Computer Modeling in
Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.021995

ARTICLE

A New Speech Encoder Based on Dynamic Framing Approach

Renyuan Liu[1], Jian Yang[1], Xiaobing Zhou[1,*] and Xiaoguang Yue[2,3,4]

[1]School of Information Science and Engineering, Yunnan University, Kunming, 650500, China

[2]Rattanakosin International College of Creative Entrepreneurship, Rajamangala University of Technology Rattanakosin, Nakhon Pathom, 73170, Thailand

[3]Department of Computer Science and Engineering, School of Sciences, European University Cyprus, Nicosia, 1516, Cyprus

[4]CIICESI, ESTG, Politécnico do Porto, Felgueiras, 4610-156, Portugal

*Corresponding Author: Xiaobing Zhou. Email: zhouxb@ynu.edu.cn

Received: 16 February 2022 Accepted: 26 September 2022

ABSTRACT

Latent information is difficult to get from the text in speech synthesis. Studies show that features from speech can get more information to help text encoding. In the field of speech encoding, a lot of work has been conducted on two aspects. The first aspect is to encode speech frame by frame. The second aspect is to encode the whole speech to a vector. But the scale in these aspects is fixed. So, encoding speech with an adjustable scale for more latent information is worthy of investigation. But current alignment approaches only support frame-by-frame encoding and speech-to-vector encoding. It remains a challenge to propose a new alignment approach to support adjustable scale speech encoding. This paper presents the dynamic speech encoder with a new alignment approach in conjunction with frame-by-frame encoding and speech-to-vector encoding. The speech feature from our model achieves three functions. First, the speech feature can reconstruct the origin speech while the length of the speech feature is equal to the text length. Second, our model can get text embedding from speech, and the encoded speech feature is similar to the text embedding result. Finally, it can transfer the style of synthesis speech and make it more similar to the given reference speech.

KEYWORDS

Speech synthesis; dynamic framing convolution network; speech encoding

1 Introduction

Speech synthesis is the necessary technology in the field of speech information processing. Speech synthesis, also known as text-to-speech technology, can convert any text information into standard fluent speech. It involves acoustics, linguistics, digital signal processing, computer science, and other disciplines. The main task is to project the text vector into audible sound features.

The end-to-end speech synthesis approach straightly converts input text to acoustic parameters. The end-to-end speech synthesis system Tacotron2 [1,2] has developed rapidly since WaveNet [3,4] was proposed in 2016. Tacotron2 can synthesize speech more similar to real voice than the traditional HMM model, thus considerably improving synthetic speech quality.

Synthesis speech from text is a challenging problem because the meaning delivered by an utterance is underspecified by the text. For example, the same input text will get a different output speech because intonation, stress, rhythm, and style are different. That is, one sentence can be spoken in various ways.

In order to produce human-like speech, it is necessary to implicitly or explicitly send many features that can't be obtained from text to the speech synthesis model. These features are collectively named prosody. Prosody includes the intonation, stress, rhythm, and style of the speech. The traditional approach to obtaining prosodic features is manually labeling the text by a linguistics researcher. Most researchers focus on utilizing these texts and labels to design a model to predict prosody labels. But manually annotating text is heavy. And not all information can be represented by a label, such as intonation. Therefore, automatically extracting prosodic features from speech through speech encoding is worth studying.

The prosodic features encoded by speech can play other roles. First, the token features from speech can label the unlabeled speech to extend the data set. Second, in a different language, the text is not uniform, but the speech is similar, so speech features can be used to uniform text. Then data sets from other languages can be adopted to join training with uniform text embedding.

The goal of speech encoding in this article is to provide detailed prosodic features as much as possible. There are two main ways in previous research to get prosodic features from speech. The two ways are frame-by-frame encoding [5,6] and speech-to-vector encoding [7,8]. Frame-by-frame encoding encodes each speech frame to a vector, and it can get more detailed features from speech. And speech-to-vector encoding encodes whole speech to a vector, and it can get span features like sentence intonation.

These methods can provide a part of prosodic features, but they can not get more detailed prosodic features, such as the prosody of words and phrases. These methods can not get more detailed features because the existing alignment methods can not support the speech encoder to get prosodic features. Therefore, this paper mainly discusses how to solve the alignment problem and get detailed prosodic features, such as the prosody of words and phrases.

Our main contributions are as follows:

(1) A new and more generalized alignment method is proposed. The previously proposed frame-to-frame and speech-to-vector methods become exceptional cases of our proposed alignment method. This method solves the alignment problem proposed before and makes it possible to obtain detailed information by introducing adjustable scales.

(2) Dynamic Framing Convolutional Network (DFCN) is proposed to implement the new alignment approach and make the alignment process can be trained by the neural network.

(3) A Dynamic Speech Encoder (DSE) based on DFCN is proposed, which can encode speech into a predetermined length. Then DSE and speech synthesis model is adopted to build a speech autoencoder, DSE encodes the speech, and the speech synthesis model reconstructs the original speech by encoding speech feature. The speech autoencoder can better adapt the encoded speech feature to various speech synthesis tasks.

The rest of the paper is as follows: Section 2 briefly introduces the related work. Section 3 describes our approach in detail. Then Section 4 describes the experiment setting and the training strategy. Finally, Section 5 is the conclusion of this paper.

2 Related Work

Most speech synthesis systems face a problem that latent information is difficult to get from the text in speech synthesis. The traditional HMM (Hidden Markov Model) needs lots of work on processing text to solve this problem. The first work normalizes text and converts text to phoneme, reducing the difference between text and speech. The second work is to get other features in the sentence, such as word position in the phrase, the sentence is declarative or interrogative. The third work predicts the suprasegmental prosodic features, such as stress and pause.

With powerful modeling ability in deep learning, the end-to-end speech synthesis system does not need any text processing step to synthesize clear speech while the data is enough. Recently, researchers have proposed the method of applying cloud computing to the end-to-end model, which reduces the gap between end-to-end model and practical application [9]. But in most languages, data is always lacking, so text processing is necessary. The text processing in the end-to-end speech synthesis system is the same as in HMM. The text normalization and other in a sentence is easy to obtain, but suprasegmental prosodic features are hard to predict. To predict suprasegmental prosodic features, the training data set about suprasegmental prosodic features and phoneme needs to be built by linguistics researchers at first. Then the model trained on these data sets is designed which can improve the accuracy of suprasegmental prosodic features prediction.

In order to consider the relationship between suprasegmental features contained in speech and the meaning it expresses, researchers tend to add pre-trained models like BERT [10] to get semantic information in the text [11,12]. And the relationship within the sentence is further constructed through the graph, such as the dependency tree [13]. However, complex text features require sufficient speech features to support, which need to be obtained manually. So researchers have considered getting the suprasegmental prosodic features and phonemes from speech. This method can build the data set automatically and reduce the work of linguistics researchers [14]. Speech encoding can also be useful in many tasks, such as processing speech modal in multimodal tasks [15].

