JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 40, 189-200 (2024)
DOI: 10.6688/JISE.202401_40(1).0012

Improving Speech Synthesis by Automatic Speech Recognition and Speech Discriminator

LI-YU HUANG AND CHIA-PING CHEN
Department of Computer Science and Engineering
National Sun Yat-sen University
Kaohsiung, 804 Taiwan
E-mail: m093040070@nsysu.edu.tw; cpchen@cse.nsysu.edu.tw

Speech synthesis (text-to-speech, TTS) and automatic speech recognition (ASR) are opposite tasks yet they can be complementary. In our work, we try to improve the TTS by using ASR. ASR plays the role of verifying the output of TTS. It compares the recognized text with the ground-truth text for an ASR loss to penalize TTS. In our experiments, TTS without ASR scored 3.96 mean opinion score (MOS), and with ASR it achieved 4.21 MOS. We also enhanced TTS using the architecture of discriminator in generative adversarial networks (GANs). By adding a speech discriminator to discriminate the mel-spectrogram synthesized by the synthesizer, it can change the learning of TTS and improve the quality of the synthesized speech. In our experiments, TTS with speech discriminator scored 4.26 MOS. Finally, our best TTS system used both ASR and speech discriminator in the synthesizer model, and it reached 4.29 MOS.

Keywords: text-to-speech, automatic speech recognition, discriminator, speech synthesis, generative adversarial network

1. INTRODUCTION

TTS is a technology that converts input text to corresponding speech. TTS based on neural networks has made long progress over recent years and matured gradually. In particular, end-to-end TTS can directly learn the correspondence between the text and the acoustic feature. A fairly popular model architecture is Tacotron2 [1]. It is an end-to-end recurrent neural network (RNN) with an attention mechanism [2] and it can predict a sequence of mel-spectrogram based on the input character sequence. FastSpeech2 [3] is a feedforward neural network based on transformer. In comparison to Tacotron2 with the autoregressive model, FastSpeech2 adopts non-autoregressive model to improve the speed of synthesis without degrading the overall quality of the synthesized speech.

In this work, we improve TTS through integration of automatic speech recognition and speech discriminator. Certain related ideas have been proposed recently. Liu *et al*. [4] improved the problem of style adjustment in TTS by adding a pre-trained ASR model to provide loss of ASR during training. Nakayama *et al*. [5] uses TTS and ASR to form a speech chain to achieve the effect of semi-supervised learning. Discriminator is neural network architecture used in GANs [6]. GANs is an unsupervised learning method that

Received October 16, 2022; accepted January 12, 2023.
Communicated by Berlin Chen.

achieves the purpose of learning through the generator and the discriminator against each other. Although the generator in GAN is different from synthesizer conceptually, we can still exploit the idea of a discriminator to guide TTS to learn to be more human-like.

In this work, we first add ASR to the training of the synthesizer in TTS, and call this method TTS-ASR. Then we add the speech discriminator, which is used to discriminate the mel-spectrogram, into the training of the synthesizer, and call this method TTS-SD. Finally, we combine TTS-ASR and TTS-SD to build TTS-ASR-SD based on ideas inspired by CycleGAN [7]. CycleGAN, a special kind of GAN architecture, is used in image-to-image translation. It uses two generators, which convert the input image from the domain A to the domain B and from the domain B to the domain A respectively, and then use two discriminators to identify the two outputs of the generators. Although the domains A and B are two different types of images, we boldly replace them with the domains of speech and text, then implement the ideas in system development.

The remainder of this paper consists of four sections. In Section 2, the model architectures and methods, including TTS-ASR, TTS-SD and TTS-ASR-SD, are introduced. In Section 3, the datasets and the experimental setup for the development of TTS systems are explained. In Section 4, experimental results and analyses are provided. In Section 5, we draw conclusion and state future work.

2. METHODS

The methods for training TTS synthesizer, namely TTS-ASR, TTS-SD and TTS-ASR-SD, are implemented. These methods progressively add additional modules to the training stage of the synthesizer. In the inference (synthesis) stage, only the synthesizer (TTS) module is in action. We first introduce the TTS architecture, and then explain other methods based on TTS.

