as English, we encounter a similar problem when we scale up the size of the database or switch to a new domain.

Recently, the continuous representation of raw word inputs, called "word embedding", has been used in various natural language processing (NLP) tasks with success [2]. Typically, this low-dimension continuous vector representation of words can be learned from raw word inputs. It has been reported that a well trained embedding vector is able to encode syntactic and semantic information [3]. Therefore, it is interesting to investigate whether such word embedding vectors can be used as the additional or alternative linguistic representations in front-end modules of TTS systems. Wang et al. utilized various types of the word embedding vectors, such as those learned by the recurrent neural network (RNN)-based language model (LM) [3] and log-linear model with negative sampling [4], for an RNN-based TTS system [5] . The objective and subjective evaluation showed that the RNN-based TTS system with the word embedding vectors performs marginally worse than that with correct POS and prosodic tags but clearly performs better than a system with neither the POS nor prosodic tags. Zhi et al. used the word embedding and triphone embedding for text-to-articulatory prediction. Their experiments also confirmed the same trend as the above work on TTS [6].

We further investigated the use of the continuous representation of input text for the neural-network-based TTS systems. Because the word embedding approaches may provide effective representations not only for word units but also at other linguistic levels, we attempted to use phoneme, syllable, and phrase embedding vectors in addition to the standard word embedding vectors. Second, these embedding vectors were examined in both RNN and deep feed-forward neural network (DNN) based acoustic models. Because RNN can capture the dependency of the sequential data across time while DNN cannot, it is interesting to know whether the power of different neural networks could influence the effectiveness of the input continuous representations. Finally, to take full advantage of the word embedding vectors, we also showed that a simple scaling approach should be used to pre-process the embedding vectors before they are fed into the acoustic model instead of using standardization method, i.e., subtracting the mean and dividing the feature by the data's standard deviation.

Manuscript received January 1, 2011.
Manuscript revised January 1, 2011.

[†]National Institute of Informatics, 2-1-2, Hitotsubashi-cho, Chiyoda-ku, Tokyo, 101-8430, Japan.

[††]SOKENDAI, 2-1-2, Hitotsubashi, Chiyoda, Tokyo, 101-8430, Japan.

[†††]The Centre for Speech Technology Research (CSTR), University of Edinburgh, Edinburgh, EH8 9LW, United Kingdom.

DOI: 10.1587/trans.E0.??.1

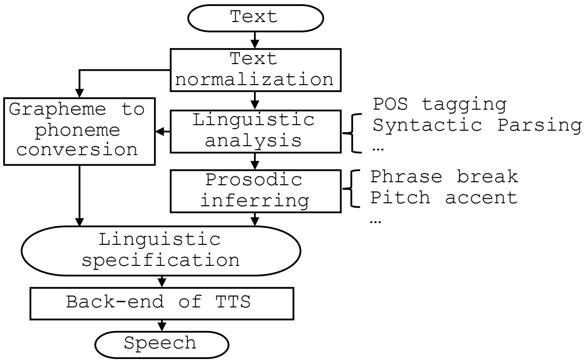


Fig. 1: Simplified diagram of conventional front-end processing flow used for typical TTS systems

In the rest of this paper, Section 2 briefly introduces the conventional front-end modules in English TTS systems. Section 3 then introduces the neural-network-based acoustic model, wherein the embedded vectors replace the conventional prosodic context. Section 4 describes the continuous representation for syllables, phonemes and phrases for TTS systems. The analysis of the effects of normalization on the word vectors is also presented. Section 5 explains the experiments and results, and Section 6 summarizes this work and describes the future work.

## 2. Conventional front-end processing flow for typical English TTS systems

To convert a text into natural speech, a TTS system must retrieve the pronunciation of words in the text and infer the prosody for the text. However, the association between pronunciation symbols and the word tokens is ambiguous in English. Also, even if the pronunciation of words is known, the prosodic information cannot be easily acquired because it is not encoded explicitly in the normal text string [7].

To tackle the above challenge, typical English TTS systems deploy the front-end and back-end architecture shown in Fig.1. The front-end of a TTS system infers the symbolic representation of both segmental and prosodic properties of speech. Then, the back-end acoustic model converts the symbolic intermediate representation into a speech waveform, typically using the unit-selection method [8] or the statistical parametric method [9].

Between the front- and back-ends, the intermediate representation encodes both segmental and suprasegmental aspects of speech. The segmental part mainly includes the phoneme sequence of every word in the input text. With a carefully produced pronunciation lexicon and well-tuned G2P algorithms, this segmental information can be inferred with high accuracy [10][11].

However, the prosody (e.g., the intonation, timing and stress of the utterance) is more difficult to predict. First, no consensus has been reached on designing the set of prosodic symbols for English that link the communicative function and acoustic realization of prosody, which is also known as the problem of "lack of reference" [12]. A widely adopted prosodic symbol set is the Tone and Break Indices (ToBI) [13] wherein pitch accent and break index represent the local pitch excursion and association between adjacent words, respectively. If the system decides to incorporate ToBI, the next problem is to predict these targets from the input text. For this purpose, the front-end of TTS infers linguistic features of the input text at first. Given the inferred POS and syntactic structure, ToBI targets can be predicted [14]. For example, as shown in Fig.1, a Part-of-speech (POS) tagger and a syntactic parser may be used at this stage to derive the sequence of POS tags and the syntactic tree of the input text. After that, the prosodic inferring module can assign the phrase break and pitch accent for the input text.

For the task of linguistic analysis and prosodic modeling in the front-end, researchers have proposed several effective methods. For example, Taylor used the hidden Markov model (HMM) to infer the phoneme sequence for each word in the input text [10]; Kupiec used HMM for POS tagging [15]; researchers also used various statistical models for syntactic parsing [16]. On the prosodic modeling part, Hirchberg utilized the decision tree to predict the pitch accent on the basis of syntactic features of the text [14].

To construct these statistical modules, expert knowledge on specific topics is required to design task-related input and output features. Also, a specific data corpus must be prepared to train each module. For example, prosodic models are usually trained on the Boston University News Radio Corpus [17] and syntactic parsers are usually trained using the Penn Treebank corpus [18]. With the expert knowledge conveyed by the designed features and data annotation, the front-end of a TTS system can exhibit good performance.