There are two ways to encode speech: frame-by-frame encoding and speech-to-vector encoding.

The frame-by-frame encoding approach encodes each speech frame to a vector. Various methods are used to extract the features of speech frames, such as Convolutional Layer, LSTM, Transformer [16], or build a graph in speech frames to get more detailed features [17]. Then the encoded speech frames can be used in many tasks, such as speech recognition [18] and direct speech translation [19].

In frame-by-frame method, researchers find many ways to detect the relationship between the features of speech frame and text features, such as using attention mechanisms to extract the relationship in Tacotron2 and transformer. CTCLoss [20,21] extends the length of the text, makes the text correspond to the voice frame one by one, and finally restores the extended text to the original text by merging the same items. GTCLoss [22] can directly establish the relationship between speech and suprasegmental features, and community detection [23,24] can directly establish the relationship between speech and text.

To utilize the speech feature from the frame-by-frame method, researchers try to combine speech synthesis and speech recognition models [25] to extend the data set and correct the inconsistent between text and speech. In the combined model, the speech recognition result is adopted to train the speech synthesis model, and the speech synthesis result is adopted to train the speech recognition model, or intermediate variables in the speech recognition model are adopted to guide the speech synthesis model. Due to the alignment approach in the frame-by-frame encoding, such as CTCLoss and GTCLoss, the speech features in the frame-by-frame encoding approaches must be quantified to merge the equal features to correspond to the text feature. So the frame vectors are independent, so it cannot get the span features, such as prosody.

The speech-to-vector approaches encode whole speech to a vector. This vector includes the features in the whole speech, such as speaker and acoustics features. Given the encoded speech vector, some methods like autoencoder or contrast learning are adopted to obtain these features [26,27]. GST (Global Style Token) [28,29] is one of the representatives. In the GST model, encoded speech vectors are added to each encoded text result to help the speech synthesis model recognize and synthesize prosody. Because this method does not consider the alignment, each text token corresponds to the whole speech. Speech to vector approaches can get sentence prosody, but they cannot get details, such as phoneme prosody, word prosody, and phrase prosody.

3 Model Architecture

We extend the Tacotron2 architecture by adding a Dynamic Speech Encoder (DSE) module that takes a speech signal as input and computes a length-L_T feature. L_T is the length of input text, and it can be the number of characters/phonemes, words, phrases even sentences. Our goal is to use the output length-L_T feature in Tacotron2 as prosody, attention context, or token representation to synthesize speech.

Our proposed Dynamic Speech Encoder (DSE) module, illustrated in Fig. 1, consists of a Duration Predictor, Dynamic Frame Convolutional Network (DFCN), and Segment Encoder. Duration Predictor predicts the window length and window position from text and speech. Then DFCN segments the speech based on predicted window length and window position. Finally, Segment Encoder encodes each segmented speech to a vector.

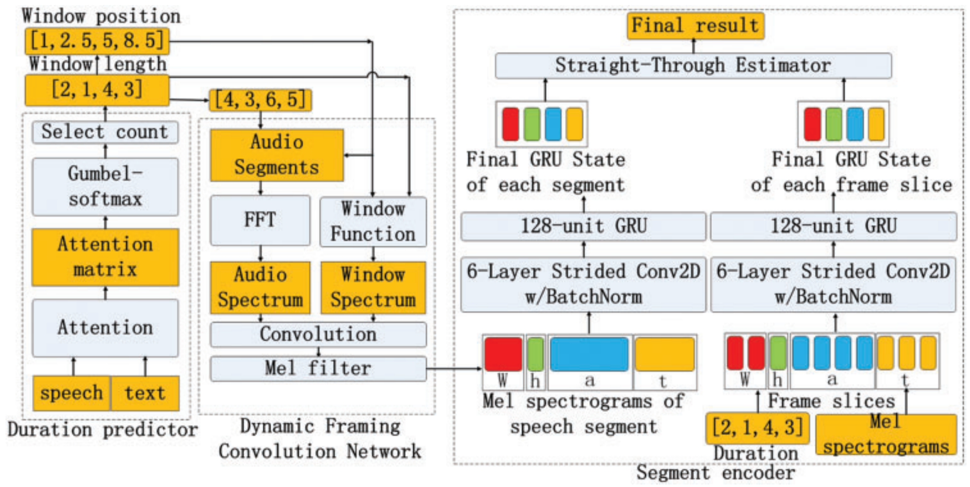


Figure 1: Model architecture of DSE

3.1 Duration Predictor

The Duration Predictor is shown in Fig. 1. The aim of Duration is to predict the window length of each window. The window length is the length of each window, and the unit of window length is frame. For example, the window number is set to the equal to the letter number of the text in Fig. 1. Assume the speech spectrum is $S = \{s_1, s_2, s_3, \ldots, s_n\}$, n is frame number, so the [2, 1, 4, 3] means the first token corresponds to first 2 frames s_1, s_2 in speech, and the second token corresponds to the next 1 frame s_3 in speech, and so on. The window position is the position of the center of each window in the speech, and the unit of window position is also frame. So the [1, 2.5, 5, 8.5] means the center of the first window is located on 1 frame in the speech, and the center of the second is located on 2.5 frame in the speech, and so on.

As can be seen from the definition of window length and window position, the window position can be calculated from the window length. There are window lengths $L = \{l_1, l_2, l_3, \ldots, l_n\}$. In this paper, windows are adjacent. So, the window position $P = \{p_1, p_2, p_3, \ldots, p_n\}$ is computed as follows:

$$p_i = \sum_{j=1}^{i-1} l_j + \frac{l_i}{2} \tag{1}$$

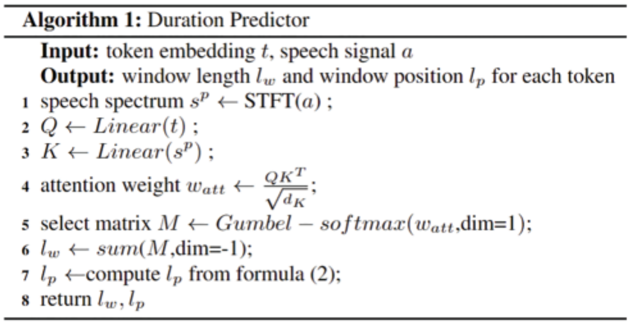
The algorithm of duration predictor is shown in Algorithm 1. First, the token representation and Mel spectrograms coefficient is sent to the Duration Predictor. The token representation can be character embedding, phoneme embedding, word vector, sentence representation, etc. The length of token representation is the number of windows. If a token is needed to correspond to three windows, a simple method is the token representation repeated three times.

Second, the attention matrix between token representation and Mel spectrograms coefficient is calculated by the attention mechanism. In our experiments, the local sensitive attention in Tacotron2 is adopted to get the attention matrix.