2.1 TTS

We use Conformer-FastSpeech2 for the synthesizer of TTS, paired with HiFi-GAN [8] as vocoder. Its architecture is shown in the left of Fig. 1. FastSpeech2 is a non-autoregressive model that was proposed to accelerate the speed of the inference time and synthesize speech with the same quality as autoregressive models. Feedforward transformer (FFT) architecture is applied in its encoder and decoder. The variance adaptor contains predictors to predict variance information needed to synthesize the speech, such as the energy, pitch and duration. Duration represents the length of the speech, *i.e.* how long the sound should be made. Pitch is an important key to affecting the prosody of speech. Energy represents the strength of the voice, which directly affects the volume and rhythm of the voice. In particular, the output of the duration predictor will go through the length regulator. The length regulator will expand the input embedding sequence multiple times according to the duration sequence predicted by the duration predictor, so as to achieve the effect of changing the length of the speech. It is used to solve the problem of aligning the input syllable sequence with the output mel-spectrogram sequence. The output of the decoder will map the result to the mel-spectrogram through 1-layer linear. Postnet will add the output of the decoder and the output of the linear mapping for residual calculation as the final output, and its architecture consists of 5-layer 1D convolution.

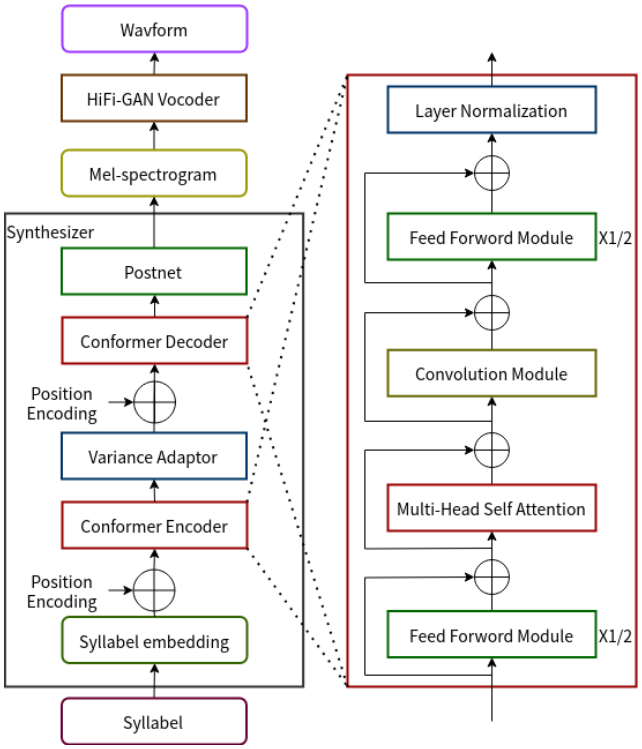


Fig. 1. Architecture of TTS. It consists of two parts, synthesizer and vocoder. Synthesizer converts the text to mel-spectrogram, and vocoder converts the mel-spectrogram to speech. We use Conformer-FastSpeech2 as synthesizer and HiFi-GAN as vocoder.

Conformer-FastSpeech2 is a model after replacing the FFT architecture in FastSpeech2 with the conformer [9]. The transformer based on the attention mechanism has great performance in extracting the dependencies of long input sequences. Conformer adds a convolution layer that extracts local features, which makes its performance better. The architecture of the conformer encoder and conformer decoder is shown in the right of Fig. 1. It is composed of macaron style feed forward network (macaron style FFN) [10], multi-head attention and convolution module. The macaron style FFN module will only contribute half the value of FNN before and after the convolution module. Increase the attention ability of the model in multiple different subspaces through multi-head attention.