## 3. Using word embedding for neural-network-based acoustic models

### 3.1 Shortcomings of conventional front-end framework

The conventional front-end framework is not ideal for TTS, especially on prosody modeling. First, it assumes that discrete prosodic symbols must be defined. However, researchers have not yet reached a consensus on the best definition of discrete prosodic form [1][12][19]. Even if a consensus is reached, another dispute is whether symbolic prosody is necessary for a speech task. After all, the acoustic feature space is continuous. This inconsistency may result in quantization noise during prosody annotation and acoustic realization given the prosodic symbols [20][21].

Additionally, as mentioned above, prosodic models based on a supervised machine learning method require sufficient training data with consistent annotation. However, consistency of prosodic annotation across annotators may not be as high as expected [22]. Inconsistency in the training data may affect the performance of the prosodic models. The errors predicted by these models may be propagated to the following acoustic model and eventually degrade the quality of the synthetic voice.

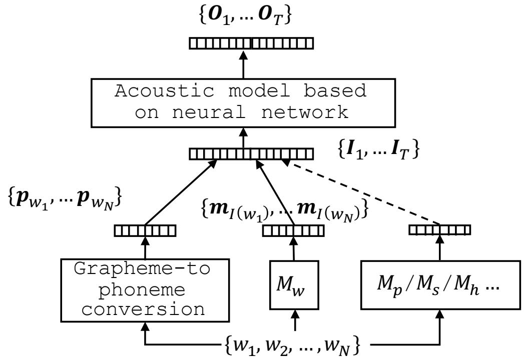

$\{\boldsymbol{O}_1, \dots \boldsymbol{O}_T\}$
Acoustic model based on neural network
$\{\boldsymbol{I}_1, \dots \boldsymbol{I}_T\}$
$\{\boldsymbol{p}_{w_1}, \dots \boldsymbol{p}_{w_N}\}$
$\{\boldsymbol{m}_{I(w_1)}, \dots \boldsymbol{m}_{I(w_N)}\}$
Grapheme-to phoneme conversion
$M_w$
$M_p / M_s / M_h \dots$
$\{w_1, w_2, \dots, w_N\}$

Fig. 2: Neural-network-based TTS with word embedding. The sequence of phonemic context $\boldsymbol{p}$ and embedded vectors $\boldsymbol{m}$ are converted into acoustic feature vectors $\boldsymbol{O}$. $\boldsymbol{M}_*$ denotes the embedded vectors for words or other linguistic units of the input text $\{w_1, \cdots, w_N\}$. Duration prediction to convert $\boldsymbol{p}$ and $\boldsymbol{m}$ sequences to $\boldsymbol{I}$ sequence is not shown here.

### 3.2 Word embedding

Although the notion of the prosodic form is beneficial, the prosodic form can be defined and modelled implicitly [20]. For example, we may directly feed the one-hot vectors of words to the acoustic model and then rely on the model to infer prosodic information during the training process. In this way, the cost of defining prosodic symbols and preparing various prosodic modules can be decreased. However, this approach is impractical because the one-hot word vector has a huge number of dimensions (e.g. 50k dimensions for Penn TreeBank corpus). More significantly, these one-hot vectors can not encode syntactic and semantic information that is useful for inferring prosody.

Recently, continuous representation of words based on the word embedding technique has became common. This method derives a continuous low-dimensional vector representation for words. It was found that, with a specific training scheme, the learned embedded vectors can encode the syntactic and semantic relationship between words. For example, Mikolov et al. utilized an RNN based language model to derive the embedded vectors [3]. The results showed that a syntactic analogy, such as “year to years is as law to laws”, can be derived through calculating the cosine distance between word vectors. Besides RNN-based LM, simple log linear models, such as continuous bags of words (CBOW) and skip-gram, have been used to derive the word embedded vectors [3].

Since word embedded vectors have fewer dimensions than one-hot word vector, they can be directly used as the input to the acoustic model. If embedded vectors indeed encode the syntactic information, the acoustic model may infer the prosodic regularity from the embedded vectors without an additional linguistic analyzer and prosodic model.

### 3.3 Acoustic modeling based on the deep neural network

The acoustic model based on the hidden Markov model (HMM) has dominated the parametric speech synthesis method for decades [9]. However, its decision-tree-based model clustering method may not be able to express complex dependency in the input linguistic representation of text [23]. Additionally, the decision-tree-based method is weak to handle continuous vectors. Thus, two kinds of neural network are adopted as acoustic models: the deep feed-forward neural network (DNN) and the deep bi-directional RNN based on long short term memory (LSTM) units [24].

While DNN is a multilayer perceptron with multiple hidden layers [25], the basic RNN further uses the hidden state of the previous time step as the input to the hidden layer at the current time step, expecting that dependency of the training data over the time span can be modelled. However, due to the gradient vanishing problem, a vanilla RNN may not be able to capture the dependency over a long time span. As a solution to this problem, LSTM has been proposed to replace the simple non-linear activation function in the hidden node of a vanilla RNN. Specifically, an LSTM unit uses three gates to control the input, output of the information flow and the state of the memory cell. This LSTM cell can also be incorporated in a bi-directional RNN, which results in bi-directional LSTM (DBLSTM) RNN [24].

For acoustic modeling, the text of an utterance is converted into a sequence of frames $\{\boldsymbol{I}_1, \boldsymbol{I}_2, \cdots, \boldsymbol{I}_t, \cdots, \boldsymbol{I}_T\}$, wherein $T$ is the total number of frames of the training data utterance. The linguistic vector $\boldsymbol{I}_t$ at time $t$ consists of the embedded vector of the word and the phonemic context at that time. Together with a sequence of acoustic feature vectors $\{\boldsymbol{O}_1, \cdots, \boldsymbol{O}_T\}$, the acoustic model can be trained. During the synthesis time, the linguistic vector of the test data can be fed into the acoustic model and acoustic features can be predicted.

## 4. Proposed methods

### 4.1 Motivation for using embedded vector of various linguistic units

Wang et. al. utilized embedded vectors of words as the input to the acoustic model instead of the ToBI symbols and POS tags [5]. However, the word is not the only linguistic unit that can be represented in the embedded space. An utterance can be hierarchically decomposed as phrases, words, syllables, or phoneme sequences. As Bian et. al. argues, the base of embedded vectors can be a sub-word unit such as a prefix, suffix, or syllable [26]. It can also be the whole sentence or document [27].