Third, the Gumbel-softmax [30] is adopted to transform the attention matrix to a binary matrix named Select Matrix. Gumbel-softmax is as follows:

$$\text{Gumbel-softmax}\,(E_s) = \text{softmax}\left(\frac{(E_s + G')}{\tau}\right) \tag{2}$$

where G' is Gumbel noises [31], and $\tau \in (0, 1]$ is a temperature parameter. As τ approaching zero, the sample from the Gumbel-Softmax distribution becomes cold and resembles the one-hot sample.



```
Algorithm 1: Duration Predictor
    Input: token embedding $t$, speech signal $a$
    Output: window length $l_w$ and window position $l_p$ for each token
speech spectrum $s^p \leftarrow$ STFT($a$) ;
$Q \leftarrow Linear(t)$ ;
$K \leftarrow Linear(s^p)$ ;
attention weight $w_{att} \leftarrow \frac{QK^T}{\sqrt{d_K}}$;
select matrix $M \leftarrow Gumbel - softmax(w_{att}$,dim=1);
$l_w \leftarrow sum(M$,dim=-1);
$l_p \leftarrow$compute $l_p$ from formula (2);
return $l_w, l_p$
```

In the Select Matrix, the row is the frame of Mel spectrograms, and the column is token representation. If the value of the third row and the second column is '1' in the Select Matrix, it means the third frame of Mel spectrograms selects the second token. And Gumbel-Softmax means each Mel spectrograms frame only selects one token in the Select Matrix. We count the selected number of the token as the window length output of the Duration Predictor. For example, if the first row has three '1' in the Select Matrix, the first token corresponds to the three frames speech segment. This method means the basic unit of predicted result is the frame.

3.2 Dynamic Framing Convolution Network

DFCN transforms the windowing operation in the time domain into a convolution operation in the frequency domain. The length and position of the window can be trained by deep learning so that the windowed speech signal can correspond to the text one by one. The architecture of DFCN is shown in Fig. 1.

First, the basic unit of predicted window length and window position is transformed to sample point as follows:

$$l_i = \begin{cases} (l_i - 1) * F_s + F_l & \text{if } l_i > 0 \\ 0 & \text{if } l_i = 0 \end{cases} \tag{3}$$

$$p_i = \begin{cases} (p_i - 1) * F_s + F_l & \text{if } p_i > 0 \\ 0 & \text{if } p_i = 0 \end{cases} \tag{4}$$

where F_s is frame shift, F_l is frame length.

Second, the FFT is used to get the audio spectrum, and the Window Function is used to estimate the window spectrum. Third, the audio spectrum and window spectrum are convoluted to get the spectrum of the windowed speech segment. Finally, the convolution result is sent to the Mel-filter to get the Mel spectrograms to simplify the computation. So, the DFCN is the convolution of the audio spectrum and window spectrum. The key of DFCN is how to estimate window spectrum and extract audio spectrum for computing.

3.2.1 Window Spectrum Estimate

The kernel of DFCN is the window function. The window function in the frequency domain is used to estimate the window spectrum. Through the window function, we can get the approximation of the window spectrum. The window function of i_{th} window in the frequency domain is as follows:

$$G_i(\omega, f) = \frac{\left(\sin\left(2\pi \frac{f_s}{2}\omega\right) - \sin\left(2\pi \frac{f_s - l_i}{2}\omega\right)\right) e^{j\omega\left(-\frac{f_s}{2} + p_i\right)}}{2\pi\omega} \tag{5}$$

where f_s is the sampling rate of ω, l_i is the length of the i_{th} window, p_i is the window position of the i_{th} window.

In these functions, f_s decides how to project the frequency parameter to the time parameter. Assuming the audio length (number of points) is T, l_{time} and l_{freq} is window length in time domain and frequency domain, ts_{time} and ts_{freq} is window shift in time domain and freq domain, so the relationship is as follows:

$$\begin{aligned} l_{time} &= \frac{l_{freq} T}{f_s} \\ ts_{time} &= \frac{\left(-\frac{f_s}{2} ts_{freq}\right) T}{f_s} \end{aligned} \tag{6}$$

3.2.2 Audio Spectrum Extract

The computation is too large when directly convolution the whole speech spectrum. For example, if the speech has 2,000,000 sample points, the spectrum length after FFT is 2,000,000. Also, the data length is 2,000,000. If the text length is 50, this sentence will generate a matrix size [50, 2,000,000]. The data is too large to use GPU to calculate it.

Based on the HMM alignment approach, the audio segment approach is adopted to reduce the computation in DFCN. First, the speech signal is segmented according to the predicted duration, and two more frames are left at both ends of each segment, i.e., the window position is not changing, and window length adds to 2. Second, FFT is used to compute the spectrum of each audio segment. Finally, the window spectrum is used to convolute the audio spectrum of each segment one by one. In Fig. 1, assume the speech spectrum is $S = \{s_1, s_2, s_3, \ldots, s_n\}$, where n is the frame number. We pad the speech signal and get $S = \{s_p, s_1, s_2, s_3, \ldots, s_n, s_p\}$ first. Then the duration predictor outputs the window length [2, 1, 4, 3] and window position [1, 2.5, 5, 8.5], so the added window length is [4, 3, 6, 5], and four speech segments can be obtained. The first segment is $\{s_p, s_1, s_2, s_3\}$, and the second segment is $\{s_2, s_3, s_4\}$, and so on. These frames correspond to four slices in speech signal, and the slices length can be computed by formulas (3), (4). The spectrum of the whole segment is sent to the convolutional layer as the audio spectrum. Finally, the audio spectrum and window spectrum are convoluted one by one.

This training strategy allows the window parameter to change within two frames in one iteration. That is to say, the result of the time length predictor can only get the change of about two frames in each gradient iteration. When the learning rate is 0.001, the change of one frame is sufficient.

This approach also can solve another problem in DFCN. The window in the time domain can let the data outside the window to zero, but the window function in the frequency domain is just an approximation of the window in the time domain. The DFCN cannot let the data outside the window to zero. It enlarges the data inside the window and reduces it outside the window. So the data outside the window becomes the noise of each frame. Therefore, pre-segment speech can discard a lot of data outside the window and reduce the difference between the window in the time domain and the window function in the frequency domain.

Finally, the algorithm of DFCN is shown in Algorithm 2.

```
Algorithm 2: Dynamic Framing Convolution Network
    Input: speech signal $a$, window length $l_i$, window position $p_i$
    Output: windowed speech spectrum $s_{window}$
1   $a_p \leftarrow$ pad the speech signal for extend window ;
2   extended window length $l_i^{add} \leftarrow l_i + 2$;
3   $l_i^{add} \leftarrow$transform the unit of window length $l_i^{add}$ from frame to sample point by formula (3) ;
4   $p_i \leftarrow$transform the unit of window position $p_i$ from frame to sample point by formula (4) ;
5   speech slice $s_{slice} \leftarrow$ a[$p_i - \frac{l_i^{add}}{2}$:$p_i + \frac{l_i^{add}}{2}$];
6   spectrum of speech slice $s_{slice}^p \leftarrow$ FFT($s_{slice}$) ;
7   window spectrum $w^p \leftarrow$ compute window spectrum by formula (5),(6) ;
8   $s_{window} \leftarrow$ convolute($s_{slice}^p, w^p$) ;
9   return $s_{window}$
```

3.3 Segment Encoder

DFCN can get several segments for each token. If each token corresponds to a single window, the length of convolution resulting from DFCN equals text length. In this case, three linear layers are adopted to encode the result of DFCN.