The loss function used in training the Conformer-FastSpeech2 consists of multiple terms

$$Loss_{\text{tts}} = Loss_{\text{mel}} + Loss_{\text{duration}} + Loss_{\text{pitch}} + Loss_{\text{energy}} \tag{1}$$

where $Loss_{\text{tts}}$ is the total loss of synthesizer. $Loss_{\text{mel}}$ is the loss of mel-spectrogram, which uses mean absolute error (MAE) as loss function. $Loss_{\text{duration}}$, $Loss_{\text{pitch}}$, and $Loss_{\text{energy}}$ are the losses of the duration predictor, pitch predictor, and energy predictor respectively, all of them use mean square error (MSE) as a loss function.

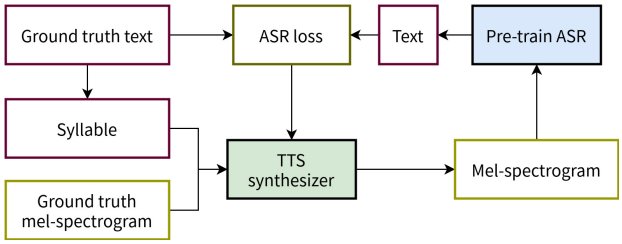


Fig. 2. Architecture of TTS-ASR. ASR receives the mel-spectrogram from synthesizer, calculates the ASR loss, and feeds back to the synthesizer. Note ASR is only used during the training stage.

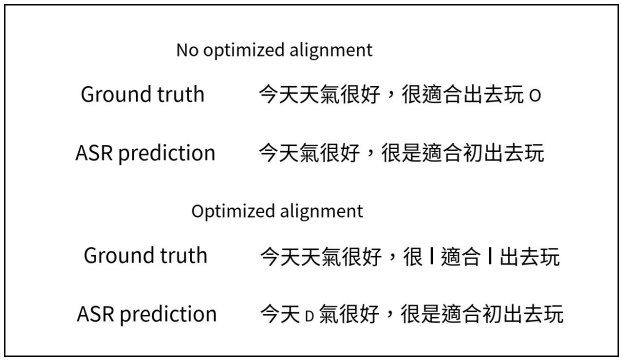


Fig. 3. Example of optimal alignment between ASR output and ground truth label for calculating ASR loss. “O” means space, “I” means insertion, and “D” means deletion.

2.2 TTS-ASR

We add a pre-trained ASR and freeze its parameters when training the synthesizer of TTS. ASR receives the mel-spectrogram synthesized by the synthesizer, and predicts the corresponding text. The text predicted by ASR and the ground-truth text is compared to yield ASR loss, which is added as a penalty to modify model training and enhance the synthesizer. This is shown in Fig. 2.

Since the length of the text output by ASR may be different from the text length of the ground-truth, optimal alignment between ASR predicted text and ground-truth text is carried out for calculating the ASR loss. See Fig. 3 for an example regarding the determination of character error rate (CER).

We use the cross-entropy function for the ASR loss

$$Loss_{\text{asr}} = CEL(ASR(M_{\text{tts}}), T_g) \tag{2}$$

where M_{tts} is the mel-spectrogram output of the synthesizer, and T_g is the ground-truth text. And the loss of ASR will be added to the loss of the synthesizer

$$Loss_{\text{tts-asr}} = Loss_{\text{tts}} + Loss_{\text{asr}} \tag{3}$$

where $Loss_{\text{tts}}$ is the loss of the synthesizer as given in Eq. (1).

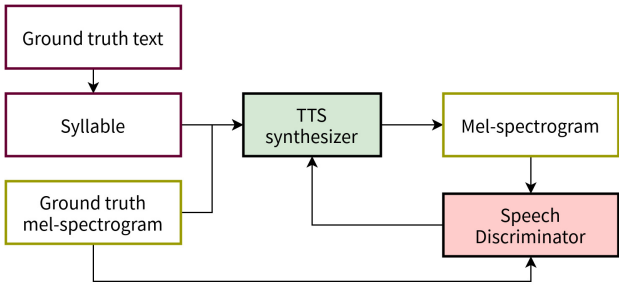


Fig. 4. Architecture of TTS-SD. Using a speech discriminator, we hope the synthesis output can be indistinguishable from real human speech.

