JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 40, 359-373 (2024)
DOI: 10.6688/JISE.202403_40(2).0010

Training Speech Recognition Model with Speech Synthesis and Text Discriminator

HOU-AN LIN AND CHIA-PING CHEN
Department of Computer Science and Engineering
National Sun Yat-sen University
Kaohsiung, 804 Taiwan
E-mail: m093040066@nsysu.edu.tw; cpchen@cse.nsysu.edu.tw

In this paper, we build neural-network model-based automatic speech recognition (ASR) systems incrementally for performance improvement. First, we add an adversarial text discriminator module to train the speech recognition model to correct typos in recognition results. Experiments show that the character error rate (CER) and word error rate (WER) of the ASR system achieved 12.3% and 31.4%. Second, we insert a pre-trained speech synthesis (text-to-speech, TTS) module to the ASR model. When we exploit a pre-trained TTS in ASR training, the CER and WER are reduced from 12.6% and 31.7% to 10.8% and 24.4%, demonstrating that pre-trained TTS can improve ASR. Finally, we include both pre-trained TTS and text discriminator in ASR training. The performance of this ASR system is further improved, achieving the CER and WER of 9.9% and 22.7% respectively. On Formosa Speech Recognition Challenge task using Taibun Hàn-jī transcription, the proposed method also achieves better CER than a system based on hybrid DNN-HMM chain model.

Keywords: automatic speech recognition, text to speech, adversarial text discriminator, DNN-HMM chain model, formosa speech recognition challenge

1. INTRODUCTION

Classical speech recognition systems are often based on hidden Markov model (HMM) [1] and Gaussian mixture model (GMM) [2]. Subsequently, deep neural networks (DNN) are exploited to estimate the posterior probabilities, which are integrated into HMM for decoding. Recent ASR systems include features such as end-to-end (E2E) models, encoder-decoder framework, self-attention mechanism, and block-synchronous decoding for streaming speech recognition. Indeed, end-to-end models are well-known to the research community, *e.g.* connectionist temporal classification (CTC) [3], attention-based models [4–7], recurrent neural network transducer (RNN-T) [8], and the hybrid CTC/attention [9–11] architectures.

In recent years, GAN [12] models have tremendous impact on computer graphics and image processing. GAN is an architecture comprises a generator and a discriminator. Playing adversarial roles in model training, both the generator and the discriminator parts can be better trained. In particular, CycleGAN [13] uses two generators in the model architecture. The two generators first convert the input from domain X to domain Y, and then convert back to domain X from domain Y. It uses two discriminators to identify

Received October 16, 2022; revised December 27, 2022 & February 16, 2023; accepted March 4, 2023.
Communicated by Jen-Tzung Chien.

whether an image is genuine or artificial (*i.e.* from the generator). Instead of the image domains, we apply the ideas to the domains of text and speech, and modify the system architecture to incorporate adversarial training.

In this study, we begin with state-of-the-art conformer models [6] for speech recognition baseline systems. Transformer [5] is an end-to-end architecture that replaces recurrent neural network (RNN) with self-attention mechanism. Conformer is essentially transformer with convolution stems to better extract contextual information. Furthermore, we use the streaming conformer encoder with contextual block processing [14] and block-wise synchronous beam search [15] towards real-time recognition output.

On the conformer models, we propose the integration of adversarial methods in ASR model training for speaker-dependent scenarios. We first introduce an adversarial text discriminator, and the architecture is called ASR-ADV. The motivation of using a text discriminator is to reduce spelling errors. The second integration is a pre-trained TTS model, and the architecture is called ASR-TTS. Combination of TTS and ASR models forms a cycle from text to text via speech or from speech to speech via text. The ASR model can be improved if we demand the speech after the cycle is similar to the original speech. Note that similar ideas have been proposed [16]. Finally, we integrate text discriminator and TTS in conformer ASR model, called ASR-TTS-ADV, for further improvement.

The rest of this paper is organized as follows. In Section 2, we present the E2E model and training methods. In Section 3, we explain our experimental setup and the dataset used. In Section 4, we analyze and discuss the experimental results. In Section 5, we draw conclusion for this work.

2. METHODS

We make incremental improvements on the conformer baseline ASR system. Four models, namely ASR, ASR-ADV, ASR-TTS, and ASR-TTS-ADV, are introduced below.

2.1 Text Preprocessing

The preprocess block is a module for word segmentation, since pauses need to be added for the pre-trained TTS to work. We use the Jieba toolkit [17] to preprocess the input text to word sequence add pauses between words.

2.2 ASR End-to-End Model

The end-to-end ASR model is based on conformer encoder and transformer decoder, and the architecture is shown in Fig. 1. The input acoustic feature is a sequence of 80-dim mel-spectrogram. The input sequence is downsampled with a 2-layer subsampling CNN with stride 2. The CNN kernel size is 3 and the number of channels is 256. The encoder consists of 12-layer conformer architecture. Furthermore, the conformer architecture processes signal with split-and-add macaron method [18] to extract information.