If each token corresponds to multi-windows. The structure of the encoder is shown in Fig. 1. It consists of 6 2D convolution layers and one GRU layer. Convolution filter length is [128, 128, 256, 256, 512, 512]. The GRU only outputs the final state as the encoded output for each segment.

There is an error between the speech spectrum obtained by DFCN and STFT. Straight-Through Estimator in VQ-VAE [32] is adopted to reduce this error. In Fig. 1, the text is 'what' and correspond 4 windows, the input named 'Mel spectrograms of speech segment' is the output of DFCN, each token corresponds to one spectrum vector, and each spectrum vector includes the feature in several frames. The input named frame slice uses the window length to slice the reference speech spectrum, each token directly corresponds to several frame in reference spectrum. Reference speech spectrum is obtained by STFT result of speech.

The algorithm of segment encoder is shown in Algorithm 3. First, the STFT output is segmented by predicted duration to get the frame slice. Then the frame slice is sent into another segment encoder. For example, the predicted duration of the first token is 2 frames in total. So the segments of the first token in DFCN output correspond to the 2 frames in STFT output. Finally, we use the encoding result of STFT to calculate the next step, but the gradient is directly propagated to the result of DFCN. Finally, the encoding results of STFT and DFCN are made as close as possible by Mean Square Error Loss (MSELoss).

```
Algorithm 3: Segment Encoder
    Input: speech signal $a$, windowed speech spectrum $s_{window}$
    Output: synthesize speech $s_{out}$, and loss for straight forward method $loss_s$
1   reference speech spectrum $s_{ref}^p \leftarrow$ STFT($a$);
2   reference speech feature $f_{ref} \leftarrow$ GRU(Convolutional Layer($s_{ref}^p$));
3   speech feature $f_{speech} \leftarrow$ GRU(Convolutional Layer(($s_{window}$));
4   $s_{out} \leftarrow f_{speech} + (f_{ref} - f_{speech})$.detach();
5   $loss_s \leftarrow$ MSELoss($f_{speech}, f_{ref}$.detach());
6   return $s_{out}$,$loss_s$
```

3.4 Contrast Learning

Based on the DSE model, contrastive learning [33] is adopted to further strengthen the connection between speech and text. Specifically, a discriminator D is adopted to distinguish positive and negative sample pairs. The discriminator consists of three linear layers with activation function ReLU. The positive sample pairs are the encoded speech features that correspond to the same text. The negative sample pairs are the encoded speech features that correspond to the different text.

However, there is no guarantee that sufficient positive samples can be obtained in batch, so a database is necessary to save the previous encoding features. Also, the corresponding features will not be sent to the discriminator when the positive sample cannot be obtained.

The loss function of the discriminator D is as follows:

$$L_D = \log \frac{e^{D(x_i,y_i)}}{\sum_{i\neq j} e^{D(x_i,y_j)}} \tag{7}$$

where the numerator represents positive samples and the denominator represents all negative samples. The loss of the discriminator is adopted as an auxiliary loss that is added to the subsequent tasks. The auxiliary contrast loss needs to be weighted because the same word or phrase may have different detail features in different sentences. Finally, the correspondence between text and speech can be further ensured by contrast learning.

4 Experiment

In this section, we measure the ability of DSE by three experiments. These experiments are Speech Compress, Text Embedding, and Style Transfer. The hyper-parameter is as follows. First, the optimizer is Adam with a 1e-3 learning rate and 1e-6 weight decay, Grad clip thresh is 0.5, the batch size is 8. In audio parameters, the sampling rate is 22,050, the frame length is 1024. The Mel channel is 512, min Mel frequency is 0 and max Mel frequency is 8000. For the networks in our model, the output dimension is 512, such as embedding, encoder, and attention.

4.1 Datasets

The proposed model is trained and tested on the public datasets LJSpeech and LibriTTS.

LJSpeech [34] is a public domain speech dataset consisting of 13,100 short audio clips of a single speaker reading passages from 7 non-fiction books. Clips vary in length from 1 to 10 s and have a total length of approximately 24 h. A transcription is provided for each clip. In this paper, we use the LJSpeech in Speech Compress and text embedding experiments.

LibriTTS [35] is a corpus of approximately 1000 h of 16 kHz read English speech, prepared by Vassil Panayotov with the assistance of Daniel Povey. The data is derived from reading audiobooks from the LibriVox project and has been carefully segmented and aligned. This is a multi-speaker dataset and usually is used in speaker adaption speech synthesis systems. In this paper, we use this dataset to enhance the result of the Style Transfer experiment.

4.2 Speech Compress

Our model in this experiment is a auto-encoder [36]. The aim of this experiment is to demonstrate that the speech feature can reconstruct the origin speech while the speech feature-length is equal to the text length.

The architecture of Speech Compress experiment is shown in Fig. 2. First, the Length Regular block in the Fastspeech [37] is adopted to extend the encoder outputs to the same length of Mel spectrograms by predicted duration. Second, the Length Regular result is input into the Tacotron2 encoder to get encoder outputs. Finally, the encoder outputs are sent into the decoder to replace the attention context.

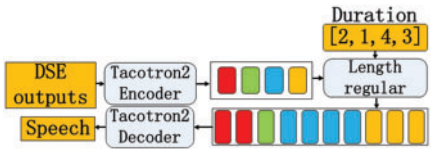


Figure 2: Speech compress architecture

In this experiment, a problem affects the model result named exposure bias. The exposure bias is the difference between the training and synthesis processes. The training process uses the real Mel spectrograms to predict the next frame, and the synthesis process uses the predicted Mel spectrograms to predict the next frame. This problem will lead to even the training loss being small, but the synthesis speech is bad. So both the training and synthesis processes are used to train the model to avoid exposure bias.

Fig. 3 shows the results of several synthetic speeches. In Fig. 3, the first line is each token corresponding to 5 windows, and the second line is each token corresponding to 3 windows. The windows with zero window length in the first line are more than the second line. It means if the window number per token is big enough, the length of a part of windows will equal frame length, and the length of the other part of the window will become zero. This alignment approach is similar to the alignment approach in the frame-by-frame encoding.