For speech application, both sub-word and supra-word vectors may be useful. The sub-word vectors can be expected to encode the segmental aspect of speech. For example, it has been shown that vowel and consonants can

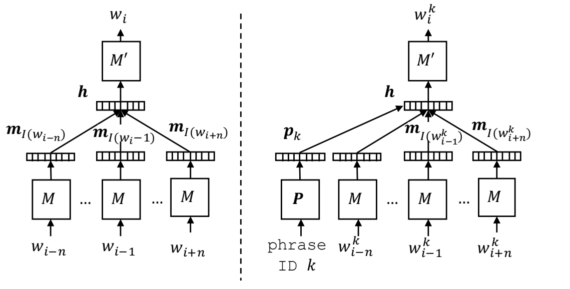


Fig. 3: CBOW (left) and Doc2Vec model (right) models

be differentiated in a two-dimensional space derived by latent semantic indexing (LSI) [28]. In this case, the phoneme symbols can be replaced by the two dimensional continuous representation and then fed into the TTS system. It is interesting to testify whether the popular embedded vectors can be beneficial to the TTS system.

Another candidate sub-word linguistic unit is the syllable. Recently, Bian et al. replaced the one-hot word vector with a 'multi-hot' vector wherein each dimension corresponding to a syllable of a certain language [26]. Their motivation is to reduce the dimension of the vector. In speech synthesis application, the use of syllable vectors mainly targets at F0 prediction. For English speech, the F0 trajectory is the acoustic realization of pitch accent while the pitch accent is usually associated with the stressed syllable [29]. When the stress information is not available for acoustic modeling (e.g., the $R_N$ system in Section 5.3), additional source of information such as the syllable vectors should be provided so that the acoustic model can implicitly infer the location of stress. Thus, it would be interesting to explore whether embedded syllable vectors can be useful.

As mentioned above, prosody of speech is also suprasegmental. A single word can be realized differently in prosody, which indicates that the word or sub-word vectors may be insufficient to encode all the prosodic information. Therefore, embedded vectors of phrases or sentences may be used. There has been a similar attempt for sentimental analysis, and Le and Mikolov [27] utilized sentence-level vectors to predict the sentiment of a sentence. Hence, it is also interesting to verify whether sentence-level vectors could benefit the acoustic model in prosodic realization.

### 4.2 Learning the embedded vectors for various linguistic units

To derive the embedded vectors for syllable and phoneme, we used the continuous bags of words (CBOW) model [4]. For CBOW, the input is a set of one-hot vectors corresponding to each word in the context $\boldsymbol{c} = \{w_{i-n}, \cdots, w_{i+n}\}$, as shown on the left side of Fig.3. The input projection layer maps the one-hot vector of context word $w$ into $\boldsymbol{m}_{I(w)}$. Because the input vector is one-hot, the projected $\boldsymbol{m}_{I(w)}$ actually corresponds to the $I(w)$-th row of the projection matrix $\boldsymbol{M}$, where $I(w)$ is the index of $w$. Then, the hidden representation $\boldsymbol{h}$ is calculated as the average of $\boldsymbol{v}$:

$$\boldsymbol{h} = \frac{1}{|\boldsymbol{c}|}\sum_{i=1}^{|\boldsymbol{c}|} \boldsymbol{m}_{I(w_i)} \tag{1}$$

where $|\boldsymbol{c}|$ is the size of the context or window size of the word context. In this case, $|\boldsymbol{c}| = 2n + 1$.

This $\boldsymbol{h}$ will be further transformed by another projection matrix $\boldsymbol{M'}$ into $\boldsymbol{u} = \boldsymbol{M'h}$. The dimension of $\boldsymbol{u}$ is the same as that of the input one-hot vector. Then, on the basis of the softmax function, the 'probability' to generate the word $w_i$ can be written as

$$p(w_i|\boldsymbol{c}; \boldsymbol{M}, \boldsymbol{M'}) = \frac{exp(\boldsymbol{m}'_{I(w_i)}\boldsymbol{h})}{\sum_j exp(\boldsymbol{m}'_j\boldsymbol{h})} \tag{2}$$

where $\boldsymbol{m}'_j$ is the $j$-th row of $\boldsymbol{M'}$ and $I(w_i)$ is the index of $w_i$. The projection matrix $\boldsymbol{\theta} = \{\boldsymbol{M}, \boldsymbol{M'}\}$ can be learned through the maximum likelihood (ML) criterion. Each row of the learned $\boldsymbol{M}$ corresponds to the embedded vector of one word. If we replace the $w$ as the token of a syllable or phoneme, the same CBOW model can be used to derive the embedded vectors for the syllable or phoneme.

For the phrase vector, we used the doc2vec model [27]. This model's structure is similar to the CBOW model except for the paragraph matrix $\boldsymbol{P}$. This structure is shown on the right side of Fig.3. At the training stage, the input to the model includes a vector $\boldsymbol{p}_k$ for the current phrase where $k$ is the ID of this phrase. Following the same training procedure as CBOW, $\boldsymbol{P}$ can be updated. Note that updating each row of $\boldsymbol{p}_k$ depends on the errors propagated backwards for all the words $w_1^k, w_N^k$ in the $k$-th phrase. At the test stage, because a test phrase is different from the training phrases most of time, the test phrase is unseen and cannot be retrieved directly from $\boldsymbol{P}$. Instead, the paragraph embedding $\boldsymbol{p}_{k'}$ for unseen phrase $k'$ must be inferred given the words in $k'$. This can be achieved using the back-propagation algorithm with all the other parameters of the model fixed.

Note that, in this paper, the quin-phone sequence is kept in $\boldsymbol{l}_t$ for all the experimental systems. The embedded vectors of all the above units are used as either supplement to or as replacement of the prosodic contexts.