The decoder consists of 6-layer transformer architecture. The decoder receives the encoder output X_e and the prefix of the output sequence. Given the prefix output sequence and X_e, the posterior probability of the entire output sequence Y is

$$p_{\text{e2e}}(Y|X_e) = \prod_u p_{\text{e2e}}(Y_{u+1}|Y_{1:u}, X_e) \tag{1}$$

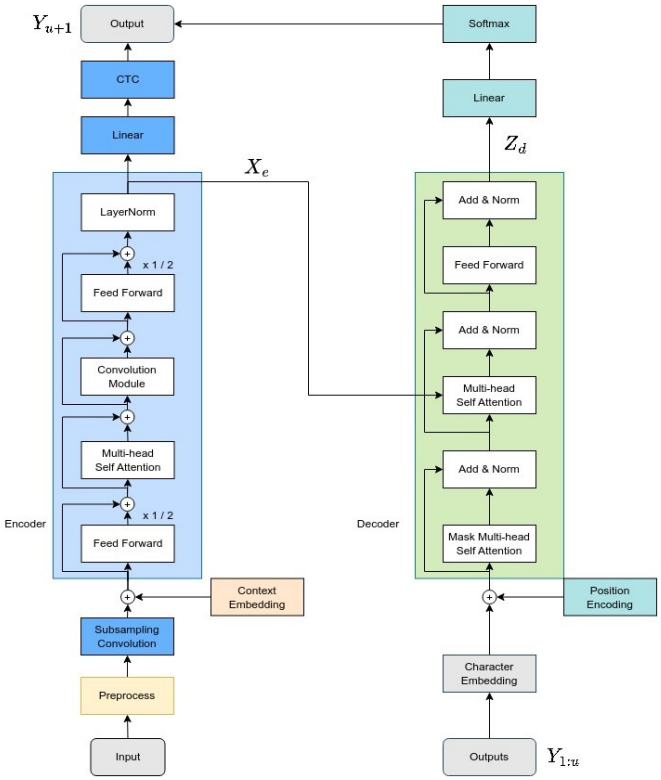

Fig. 1. The end-to-end ASR architecture; The encoded output is received by the transformer decoder and the connectionist temporal classification (CTC) module.

$$p_{\text{e2e}}(Y_{u+1}|Y_{1:u},X_e) = \text{softmax}(Z_d W_{\text{att}} + b_{\text{att}}) \tag{2}$$

where Z_d is the decoder output, $W_{\text{att}} \in \mathbb{R}^{d_{\text{att}} \times d_{\text{dic}}}$ and $b_{\text{att}} \in \mathbb{R}^{d_{\text{dic}}}$ are attention weight matrix and bias for computing Y_{u+1}, and d_{att} and d_{dic} are decoder output and dictionary size.

2.3 Streaming Method

We use contextual block processing and blockwise synchronous beam search methods for a streaming ASR system. Contextual block processing divides all input frames into blocks. Each block divides the internal frames into past, current, and future parts. The current block can refer to the past and future blocks. It has an inheritance mechanism allowing the contextual embedding to be passed to the next block. The attention-based decoder often predicts the end-of-sequence token prematurely or predicts duplicated tokens [9]. The case of duplicated tokens happens because the attention mechanism of the decoder processes a position that has already been attended. The end-of-sequence is prematurely predicted because the attentions reach the place where the encoder block is insufficient. Therefore, blockwise synchronous beam search with block boundary detection (BBD) mechanism is adopted to determine whether the next token predicted by the current block is reliable. If BBD determines that it is unreliable, it will discard the cur-