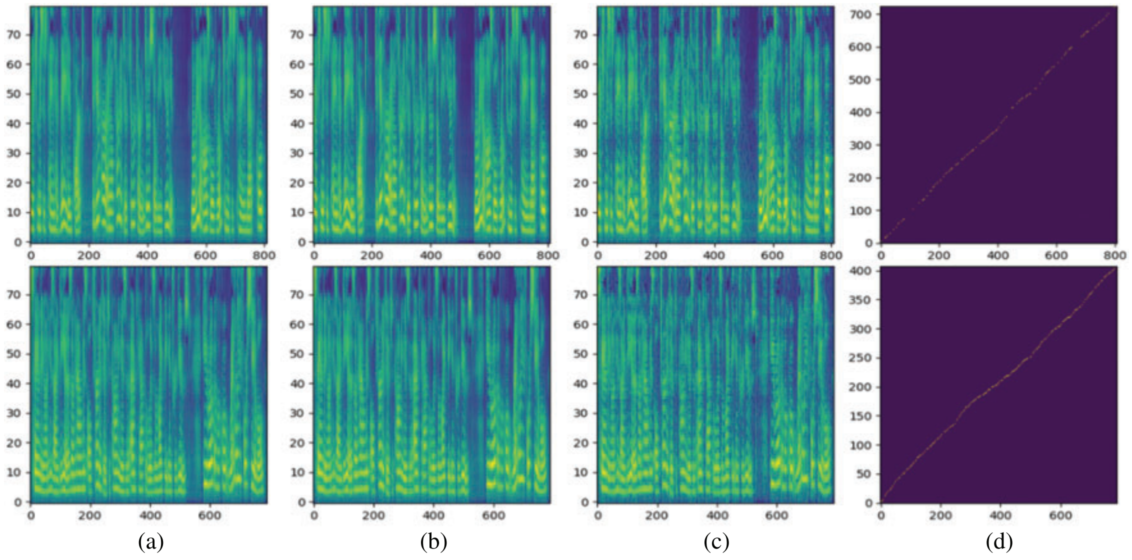


Figure 3: Results of training process (a), results of synthesis process (b), real spectrum (c) and window length (d)

Finally, the average loss of Mel spectrograms obtained by the training process in evaluating the data set is about 0.48, and the synthesis loss in evaluating data set is about 0.5. This loss is close to the final loss in Tacotron2 of 0.45. Although the synthesis speech cannot be played on paper, the synthesis process's loss is also close to the final loss in Tacotron2. Also, in Fig. 3, the synthetic speech cepstrum is not much different from the real cepstrum. So a conclusion can draw that the reconstruction speech is close to the real voice. The proposed Dynamic Speech Encoder achieves our aim in the Speech Compress experiment.

4.3 Text Embedding Experiment

This experiment tries to quantify the Dynamic Speech Encoder output, the speech feature which corresponds to the same token will be quantified to a common vector. Then the quantified features are used to reconstruct the original speech.

As shown in Fig. 4, the quantified features are used to replace the text embedding vector in Tacotron2 to synthesize speech. This experiment is jointly trained with the Speech Compress experiment to prevent the model from falling into the local optimum.

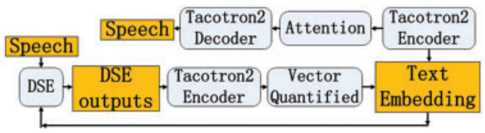


Figure 4: Text embedding architecture

The Duration Predictor also needs a text embedding vector, so we need to use the quantified features to replace the text embedding vector in the Duration Predictor. First, we train the model with uniform segmentation. Second, we use the quantified features to replace the text embedding vector in Duration Predictor after 1000 iterations.

Next, we compute the mean square error between text embedding vector and quantified speech feature. If the quantified vector is used for speech synthesis, the model will require the quantified vector to include more speech features. If the quantified vector is not used for speech synthesis, the mean square error between the quantified vector and text embedding vector is 0.005. The mean square error between the quantified vector and text embedding vector increases to 0.03. Both 0.005 and 0.03 are small enough to support the result that the quantified vector is closed to the text embedding vector.

Then we need to confirm that the quantified Dynamic Speech Encoder output can synthesize speech. The synthesis speech cepstrum is shown in Fig. 5.

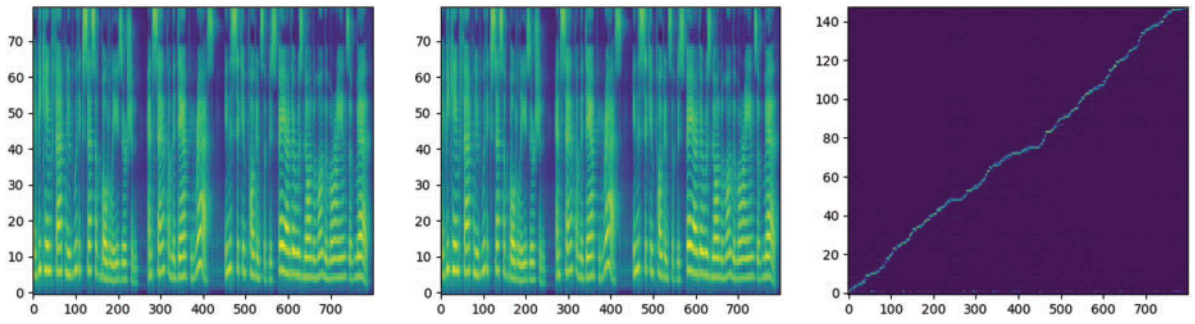


Figure 5: Synthesis output from quantified Dynamic Speech Encoder output

The average loss converges to 0.45, close to the Tacotron2 final loss. The synthetic speech cepstrum is also slightly worse than the speech synthesized by Tacotron2, but it can also get clear speech. So the synthesis speech from quantified Dynamic Speech Encoder output is also close to the real voice. Finally, Dynamic Speech Encoder achieves our goal. It can extract text embedding vectors from speech, and quantified Dynamic Speech Encoder output can replace the text embedding vector to synthesize speech.

4.4 Style Transfer Experiment

This experiment is an improvement of the Style Transfer experiment in GST. As shown in Fig. 6, the Dynamic Speech Encoder outputs are sent to multi-head attention to get Local Style Token. Then the Local Style Token is added to Tacotron2 encoder outputs as a prosody feature. Global Style Token can be regarded as a particular Local Style Token when the speech segments correspond to a single vector. This experiment aims to demonstrate that the prosody of synthesis speech is similar to reference speech when the LSF obtained from the referenced speech is added to encoder outputs.

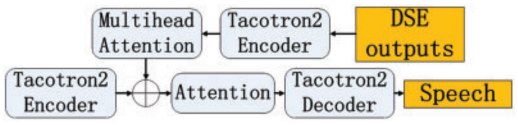
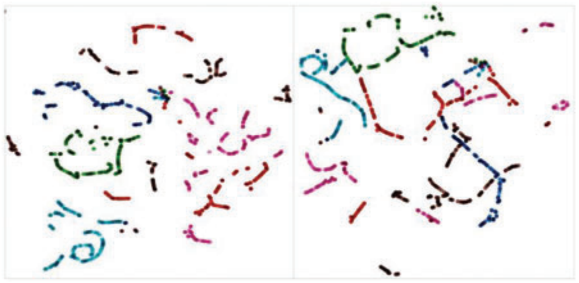


Figure 6: Style transfer architecture

In Fig. 7, the Local Style Token of multi-speaker speech is visualized by t-SNE. Different colors represent different speakers.