### 4.3 Normalization method for the embedded vectors

For training the neural network model, the input and output features need to be normalized. However, care should be taken when the embedded vectors are normalized. As Mikolov et al. showed, the distance between embedded vectors of words $w_1$ and $w_2$ is [3]

$$cos(\boldsymbol{m}_{I(w_1)}, \boldsymbol{m}_{I(w_2)}) = \frac{\boldsymbol{m}^{\mathsf{T}}_{I(w_1)}\boldsymbol{m}_{I(w_2)}}{||\boldsymbol{m}_{I(w_1)}|| \cdot ||\boldsymbol{m}_{I(w_2)}||} \tag{3}$$

If we normalize the vector as $\hat{m}_k = \frac{m_k - \mu_k}{\sigma_k}$ (wherein $\mu_k$ and $\sigma_k$ are the $k-$th dimension of the mean and variance vector, respectively), the distance $cos(\hat{\boldsymbol{m}}_{I(w_1)}, \hat{\boldsymbol{m}}_{I(w_2)})$ between the normalized vectors will be unequal to the original distance.

To verify the above argument, we conducted a syntactic test using the word embedding derived from an RNN language model. The original embedded vectors exhibited an accuracy of 16.2%, which is identical to that reported in [3]. However, after we normalized the vectors, the accuracy dropped from 16.2% to 10.68%.

The raw embedded vectors can be used directly without normalization. However, the value of each dimension may be too small. Thus, another strategy is to scale the dimension of every embedded vector as $m_k = \frac{m_k}{\sigma_k}$. With this scale, the accuracy on the syntactic test increases from 10.68% to 12.75%. Part of the information encoded in the scaled vectors may be lost. However, it is interesting to know whether the simply scaled vectors can be more useful.

## 5. Experiments and Results

### 5.1 Preparing the embedded vectors of linguistic units

Experiments were conducted for the English TTS task. Embedded vectors of word, phoneme, syllable and phrase were involved in the experiments. For the vectors of words, we directly used the vectors trained on the basis of the RNN language model [†]. The same set of word vectors was also used by Wang et al. [5]. This vector data set covers most of the words in the speech corpus used for the following experiments; only 358 out of around 340,000 words are not covered. The vectors of out-of-vocabulary words were simply set as the global mean of all the word vectors.

The embedded vectors at the phoneme and syllable level were derived using the word2vec tool [4]. The training data were the English text in the news domain [††]. First, text was normalized [†††]. Then, Flite [30] was used to convert the text into sequences of syllables and phonemes. After that, CBOW models for syllables and phonemes were trained separately using the word2vec tool. The training process was iterated 15 times with negative sampling [4]. The window size for CBOW was set to 10 according to the experimental results in [31].

The phrase level vectors were learned using the distributed memory model of paragraph vectors (PV-DM) [27], which is shown on the right in Fig.3. The same news data corpus for syllable and phoneme vectors was used for model training. Phrases were first extracted from the corpus in accordance with the punctuation in the utterance. Then, the PV-DM model was trained with negative sampling for 15 iterations and a window size of word as 10. The dimension of both word and phrase vector was 100 for PV-DM model. Given the PV-DM model, phrase and associated word vectors for all the grammatical phrases in the speech data corpus were inferred using the back-propagation algorithm. These phrase and word vectors were concatenated as the final phrase vector for each word. We will refer this final phrase vector as phrase vector in the rest of the paper. The word vector inferred from PV-DM model is different from that from word2vec or RNN language model because PV-DM factorizes the phrase and word-level information [27].

The utilized embedded vectors are listed in Tab.1. For the pre-trained word vector, its original dimension is 80. For other vectors, the dimension of 200 is chosen based on the results of a NLP research work showing that various embedded vectors with 200 dimensions can achieve consistently good performance on name-entity recognition and sentence chunking tasks [32].

### 5.2 Experimental setup for the TTS task

Experiments in this section are introduced in a chronological order. First, Section 5.3 compares the performance of the RNN-based acoustic model with different embedded vectors as input features. The goal is to identify effective embedded vectors for further experiments. Then, Section 5.4 compares the performance of word and phrase vectors in RNN-based and DNN-based acoustic models. Meanwhile, experiments in Section 5.5 showed the influence of feature normalization methods on word and phrase vectors.

The database for acoustic model training contains 12072 English utterances by a female speaker recorded with the sampling rate of 48KHz. Five-hundred utterances were randomly chosen as the test set. Given the transcription of each utterance, its phoneme sequence was acquired using Flite [30]. Meanwhile, Mel-generalized cepstral coefficients (MGC) of order 60, a one-dimensional continuous F0 trajectory, the voiced/unvoiced condition, and band aperiodicity of order 25 were extracted for each speech frame. These parameters were extracted based on the STRAIGHT vocoder [33]. The F0 trajectory was further converted to Mel-scale according to $m = 1127 * \log(1 + f/700)$ where $f$ is frequency in Hz [34]. Although an RNN-based acoustic model was assumed to be able to model the inter-frame dependency of consecutive frames, we still used the delta and delta-delta components of the acoustic features except the voiced/unvoiced condition. Thus, the number of dimension of the acoustic feature per frame was $(60 + 25 + 1) \times 3 + 1 = 259$. During the synthesis time, the MLPG algorithm was used to derive the smooth trajectory of the predicted acoustic feature [9].

$R$ and $D$ are used to denote the RNN and DNN systems, respectively. Subscripts shown in Tab.1 are used to denote system with specific input features in addition to the quin-phone information. $R_p$ and $D_p$ are implementation of the full-fledged TTS system shown in Fig.1. All experimental systems used the quin-phone with 292 dimensions as input. The quin-phone vector includes five one-hot vectors indicating the phoneme identity in the quin-phone window and another vector indicating the vowel identity of current syllable. On the other hand, the prosodic context included binary (e.g. whether the current syllable is pitch accented) and positional information (e.g. distance to the next pitch accent)

[†] http://rnnlm.org

[††] http://www.statmt.org/wmt14/training-monolingual-news-crawl/news.2013.en.shuffled.gz

[†††] http://word2vec.googlecode.com/svn/trunk/demo-train-big-model-v1.sh

Table 1: Additional features besides quin-phones as the input to the acoustic model. Feature ID is used to identify experimental systems. For example, $R_{ps}$ denotes the RNN system with prosodic context and syllable vectors as input. For reference, the dimension of quin-phone feature is 292.

| Feature ID | Description | Dimension |
|---|---|---|
| $N$ | no prosodic context or embedded vectors | - |
| $p$ | traditional prosodic context | 90 |
| $w$ | embedded vectors of word | 80 |
| $e$ | embedded vectors of phoneme | 200 |
| $s$ | embedded vectors of syllable | 200 |
| $h$ | embedded vectors of phrase | 200 |

derived using Flite [30].