2.3 TTS-SD

We added a speech discriminator when training the synthesizer, and its architecture is shown in Fig. 4. Speech discriminator identifies whether the input mel-spectrogram is from synthesizer or from genuine speech. It provides a loss to enhance the training of the synthesizer. Speech discriminator consists of two parts, convolution block and post-processing. The convolution block does padding first and then goes through 1D convolution followed by 1D batch normalization, ReLU function and dropout. The post-processing part performs average pooling, reduces the output size through the fully connect layer, and uses the sigmoid function to map the output to a value between 0 (fake/synthetic) and 1 (real/ground-truth).

Speech discriminator is only used during the training stage. It receives ground-truth mel-spectrogram and mel-spectrogram synthesized by the synthesizer at every iteration. According to the result, the loss is returned to itself and the synthesizer. We use the binary cross-entropy (BCE) functions

$$Loss_{\mathrm{s}} = BCE(SD(m_{\mathrm{s}}), 1) \tag{4}$$

$$Loss_{\mathrm{sd}} = 1/2(BCE(SD(m_{\mathrm{s}}), 0) + BCE(SD(m_{\mathrm{g}}), 1)) \tag{5}$$

where $Loss_{\mathrm{s}}$ and $Loss_{\mathrm{sd}}$ are the loss for synthesizer and speech discriminator, respectively. The function $SD(x)$ represents the result obtained by x through the speech discriminator, and m_s and m_g are the mel-spectrogram from ground-truth and synthesizer respectively.

When training TTS-SD, we use multi-task learning, adding $Loss_{\mathrm{s}}$ to the loss of the synthesizer, which can be written as

$$Loss_{\text{tts-sd}} = Loss_{\text{tts}} + Loss_{\mathrm{s}} \tag{6}$$

where $Loss_{\text{tts}}$ is the loss of the synthesizer.

2.4 TTS-ASR-SD

CycleGAN is a GAN-based approach for image-to-image translation. The task of image-to-image translation is to map the input image to the output image given a learning set of images. But in most cases, the input image of the training data is difficult to match with the output image. CycleGAN is trained using data set from two domains without

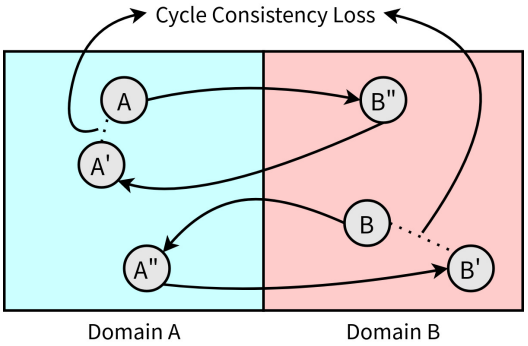


Fig. 5. The cycle consistency loss. It measures the difference between switching back and forth, *i.e.* forming a cycle, in two domains.

strict correspondence between the data in different domains. Its architecture contains two generators G_{A2B} and G_{B2A}, which convert the images in domain A to domain B and convert the images in domain B to domain A, respectively. In addition, two discriminators D_B and D_A identify whether the generated image is in the domain B or domain A after the conversion. The loss used in the training of CycleGAN is divided into two parts, namely the generative adversarial loss and the cycle consistency loss. The generative adversarial loss is the loss against the generator and the discriminator in the generative adversarial network architecture

$$\begin{aligned} Loss_{\text{gan}} &= L_{GAN}(G_{A2B}, D_B, A, B) + L_{GAN}(G_{B2A}, D_A, A, B) \\ &= E_b[\log D_B(b)] + E_a[\log(1 - D_B(G_{A2B}(a)))] \\ &+ E_a[\log D_A(a)] + E_b[\log(1 - D_A(G_{B2A}(b)))]. \end{aligned} \tag{7}$$