The Local Style Token becomes several line segments. Fig. 7a shows that some line segments will get together, such as brown and blue points. The other colors are distributed separately, such as red, pink, and brown points. The reason for this result is the t-SNE parameters. Fig. 7b is used t-SNE with different parameters. In Fig. 7, only pink points are distributed separately. In both Figs. 7a and 7b, we also can see these colors have several points in one place. We guess it is the silence part and the point from silence to speech.

Fig. 8 (left) shows three spectrograms (baseline model and two prosody references) for the same text. The block in spectrograms is synthesize part correspond one by one. Note that the spectrogram from the model with prosody reference is more similar to the reference speech than that generated by the baseline model. The most obvious point is the time of the spectrograms from the baseline model is shorter than others because the reference speech is long. And the tone and pause characteristics from the prosody reference model are more similar to the reference signal.



(a) (b)

Figure 7: Local style token of multi-speaker

Fig. 8 (right) shows the pitch tracks for the same triplet of text. In the range of 0 to 400 Hz, the pitch from the baseline model is gentle. Then the middle synthetic speech has a little change in tone but changes the pause characteristics a lot. Finally, the bottom speech changes tone a lot. Fig. 8 can see different style tokens will change the speech prosody.

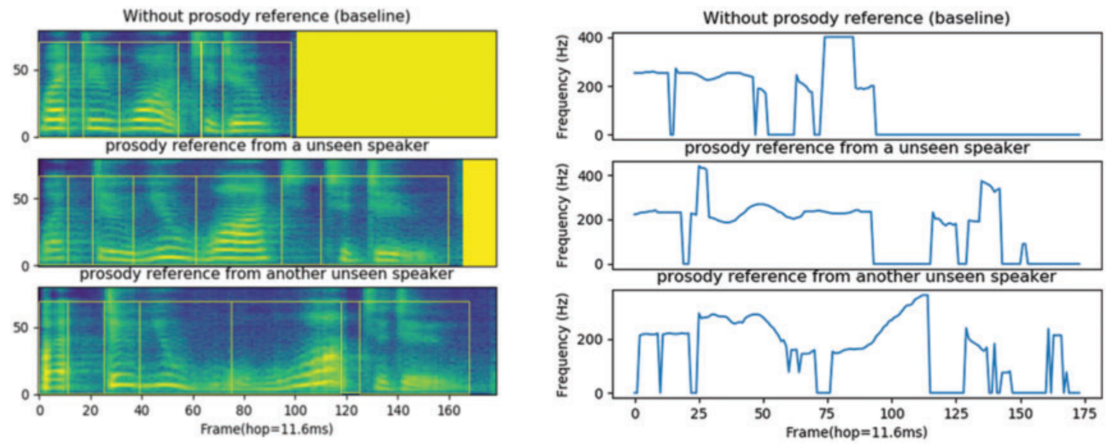


Figure 8: Mel spectrograms (Left) and pitch (Right) for the input text "what do you want to do?" synthesized text from a model without reference (Top). Reference text from an unseen speaker (Middle). Reference text from an unseen speaker (Bottom)

Fig. 9 shows the first 400 frames of alignments for the synthesized speech from the input text "And it is worth mention in passing that as an example of fine typography". These speeches are synthesized by the GST model and our DSE model with three different speakers' different emotional sounds (a), (b), and (c) as reference speech. It can be seen from (a) that the speech synthesized by our DSE has no great distorted parts (parts inside the rectangle), and the coherence is better than that synthesized by the GST. It can be seen from (b) that the distorted parts of speech synthesized by our DSE are shorter than that synthesized by the GST. And the noise in the silence span of synthetic speech from our DSE is less than that in the GST. It can be seen from (c) that the speech synthesis of GST synthesis cannot continue in the middle, while the synthesis of the DSE is continuous, which shows that the exposure bias of DSE is smaller than that of the GST. This means our model can more effectively resist the noise caused by reference speech.

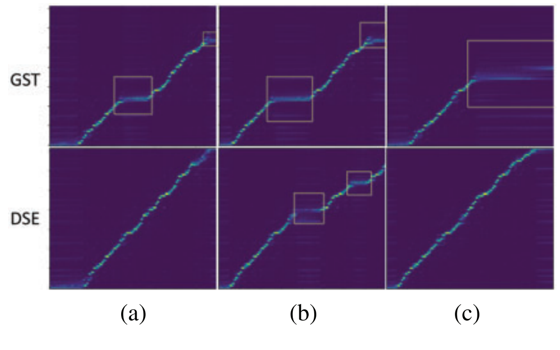


Figure 9: Alignment for the synthesized speech with input text "and it is worth mention in passing that as an example of fine typography". This speech is synthesized by GST model (top) and DSE (bottom) with three different references (a), (b), (c)

Fig. 10 shows the mel spectrogram of synthesized speech which corresponds to the alignment in Fig. 9. The connected blocks in Fig. 10 indicate a similar speech spectrum. The distorted parts in alignment correspond to the silence position in mel spectrograms. Fewer and shorter silence segments in the sentence can make the generated speech more coherent. The intonation, duration, and pause of the synthetic speech are different because the input reference speech features are different, so the spectrum cannot be the same. Especially the reference speech in (c) brings more noise, resulting in the spectrum of the synthetic speech being more blurred than the other two groups of speech. Synthesis speech is similar because they are synthesized with the same input text. The detail compare of synthesized speech (a) and the corresponding relationship between GST and DSE are shown in Fig. 11.

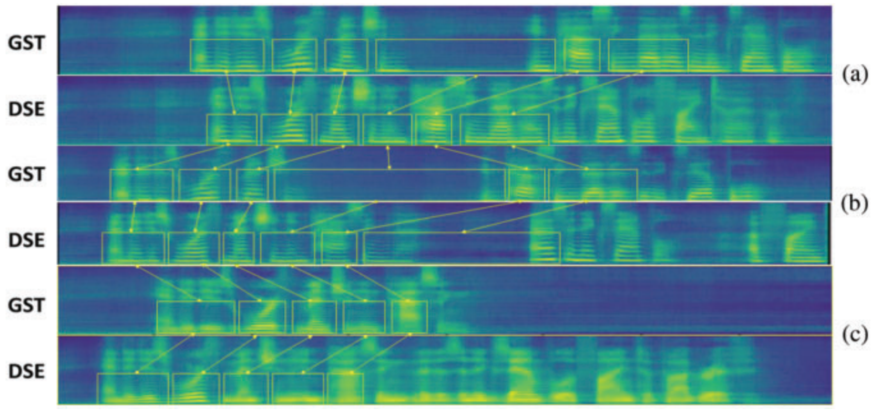


Figure 10: Mel spectrograms for the synthesized speech with input text "and it is worth mention in passing that as an example of fine typography" these speech is synthesized by GST model and DSE with three different references (a), (b), (c)

In Fig. 11, (a–i) and (1–9) represent nine parts of the mel spectrogram synthesized by DSE and GST. These parts are corresponding one by one. First, the silence segment separates the originally coherent speech like (d) in part (4). Second, the spectrogram of parts (6) and (8) are flatter, while the corresponding spectrogram of parts (f) and (h) changed a little more. This means that the speech tones synthesized by GST change less in part (6) and (8), while the speech tones synthesized by DSE change more in parts (f) and (h). This shows that DSE captures the suprasegmental features in more detail. The duration of the spectrogram in part (1) is longer than that in (a). Third, the synthetic duration of the first few words is shorter, which is more in line with people's speaking habits when the synthetic text is "and it is worth mention in passing that as an example of fine typography".