The model structure for all RNN systems contained two normal feed-forward layers with the sigmoid activation function and two bi-directional LSTM layers. Except the first feed-forward layer, the number of hidden nodes of the following layers was fixed at $(512, 256, 256)$, respectively. For systems using more than one kind of embedded vector, e.g. $R_{es}$, the size of the first hidden layer was 1024. Otherwise, it was 512. The DNN systems adopted similar structure, except the LSTM layer was replaced by a normal feed-forward layer with 512 nodes. Note that, the network structure of $R_p$ is identical to that in [5].

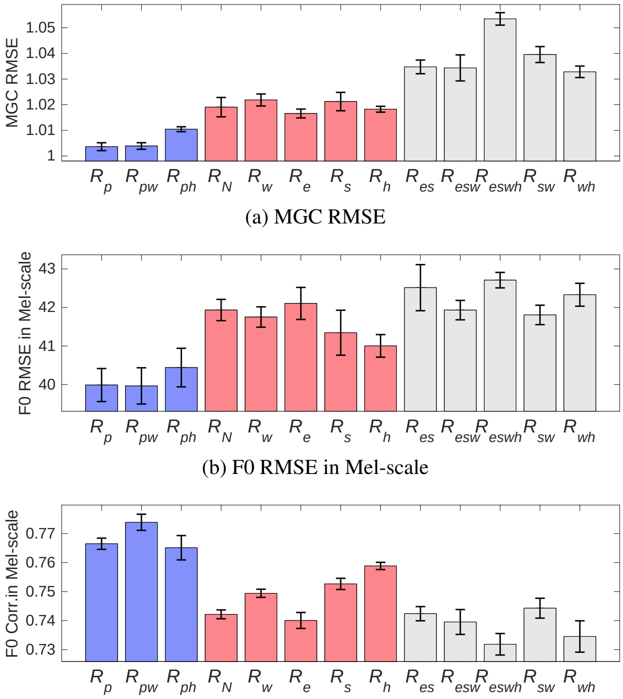
### 5.3 Performance of DBLSTM-RNN based systems with different input embedded vectors

We adopted three types of objective measures to show the performance of each system: RMSE of the predicted MGC coefficients, and RMSE and correlation coefficients of F0 trajectory on the Mel-scale. To yield meaningful results, we took the average of the objective measures on the test set using the model parameters learned from the last five training epochs for each system. The voiced/unvoiced error rate is not shown because the difference across systems was trivial.

As Fig.4(a) and 4(b) show, $R_p$ and $R_{pw}$ with the conventional prosodic context performed the best. These two systems were expected to outperform $R_N$. However, when the word vectors were used as input features, $R_w$ was only better than $R_N$ on F0 modeling. This is different from the results in [5] which indicate that $R_w$ should achieve better performance than $R_N$ both in F0 and spectral modeling part. This difference may be due to the difference of corpora. However, this can not be verified further.

Embedded vectors of other linguistic units showed different results. Specifically, $R_h$ performed the best on F0 modeling among the systems without prosodic context. This result is interesting because it indicates that the phrase vector may encode additional information related to the suprasegmental property of speech. Unfortunately, when vectors of different levels were combined, the results only degraded. Full reason for this result is unknown. But one possible reason may be limited size of the neural network.

From the results in Fig.4 we can at least infer that the phrase vector is the candidate for further interesting experiment. Besides, considering the unexpected performance of the word vectors, we decide to chose the embedded vectors of word and phrase level for further investigation.



(a) MGC RMSE

(b) F0 RMSE in Mel-scale

(c) F0 Correlation in Mel-scale

Fig. 4: Objective evaluation of the DBLSTM-RNN systems on the test set. The statistcs were calculated based on each system's performance in the last five training epoch.

### 5.4 Effectiveness of word and phrase embeddings vectors in systems based on DBLSTM-RNN and DNN

Using embedded vectors did not bring in significant improvements. Because the difference between $R_p$ and $R_N$ was less than 2 mel in F0 modeling, we wondered whether DBLSTM-RNN could directly infer the linguistic information from the quin-phone and acoustic feature sequences even without additional input formation. Thus, we prepared another five systems using the feed-forward neural network (represented by $D_*$) to replace DBLSTM-RNN. The network structure was similar, except the LSTM layers were replaced with feed-forward layers with 512 nodes. By comparison the performance of the DNN systems, we can see the effectiveness of word vectors more clearly.

The results are listed in Fig.5 and Fig.6. The first observation is that the RNN system without any prosodic context ($R_N$ in Fig.5) performs better than the DNN-based system with prosodic context ($D_p$ in Fig.6) not only on spectral part but also on F0 modelling. Readers may doubt that the improvement on the objective evaluation may be due to the smoother trajectory of the acoustic parameters predicted by the RNN. However, at least for the F0, the predicted sample trajectory in Fig.9 shows that even the DNN system without

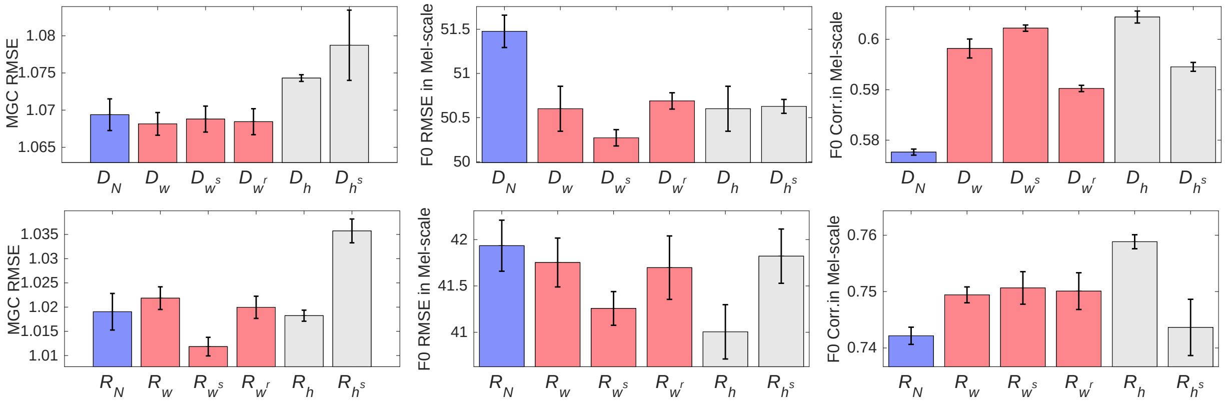


Fig. 5: Objective evaluation on systems WITHOUT using prosodic context. The first and second row correspond to DNN and RNN systems respectively. $h^s$ and $w^s$ denotes phrase and word vectors scaled by data variance while $w^r$ denotes raw vectors without pre-processing.