Note the generative adversarial loss consists of two parts since CycleGAN consists of two architectures in opposite directions. For the cycle consistency loss, since the goal is to minimize the difference between A and A$'$ in the process of converting A to B and then converting from B to A$'$, ideally we want to have $G_{B2A}(G_{A2B}(a)) = a$. The same is true for the other direction. It is shown in Fig. 5. Thus, it is reasonable to adopt

$$Loss_{\text{cycle}} = E_a||G_{B2A}(G_{A2B}(a)) - a||_1 + E_b||G_{A2B}(G_{B2A}(b)) - b||_1 \tag{8}$$

for the cycle consistency loss function. Combining generative adversarial loss and the cycle consistency loss, we have

$$Loss_{\text{total}} = Loss_{\text{gan}} + Loss_{\text{cycle}} \tag{9}$$

Inspired by CycleGAN, we further integrate TTS-ASR and TTS-SD into a new architecture called TTS-ASR-SD. This is shown in Fig. 6. First, the loss provided by the speech discriminator is used, just like the generative adversarial loss provided by the discriminator to the generator in CycleGAN. Second, the loss provided by ASR is defined by

$$Loss_{\text{asr}} = BCE(ASR(TTS(t)), t) \tag{10}$$

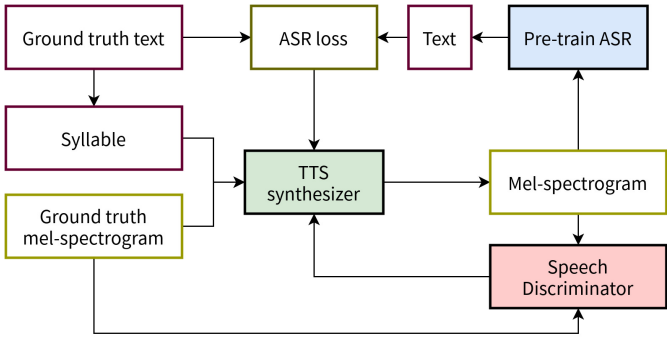


Fig. 6. Architecture of TTS-ASR-SD. The idea is inspired by CycleGAN. It consists of three parts: TTS that converts text to speech, ASR that converts speech to text, and speech discriminator.

as the goal is to achieve $ASR(TTS(t)) = t$. It corresponds to the cycle consistency loss in CycleGAN. Since the training data we use is matched with corresponding text, we do not use complete unsupervised learning. Instead, we add the loss from speech discriminator and the loss of ASR to the synthesizer loss. Thus the loss function of TTS-ASR-SD can be written as

$$Loss_{\text{tts-asr-sd}} = Loss_{\text{tts}} + Loss_{\text{cycle}} + Loss_{\text{gan}} = Loss_{\text{tts}} + Loss_{\text{asr}} + Loss_{\text{sd}}. \tag{11}$$

3. EXPERIMENT

We use the software toolkit of ESPnet2 [11] for system implementation. In the experiments we use the same vocoder to compare synthesizers trained by different methods. We describe the key system modules in details below.

3.1 Datasets

Biaobei [12] is a Mandarin dataset with a single female speaker, containing $10,000$ utterances with the total time of about 12 hours. Among them, to improve the speech synthesis ability of short sentences, we cut 200 utterances into multiple short-utterance data and added them to the original data set as our training data set. We call it Biaobei+ which contains $11,112$ utterances with the total time of about 12.5 hours.

3.2 Experiment Setting

In our TTS systems, the vocoder and synthesizer are trained separately. The vocoder uses Biaobei as training data and the synthesizer uses Biaobei+ as training data. Conformer-FastSpeech2 uses 4-layer conformer encoder and 4-layer conformer decoder. The kernel sizes of the depth-wise convolution in conformer encoder and conformer decoder are 7 and 31. The used optimizer is Adam [13], and the learning rate is set to 1. In the vocoder, we set the batch size to 16 and use Adam as optimizer, and the learning rate is set to 2×10^{-4}. The speech discriminator uses 4-layer convolution block. And the (input channel, output channel) of the 1D convolution are $(80,128)$ and $(128,128)$ respectively, the kernel size of them is 5.