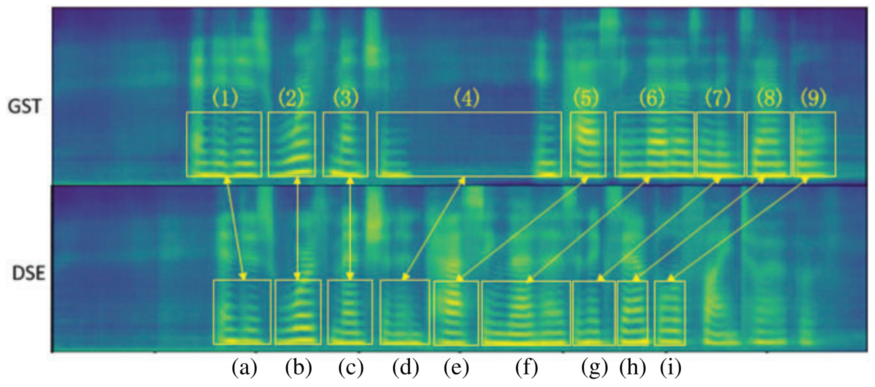


Figure 11: The detail compare of synthesized speech (a) and the corresponding relationship between GST and DSE

To quantitatively evaluate our DSE model, we train and test it on LJSpeech and LibriTTS datasets. First, we randomly sample 50 sentences from each dataset and then send the sampled sentences to GST and DSE for synthesis. Second, we further randomly sample 100 sentences from the LibriTTS dataset, of which 50 are used for synthesis and the other 50 are used as reference speech for prosodic transfer. In this way, we evaluate the performance of GST and DSE with different reference speech. Finally, we use three methods to quantitatively evaluate the synthesized speech, which are MOS (mean opinion score) [38], ABX test [39], and WER (word error rate). We expand the number of synthesized sentences in each experiment to 200, then send them to the pre-trained speech recognition model ESPNet [40] to get the WER. The results are shown in Table 1.

Table 1: The evaluate results

	LJSpeech			LibriTTS			LibriTTS + Prosody transfer		
Model	MOS	ABX	WER	MOS	ABX	WER	MOS	ABX	WER
GST	4.15	44.59%	11.72%	2.83	35.96%	18.71%	2.87	44.51%	18.37%
DSE	**4.20**	**55.41%**	**11.64%**	**3.29**	**64.04%**	**16.61%**	**3.44**	**55.49%**	**14.38%**

MOS score is the average of human scores on all speech. In the three experiments, all the scores of DSE are higher than those of GST, indicating the synthesized speech by DSE is closer to the real speech. The results of ABX represent the percentage of the number of people who support GST and who support DSE in all synthetic speech. More people supporting DSE than GST means that DSE performs better than GST. WER represents the percentage of word error rate in recognizing the synthesized speech. The smaller WER of DSE represents that the speech synthesized by DSE is clearer and easier to be recognized.

5 Discussion

The advantages of DSE are as follows:

1) DSE can get more detailed features by adjusting the scale. If you need word prosody information, you can set each window in DSE to correspond to a word. Similarly, if you need the prosody information of a phrase, you can set each window in the DSE to correspond to a phrase.
2) Frame-to-frame and speech-to-vector methods become exceptional cases of our proposed alignment method. If each window in DSE corresponds to a speech frame, DSE can be transformed into a frame-to-frame encoder. If only one window in the DSE corresponds to the entire sentence, the DSE can be transformed into a speech-to-vector encoder.
3) DSE reduces the interference caused by speech information added to the text. Therefore, DSE can synthesize more fluent speech than the GST method.

The disadvantages of DSE are as follows:

1) DSE needs a longer training time than traditional speech synthesis methods.
2) DSE cannot solve the problem of over migration. For example, if the female speech is used as the reference speech and the male speech is used as the synthesis baseline, the final synthesized

voice will be more inclined to the female voice. This means that the prosodic transfer is to transfer all the speech information, not only the prosodic features.

6 Conclusion

This paper proposes a model with an adjustable scale based on a new alignment approach named Dynamic Speech Encoder. Our model is based on the new proposed alignment approach in conjunction with the frame-by-frame encoding and speech-to-vector encoding models. Three experiments prove that our model achieves and optimizes the functions of the frame-by-frame and speech-to-vector encoding models.

The first experiment is the Speech Compress. This experiment is a predecessor experiment of the text embedding. If the speech compress experiment is unsuccessful, the text embedding experiment will also fail. The speech can be encoded more similarly to the text feature in the speech compress experiment. The compression ratio is higher than other existing speech coding models.

The second experiment is the Text Embedding experiment. In this experiment, we successfully extract text vectors from speech and use the extracted text vector to synthesize. Theoretically speaking, this experiment can project the text vector in a different language to a common space and let the speech from a different language be jointly trained. This experiment also can reduce the inconsistent between speech and text.

The third experiment is the Style Transfer experiment. In this experiment, the speech feature can change the synthetic speech prosody to reference speech prosody. This result illustrates the change of prosody in speech can be observed in speech feature and help speech synthesis system to synthesize prosody better.

In this paper, we only implement the function of extracting text vectors from speech. In the future, we can use the data set from a different language to train the model and use our Dynamic Speech Encoder to solve the actual problems. We also can continue to apply this system to the Transformer to speed up model training.

Funding Statement: This work was supported by National Key R&D Program of China (2020AAA0 107901).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N. et al. (2018). Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783. Calgary.
2. Wang, Y., Skerry-Ryan, R. J., Stanton, D., Wu, Y., Weiss, R. J. et al. (2017). Tacotron: Towards end-to-end speech synthesis. *Proceedings of the Interspeech 2017*, pp. 4006–4010. Stockholm.
3. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O. et al. (2016). WaveNet: A generative model for Raw audio. *Proceedings of the 9th International Symposium on Computer Architecture (ISCA) Speech Synthesis Workshop*, Seoul.
4. Jiao, Y., Gabryś, A., Tinchev, G., Putrycz, B., Korzekwa, D. et al. (2021). Universal neural vocoding with parallel wavenet. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6044–6048. Toronto.