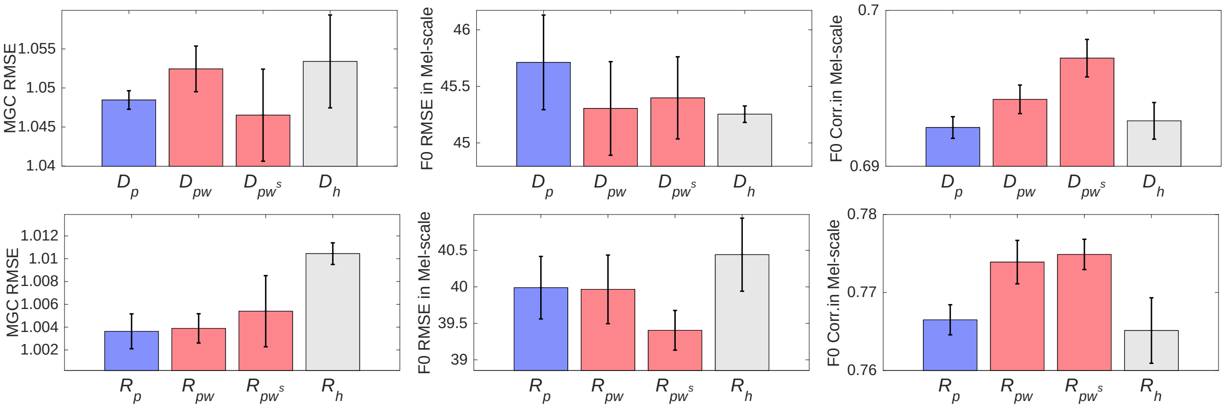


Fig. 6: Objective evaluation on systems WITH prosodic context as input.

any prosodic context can predict smooth trajectory based on the MLPG generation algorithm. Thus, the better performance of RNN may be due to the better hidden representations computed by the recurrent link in RNN.

Since DNN is weak in deriving useful hidden features, the input to DNN should encode sufficient information. The results show that the difference between DNN systems using prosodic context or not ($D_N$ VS $D_p$) was larger than the RNN case ($R_N$ VS $R_p$), which indicates the usefulness of prosodic context for DNN-based systems. The prosodic context may be less useful for the RNN, possibly due to the noise in the prosodic context of the corpus. Besides, by comparing $D_N$, $D_w$, and $D_h$ on F0 stream, we observed that the differences between $D_N$ and the other two systems were larger than their RNN counterparts ($R_N$, $R_w$ and $R_h$). This result suggests that that, similar to the prosodic context, the word and phrase vectors encode useful information for F0 modelling in DNN. But it may be less informative for the RNN systems.

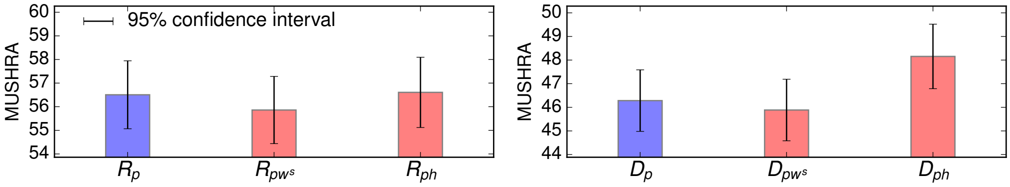
The word and phrase vector may encode similar information as that in the prosodic context. By comparing $D_N$ with $D_w$ and $D_h$, we can see that word and phrase vectors improve the performance of F0 modelling. However, when prosodic context is available, the improvement brought by the continuous vectors is less as the comparison between $D_p$, $D_{pw}$ and $D_{ph}$ shows.

In general, word and phrase vectors encode useful information for speech synthesis. However, they may provide redundant information when either prosodic context is available or useful hidden representation can be inferred by the recurrent neural network.

### 5.5 Influence of normalization on embedded vectors

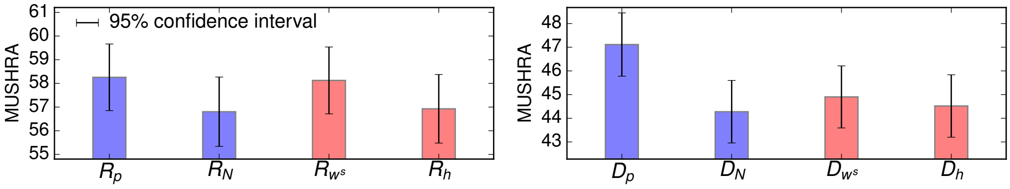
Another perspective to examine the performance of the embedded vectors is to re-examine the normalization methods. According to Section 4.3, additional systems were trained on the basis of the embedded vectors of words using different normalization methods. For $D_{w^s}$ and $R_{w^s}$ in Fig.5 and Fig.6, the embedded vectors were scaled without subtracting the mean vector; for $D_{w^r}$ and $R_{w^r}$, the embedded vectors were directly fed into the model.

The comparison between $R_w, R_{w^s}$ and $R_{w^r}$ in Fig.5 suggests that the scaling strategy results in the better perfor-



(a) (b)

Fig. 7: Performance of system when continuous vectors were used in addition to the conventional prosodic context as input to RNN-based (a) and DNN-based (b) acoustic model.



(a) (b)

Fig. 8: Performance of system when continuous vectors were used without prosodic context for RNN-based (a) and DNN-based (b) acoustic model.

mance than the other two normalization methods. This improvement can also be observed for F0 stream even prosodic context is added to the systems, which is shown by $R_{pw}$ and $R_{pw^s}$ in Fig.6.