The computation of ASR loss requires optimal alignment between text sequences. To reduce the overall training time, we use the fine-tune method in transfer learning [14]. In TTS-ASR and TTS-ASR-SD, the synthesizers without ASR is pre-trained with 180 epochs and fine-tuned with 20 epochs with ASR. In order to avoid the speech discriminator with poor discriminative ability dragging down the training of the synthesizer in the early stage of training, We also bootstrap TTS-SD and TTS-ASR-SD with pre-training. The speech discriminator is trained with TTS-SD for 200 epochs, and then used to re-train the synthesizer in TTS-SD or TTS-ASR-SD.

3.3 Evaluation

We used the MOS measure in TTS system evaluation. The score is between 0 to 5 and one decimal place. The average of the scores is rounded to the two decimal places. We picked three sentences in Mandarin as test data. The sentences are taken from different categories of news or fragments of articles be found in the Internet. We used these sentences as test material and played to the 9 members of our team who scored fairly. In addition, we also plotted the waveform of synthesized speech to observe and annotate the subtle differences in the outputs of the systems.

We found that the performance of different architectures in short texts is not very different, except for speech quality. But some noticeable problems can be heard in long texts. We divide the problem into two types: pronunciation errors and sentence segmentation errors. The pronunciation error means that the synthesized speech does not match the text. This is because there is heteronym in Mandarin, that is, the same character but there are multiple pronunciations. The sentence segmentation error refers to the wrong position of the pause, such as a pause in speech before a word is finished. We use long test sentences and manually annotate the errors. We report the occurrence of the two errors in test sentences.

4. RESULTS AND ANALYSIS

The results in MOS of the proposed methods are shown in Table 1. The MOS of TTS is 3.96, while the MOS of TTS-ASR is 4.21. The MOS of TTS-SD is 4.26, and the MOS of TTS-ASR-SD is 4.29. Examples of the speech waveforms synthesized by the proposed methods are shown in Figs. 7 and 8. Five sentences, each containing 40–60 words, are used to evaluate the performance regarding pronunciation errors and sentence segmentation errors. The results are summarized in Table 2 and two examples are shown in Fig. 9.

Table 1. Results in MOS of the proposed methods. The addition of ASR and speech discriminator improved MOS. The TTS-ASR-SD method achieved the highest MOS score among them.

TTS	TTS-ASR	TTS-SD	TTS-ASR-SD
3.96	4.21	4.26	4.29

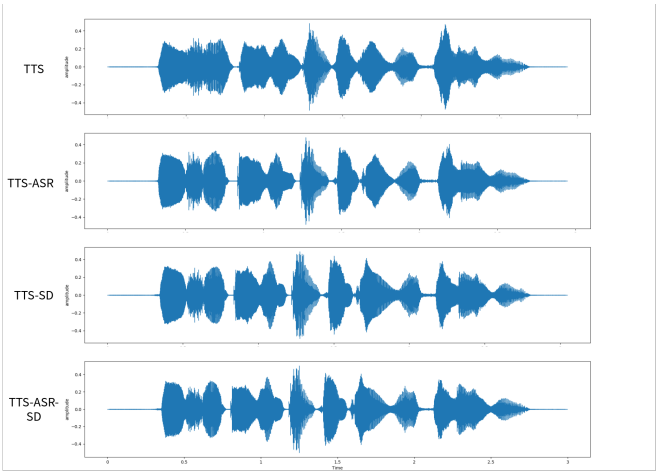


Fig. 7. The synthesized speech waveforms for input text ”陰時多雲短暫陣雨或雷雨” of the proposed methods.

4.1 TTS-ASR

TTS-ASR achieves significant improvement in MOS over the TTS baseline. In TTS-ASR, the waveforms are relatively loose, while the waveforms in TTS are more dense. Its is because ASR loss is the loss of the text, which penalizes TTS and corrects its articulation and intelligibility of speech. Listening to the waveforms, we feel the pronunciation of words in TTS-ASR results sounds clearer and more rhythmic. That is, the results in TTS are relatively dull and not as rhythmic as in TTS-ASR. Regarding pronunciation error and sentence segmentation error, TTS-ASR is not particularly effective over TTS. The ASR loss is based on single words and has no direct relationship with the context. Therefore, it does not have a special performance in sentence segmentation error.