5. Wu, H., Sangaiah, A. K. (2021). Oral English speech recognition based on enhanced temporal convolutional network. *Intelligent Automation and Soft Computing, 28(1),* 121–132. DOI 10.32604/iasc.2021.016457.
6. Thimmaraja, Y. G., Nagaraja, B. G., Jayanna, H. S. (2021). Speech enhancement and encoding by combining SS-VAD and LPC. *International Journal of Speech Technology, 24(1),* 165–172. DOI 10.1007/s10772-020-09786-9.
7. Skerry-Ryan, R. J., Battenberg, E., Xiao, Y., Wang, Y., Stanton, D. et al. (2018). Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 4693–4702. Stockholm.
8. Xue, L., Pan, S., He, L., Xie, L., Soong, F. K. (2021). Cycle consistent network for end-to-end style transfer TTS training. *Neural Networks, 140,* 223–236. DOI 10.1016/j.neunet.2021.03.005.
9. Wu, Y., Mao, W., Feng, J. (2021). AI for online customer service: Intent recognition and slot filling based on deep learning technology. *Mobile Networks and Applications,* 1–13. DOI 10.1007/s11036-021-01795-5.
10. Kenton, J. D. M. W. C., Toutanova, L. K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 4171–4186. Minneapolis.
11. Jia, Y., Zen, H., Shen, J., Zhang, Y., Wu, Y. (2021). PnG BERT: Augmented BERT on phonemes and graphemes for neural TTS. *Proceedings of the Interspeech 2021*, pp. 151–155. Brno.
12. Chen, L., Deng, Y., Wang, X., Soong, F. K., He, L. (2021). Speech bert embedding for improving prosody in neural tts. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6563–6567. Toronto.
13. Sun, K., Zhang, R., Mensah, S., Mao, Y., Liu, X. (2019). Aspect-level sentiment analysis via convolution over dependency tree. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5679–5688. Hong Kong.
14. Stanton, D., Wang, Y., Skerry-Ryan, R. J. (2018). Predicting expressive speaking style from text in end-to-end speech synthesis. *Proceedings of the 2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 595–602. Athens.
15. Wu, Y., Ma, Y., Wan, S. (2021). Multi-scale relation reasoning for multi-modal visual question answering. *Signal Processing: Image Communication, 96,* 116319. DOI 10.1016/j.image.2021.116319.
16. Li, N., Liu, S., Liu, Y., Zhao, S., Liu, M. (2019). Neural speech synthesis with transformer network. *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 6706–6713. Hawaii.
17. Liu, R., Sisman, B., Li, H. (2021). Graphspeech: Syntax-aware graph attention network for neural speech synthesis. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6059–6063. Toronto.
18. Basak, S., Agrawal, H., Jena, S., Gite, S., Bachute, M. et al. (2022). Challenges and limitations in speech recognition technology: A critical review of speech signal processing algorithms, tools and systems. *Computer Modeling in Engineering & Sciences, 135(2),* 1053–1089. DOI 10.32604/cmes.2022.021755.
19. Kano, T., Sakti, S., Nakamura, S. (2021). Transformer-based direct speech-to-speech translation with transcoder. *Proceedings of the 2021 IEEE Spoken Language Technology Workshop (SLT)*, pp. 958–965. Shenzhen.
20. Li, J., Ye, G., Das, A., Zhao, R., Gong, Y. (2018). Advancing acoustic-to-word CTC model. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5794–5798. Calgary.
21. Hagen, S., Liao, H., Sak, H. (2017). Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition. *Proceedings of the Interspeech 2017*, pp. 3707–3711. Stockholm.

22. Moritz, N., Hori, T., Le Roux, J. (2021). Semi-supervised speech recognition via graph-based temporal classification. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6548–6552. Toronto.
23. Luo, J., Du, Y. (2020). Detecting community structure and structural hole spanner simultaneously by using graph convolutional network based auto-encoder. *Neurocomputing, 410,* 138–150. DOI 10.1016/j.neucom.2020.05.039.
24. Du, Y., Zhou, Q., Luo, J., Li, X., Hu, J. (2021). Detection of key figures in social networks by combining harmonic modularity with community structure-regulated network embedding. *Information Sciences, 570,* 722–743. DOI 10.1016/j.ins.2021.04.081.
25. Baskar, M. K., Burget, L., Watanabe, S., Astudillo, R. F. (2021). EAT: Enhanced ASR-TTS for self-supervised speech recognition. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6753–6757. Toronto.
26. Lu, X., Tsao, Y., Matsuda, S., Hori, C. (2013). Speech enhancement based on deep denoising autoencoder. *Proceedings of the Interspeech 2013*, pp. 436–440. Lyon.
27. Lu, Y., Huang, M., Qu, X., Wei, P., Ma, Z. (2022). Language adaptive cross-lingual speech representation learning with sparse sharing sub-networks. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6882–6886. Singapore.
28. Wang, Y., Stanton, D., Zhang, Y., Ryan, R. S., Battenberg, E. et al. (2018). Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 5180–5189. Stockholm.
29. Bae, H., Bae, J. S., Joo, Y. S., Kim, Y. I., Cho, H. Y. (2021). A neural text-to-speech model utilizing broadcast data mixed with background music. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6603–6607. Toronto.
30. Jang, E., Gu, S., Poole, B. (2017). Categorical reparametrization with gumble-softmax. *Proceedings of the International Conference on Learning Representations (ICLR 2017)*, Singapore.
31. Gumbel, E. J. (1954). *Statistical theory of extreme values and some practical applications: A series of lectures*, vol. 33. USA: US Government Printing Office.
32. Zeng, Z. (2020). Implementation of embedded technology-based English speech identification and translation system. *Computer Systems Science and Engineering, 35(5),* 377–383. DOI 10.32604/csse.2020.35.377.
33. Li, G., Yu, Y. (2016). Deep contrast learning for salient object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 478–487. Las Vega.
34. Ito, K., Johnson, L. (2017). The LJSpeech dataset. https://keithito.com/LJ-Speech-Dataset/.
35. Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R. J. et al. (2019). LibriTTS: A corpus derived from LibriSpeech for text-to-speech. *Proceedings of the Interspeech 2019*, pp. 1526–1530. Graz.
36. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pp. 1096–1103. Helsinki.
37. Ren, Y., Ruan, Y., Tan, X., Qin, T., Zhao, S. et al. (2019). Fastspeech: Fast, robust and controllable text to speech. *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 3171–3180. Vancouver.
38. Streijl, R. C., Winkler, S., Hands, D. S. (2016). Mean opinion score (MOS) revisited: Methods and applications, limitations and alternatives. *Multimedia Systems, 22(2),* 213–227. DOI 10.1007/s00530-014-0446-1.
39. Munson, W. A., Gardner, M. B. (1950). Standardizing auditory tests. *The Journal of the Acoustical Society of America, 22(5),* 675–675. DOI 10.1121/1.1917190.
40. Mehta, S., Rastegari, M., Shapiro, L., Hajishirzi, H. (2019). Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9190–9200. Long Beach.