Additional experiments were conducted for systems with phrase vectors $h$ as input. However, as the comparison between $R_h$ and $R_{h^s}$ shows, the scaling strategy does not surpass the normal normalization method. However, further investigation is required before claiming that the simple scaling strategy is not useful. First, we must find out whether the similarity between phrases can be measured in the same way as word vectors. If not, normalizing the phrase vectors may not distort the similarity in the phrase vector space. As far as we know, this is still an open question.

### 5.6 Subjective evaluation results

Four groups of MUSHRA test were conducted in order to compare the effectiveness of continuous representation of word and phrase embedding by subjective evaluation. For each group, 30 native English speakers evaluated and assigned a score from 0 (bad) to 100 (good) to 20 synthetic utterances from each experimental system.

Fig.7(a) compares the RNN-based systems with word or phrase vectors, in addition to the conventional prosodic context, as input information. The results indicate that neither continuous vector can improve the systems' performance significantly. The slightly degraded performance of $R_{pw^s}$ was also reported by Wang et al. [5]. To explain this result, they argued that prosodic context and word embeddings may contain redundant information. This is also consistent with objective evaluation shown in Fig.5 and Fig.6.

The difference between RNN-based systems is generally insignificant. This may be due to two possible reasons. First, RNN is able to infer suprasegmental information from input phoneme sequence and output acoustic features. This argument may be supported by the observation that, while the difference between $R_p$ and $R_N$ in Fig.8(a) is insignificant, a huge gap exists between $D_p$ and $D_N$ as Fig.8(b) shows. Second, the word embedding vectors may encode insufficient cues for prosody, which is indicated by the trivial difference between $D_{w^s}$ and $D_N$ in Fig.8(b). This may be reasonable because the word vectors were trained without any external prosodic knowledge. The above results are also consistent with the objective evaluation results.

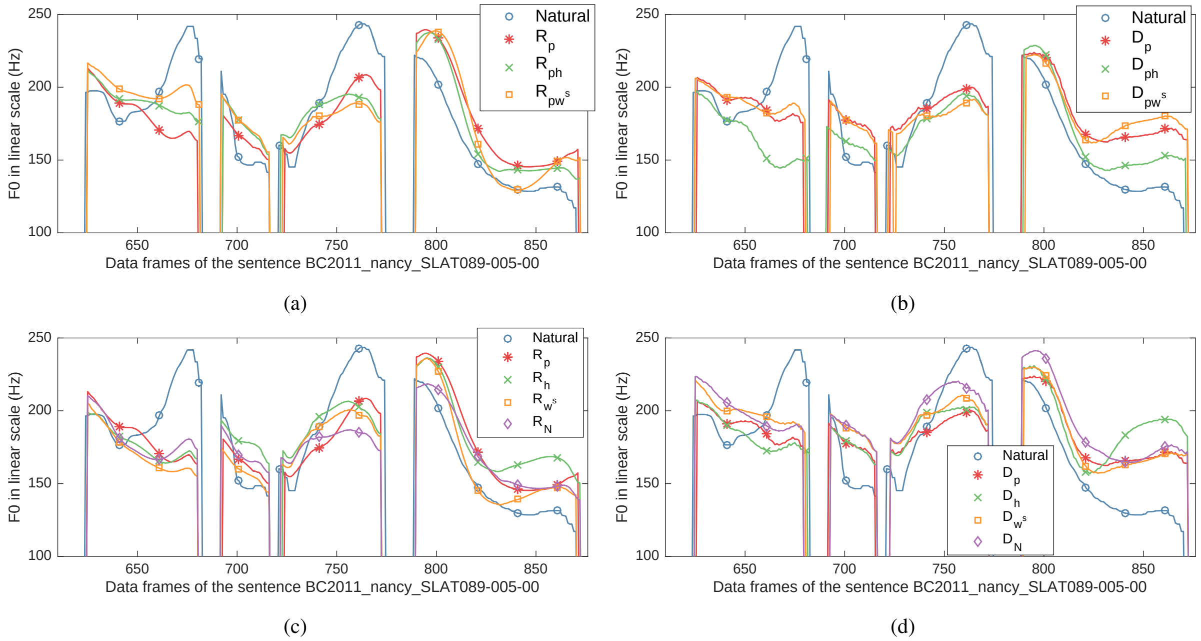
Interestingly, the phrase embeddings improved the performance of $D_{ph}$ when we compared it with $D_p$ and $D_{pw^s}$ in Fig.7(b). One reason may be that the phrase vector discriminates the acoustic units in different phrase contexts. Then, the neural network 'clusters' the units with similar phrase context and prosodic feature.

Predicted F0 trajectories by the experimental systems are shown in Fig.9. While other systems predicted similar F0 trajectory for this sample segment, $D_{ph}$ in Fig.9(b) predicted the F0 trajectory with a larger dynamic range. Although $D_{ph}$ predicted a low pitch accent around the first pick of the natural F0, this low pitch accent sounds better than the averaged F0 pattern in other systems [†]. However, due to the mismatch of the pitch accent pattern, $D_{ph}$ yields a higher RMSE in this segment. This may partially explain why $D_{ph}$ is preferred in subjective evaluation while its objective measure is not the best.

## 6. Conclusion

We investigated the embedded vectors of various linguistic units to replace the conventional linguistic context as input features to an acoustic model. The results indicated that embedded vectors of various linguistic unit only lead to in-

[†] Synthetic samples can be found here http://tonywangx.github.io



(a)

(b)

(c)

(d)

Fig. 9: Comparison of the predicted F0 trajectory by DBLSTM-RNN-based (a,c) and DNN-based (b,d) acoustic model. Systems in (a,b) utilized prosodic context as input while (c,d) not.

significant improvement of the RNN-based acoustic model. However, the results suggested that the phrase vectors may capture useful information that can be used together with the normal prosodic context.

Overall, embedded vectors for TTS must be investigated further. Typically, all embedded vectors are learned on the basis of the "meaning by collocation" assumption. It is doubtful whether this assumption is valid for speech-related tasks. In fact, many researchers recently argued that "meaning by collocation" is not ideal. Better embedded vectors may still require domain-specific knowledge [35][36]. In the future, embedded vectors may be fine-tuned with speech-related tasks so that sufficient acoustic information can be encoded in the embedded space.