4.2 TTS-SD

TTS-SD also achieves significant improvement in MOS over the TTS baseline. This is because the addition of a speech discriminator guides the synthesizer to output data that is indistinguishable from real human speech. In TTS-SD, the sentence segmentation error has been significantly improved. This is because TTS-SD can synthesize speech that is closer to real speech than TTS, so the rhythm of the speech becomes more natural.

Table 2. The pronunciation errors and sentence segmentation errors of the proposed methods. Pronunciation error means that the synthesized speech does not fit the text correctly. Sentence segmentation error refers to inappropriate pauses.

	TTS	TTS-ASR	TTS-SD	TTS-ASR-SD
pronunciation error	8	7	7	7
sentence segmentation error	7	7	3	2

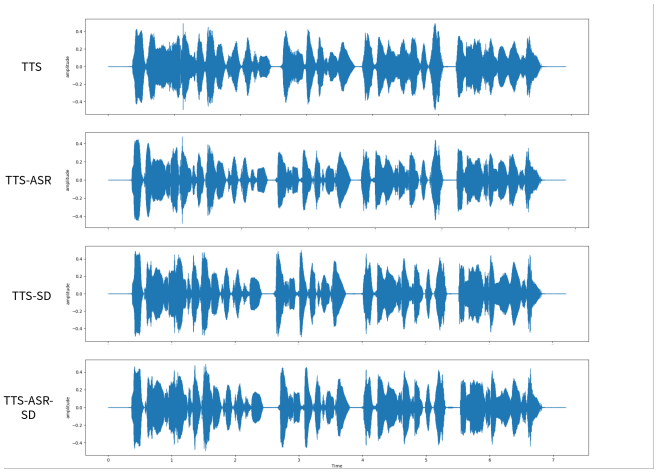


Fig. 8. The synthesized speech waveforms for input text ”美國運輸安全管理局指出，在疏散過程中有3人受傷，不過都沒有生命危險” of the proposed methods.

(a)	
TTS	今天至明天下午天氣好轉、氣O溫升，各地大都為多雲時晴，僅部分地區偶O有零星短暫雨的機率
TTS-ASR	今天至明天下午天氣好轉、氣O溫升，各地大都為多O雲時晴，僅部分地區偶O有零星短暫雨的機率
TTS-GANs	今天至明天下午天氣好轉、氣溫升，各地大都為多雲時晴，僅部分地區偶有零星短暫雨的(機)率
TTS-ASR-GANs	今天至明天下午天氣好轉、氣溫升，各地大都為多雲時晴，僅部分地區偶有零星短暫雨的(機)率
(b)	
TTS	物理學是最(接)近天文學的科學，實際上天文學是物理學的一個分支，天文學(家)運O用物理學的知(識)和定律，來O解釋我們所觀測到的現象
TTS-ASR	物理學是(最)接近天文學的科學，實際上天文學是物理學的一個分支，天文學(家)運用物理學的知(識)和定律，來O解釋我們所觀測到的現象
TTS-GANs	物理學是最(接)近天文學的科學，實際上天文學是物理學的一個分支，天文學家運用物理學的知(識)和定律，來O解釋我們所觀測到的現象
TTS-ASR-GANs	物理學是最(接)近天文學的科學，實際上天文學是物理學的一個分支，天文學家運用物理學的知(識)和定律，來解釋我們所觀測到的現象

Fig. 9. Examples of pronunciation error and sentence segmentation error. Word “()” flags pronunciation error, and “O” flags sentence segmentation error.

4.3 TTS-ASR-SD

TTS-ASR-SD achieves the highest score in MOS among the proposed methods. It also has the best performance on pronunciation error and sentence segmentation error.

TTS-ASR-SD combines TTS-ASR and TTS-SD in architecture, so it can benefit from both ideas. From the waveforms, we can see that TTS-ASR-SD has the characteristics of TTS-ASR, and the improvement of TTS-SD in sentence segmentation error.

5. CONCLUSION

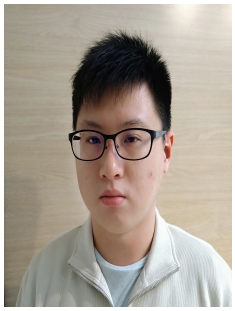
In this work, we improve TTS by adding ASR and speech discriminator during training. When we add a pre-trained ASR, the intelligibility of the synthesized speech is enhanced. Furthermore, the rhythm of speech synthesized by TTS is improved. When we add a speech discriminator, the naturalness of speech is significantly improved. Finally, we use both ASR and speech discriminator during training. It retains the above-mentioned advantages and achieves further improvements.

One thing to try is to jointly train TTS and ASR, instead of unilaterally. Another direction is to experiment with different discriminator architectures. The speech discriminator in this work has a simple neural network architecture. We can try a more complex architecture and the method of TTS-ASR-SD can become more comprehensive.

REFERENCES

1. J. Shen *et al.*, "Natural TTS synthesis by conditioning wavenet on mel-spectrogram predictions," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 4779-4783.
2. A. Vaswani *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, Vol. 30, 2017.
3. Y. Ren *et al.*, "Fastspeech 2: Fast and high-quality end-to-end text to speech," *arXiv Preprint*, 2020, arXiv:2006.04558.
4. D.-R. Liu, C.-Y. Yang, S.-L. Wu, and H.-Y. Lee, "Improving unsupervised style transfer in end-to-end speech synthesis with end-to-end speech recognition," in *Proceedings of IEEE Spoken Language Technology Workshop*, pp. 640-647.
5. S. Nakayama, A. Tjandra, S. Sakti, and S. Nakamura, "Speech chain for semi-supervised learning of Japanese-English code-switching ASR and TTS," in *Proceedings of IEEE Spoken Language Technology Workshop*, 2018, pp. 182-189.
6. I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, Vol. 27, 2014, pp. – –.
7. J.-Y. Zhu, T. Park, P. Isola, and A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of IEEE International Conference on Computer Vision*, 2017, pp. 2223-2232.
8. J. Kong, J. Kim, and J. Bae, "HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, Vol. 33, 2020, pp. 17 022-17 033.
9. A. Gulati *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *arXiv Preprint*, 2020, arXiv:2005.08100.
10. Y. Lu *et al.*, "Understanding and improving transformer from a multi-particle dynamic system point of view," *arXiv Preprint*, 2019, arXiv:1906.02762.

11. S. Watanabe *et al.*, "Espnet: End-to-end speech processing toolkit," *arXiv Preprint*, 2018, arXiv:1804.00015.
12. DataBaker Technology, "Chinese standard Mandarin speech corpus," https://www.data-baker.com/open source.html, 2017.
13. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv Preprint*, 2014, arXiv:1412.6980.
14. S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, 2009, pp. 1345-1359.



Li-Yu Huang received BS degree in Computer Science and Information Engineering from Tamkang University. He also earned the MS degree in Computer Science and Engineering from National Sun Yat-sen University. He has worked on speech synthesis in the Multimedia Information Technology Lab during his graduate study. His research topics include code-switching speech synthesis and methods for training speech synthesis with speech recognition and speech discriminators.

Chia-Ping Chen received the BS degree in Physics from National Taiwan University, the MS degree in Physics from National Tsing-Hua University, and the MS and Ph.D. degrees in Electrical Engineering from the University of Washington at Seattle. Since 2005, he has been a member of the faculty of the Department of Computer Science and Engineering of National Sun Yat-sen University, Kaohsiung, Taiwan. His research interests mainly focus on spoken language technology and applications, including speech recognition, speech synthesis, speaker recognition, and acoustic sound event detection.