Meanwhile, the syllable vector also deserves further investigation. To fully clarify the performance of $R_s$, another independent work may be required to examine, for example, whether simple one-hot vector of syllable could also be effective as one-hot representation may be equivalent to continuous representation when the number of syllables is somehow finite.

## Acknowledgments

This work was partially supported by EPSRC through Programme Grant EP/I031022/1 (NST) and EP/J002526/1 (CAF), by the Core Research for Evolutional Science and Technology (CREST) from the Japan Science and Technology Agency (JST) (uDialogue project), by MEXT KAKENHI Grant Numbers (26280066, 26540092, 15H01686, 15K12071, 16K16096) and by The Telecommunications Advancement Foundation Grant. Shinji Takaki was supported in part by NAVER Labs.

### References

[1] P. Taylor, Text-to-Speech Synthesis, Cambridge University Press, 2009.

[2] 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.

[3] T. Mikolov, W.t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations.," HLT-NAACL, pp.746–751, 2013.

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

[5] P. Wang, Y. Qian, F.K. Soong, L. He, and H. Zhao, "Word embedding for recurrent neural network based tts synthesis," ICASSP-2015, pp.4879–4883, IEEE, 2015.

[6] P. Zhu, L. Xie, and Y. Chen, "Articulatory movement prediction using deep bidirectional long short-term memory based recurrent neural networks and word/phone embeddings," INTERSPEECH-2015, 2015.

[7] M.A.K. Halliday, An Introduction to Functional Grammar, 2nd ed ed., London, UK: Edward Arnold, 1994.

[8] A.J. Hunt and A.W. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," ICASSP-1996, pp.373–376, IEEE, 1996.

[9] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura, "Speech synthesis based on hidden Markov models," Proceedings of the IEEE, vol.101, no.5, pp.1234–1252, 2013.

[10] P. Taylor, "Hidden markov models for grapheme to phoneme conversion," INTERSPEECH-2005, pp.1973–1976, 2005.

[11] A.W. Black, K. Lenzo, and V. Pagel, "Issues in building general letter to sound rules," SSW3-1998, 1998.

[12] Y. Xu, A. Lee, S. Prom-on, and F. Liu, “Explaining the penta model: a reply to arvaniti and ladd,” Phonology, vol.32, pp.505–535, 12 2015.

[13] K.E.A. Silverman, M.E. Beckman, J.F. Pitrelli, M. Ostendorf, C.W. Wightman, P. Price, J.B. Pierrehumbert, and J. Hirschberg, “TOBI: a standard for labeling English prosody,” ICSLP-1992, pp.867–870, 1992.

[14] J. Hirschberg, “Pitch accent in context predicting intonational prominence from text,” Artificial Intelligence, vol.63, no.1, pp.305–340, 1993.

[15] J. Kupiec, “Robust part-of-speech tagging using a hidden markov model,” Computer Speech & Language, vol.6, no.3, pp.225–242, 1992.

[16] M. Collins, “Head-driven statistical models for natural language parsing,” Computational linguistics, vol.29, no.4, pp.589–637, 2003.

[17] M. Ostendorf, P.J. Price, and S. Shattuck-Hufnagel, “The boston university radio news corpus,” Linguistic Data Consortium, 1995.

[18] M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini, “Building a large annotated corpus of english: The penn treebank,” Computational linguistics, vol.19, no.2, pp.313–330, 1993.

[19] D. Hirst, “The phonology and phonetics of speech prosody: between acoustics and interpretation,” Speech Prosody, 2004.

[20] E. Shriberg and A. Stolcke, “Prosody modeling for automatic speech recognition and understanding,” in Mathematical Foundations of Speech and Language Processing, pp.105–114, Springer, 2004.

[21] A. Batliner and B. Möbius, “Prosodic models, automatic speech understanding, and speech synthesis: Towards the common ground?,” in The integration of phonetic knowledge in speech technology, pp.21–44, Springer, 2005.

[22] C.W. Wightman, “ToBI or not ToBI?,” Speech Prosody, 2002.

[23] H. Zen, A. Senior, and M. Schuster, “Statistical parametric speech synthesis using deep neural networks,” ICASSP-2013, pp.7962–7966, IEEE, 2013.

[24] A. Graves, Supervised sequence labelling with recurrent neural networks, Ph.D. thesis, Technische Universität München, 2008.

[25] Y. Bengio, “Learning deep architectures for ai,” Found. Trends Mach. Learn., vol.2, no.1, pp.1–127, Jan. 2009.

[26] J. Bian, B. Gao, and T.Y. Liu, “Knowledge-powered deep learning for word embedding,” in Machine Learning and Knowledge Discovery in Databases, pp.132–148, Springer, 2014.

[27] Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” ICML-14, pp.1188–1196, 2014.

[28] O.S. Watts, Unsupervised learning for text-to-speech synthesis, Ph.D. thesis, University of Edinburgh, 2013.

[29] M.E. Beckman and J.B. Pierrehumbert, “Intonational structure in Japanese and English,” Phonology, vol.3, no.01, pp.255–309, 1986.

[30] HTS Working Group, “The English TTS System ”Flite+hts_engine”,” 2014.

[31] O. Levy, Y. Goldberg, and I. Dagan, “Improving distributional similarity with lessons learned from word embeddings,” Transactions of the Association for Computational Linguistics, vol.3, pp.211–225, 2015.

[32] J. Turian, L. Ratinov, and Y. Bengio, “Word representations: a simple and general method for semi-supervised learning,” Proceedings of the 48th annual meeting of the Association for Computational Linguistics, pp.384–394, 2010.

[33] 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.

[34] D. O’Shaughnessy, Speech communications: human and machine, Institute of Electrical and Electronics Engineers, 2000.

[35] D. Rubinstein, E. Levi, R. Schwartz, and A. Rappoport, “How well do distributional models capture different types of semantic knowledge?,” Proceedings of ACL, pp.726–730, 2015.

[36] C. Xu, Y. Bai, J. Bian, B. Gao, G. Wang, X. Liu, and T.Y. Liu, “Rc-net: A general framework for incorporating knowledge into word representations,” CIKM-14, pp.1219–1228, 2014.

**Xin Wang**

**Shinji Takaki**

**Junichi Yamagishi**