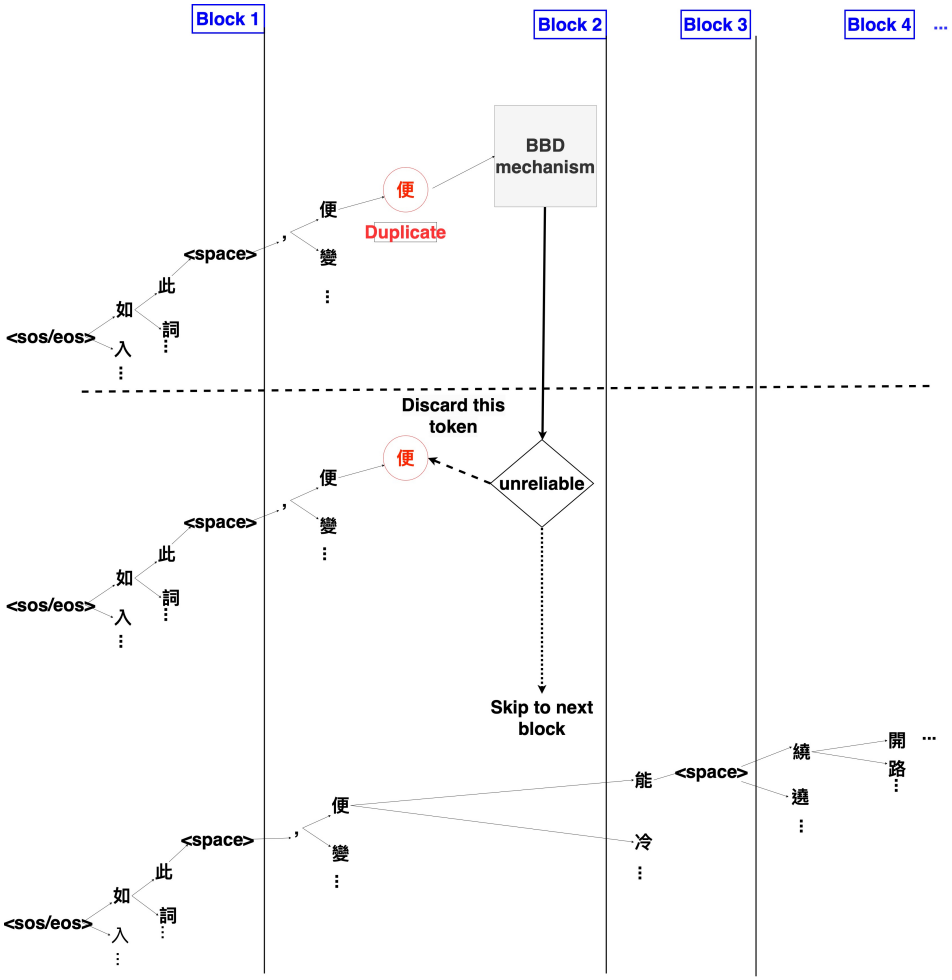

Fig. 2. Example of blockwise synchronous beam search with block boundary detection (BBD) mechanism. During decoding, BBD is used to judge whether the output hypothesis is reliable. In the process, it can be seen the duplicated token is predicted in the second block. Therefore, the token is determined to be unreliable and the decoder will skip to the next block to continue decoding.

rently predicted token and skip to the next block to continue decoding. Example of BBD processing is shown in Fig. 2.

2.4 ASR Training and Decoding

All systems introduced in this paper are based on the architecture of ASR combined with CTC. Greedy Search, which means every step taken must be the best, is used in the CTC decoding. That is, the symbol with the highest output probability in each time step is directly taken as the final result. Secondly, consecutive repeated characters and the blank symbols are removed to obtain the final prediction result after CTC decoding. An

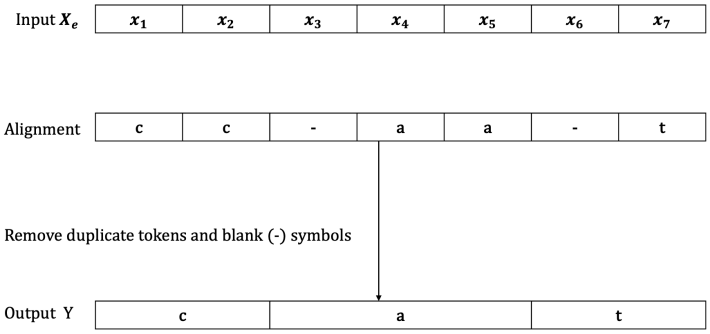

Fig. 3. Example of CTC decoding process; Each input frame will get a token after being predicted by CTC, and finally the continuously repeated token and blank (-) symbol will be removed.

example of the CTC decoding process is shown in Fig. 3.

CTC joint training effectively speeds up the learning and allows the model to converge faster [10, 11]. The CTC module calculate the CTC loss based on the posterior probability of the target output given input as follows,

$$\begin{aligned} C &= \text{softmax}(X_e W_{\text{ctc}} + b_{\text{ctc}}), \\ p(\pi|X_e) &= \prod_{t=1}^{T} C[t, \pi[t]], \\ p_{\text{ctc}}(Y|X_e) &= \sum_{\pi \in \beta^{-1}(Y)} p(\pi|X_e). \end{aligned} \tag{3}$$

where $X_e \in \mathbb{R}^{T \times d_c}$ is the encoder output, in which T is the number of frames and d_c is feature dimension. W_{ctc} and b_{ctc} are matrix and bias for the linear transformation on X_e to be the argument of the softmax function for character probabilities. $C[t, \pi[t]]$ is the probability of output symbol $\pi[t]$ for the t-th frame of X_e. Given X_e, the probability of output path π is the product of the probability of output $\pi[t]$ at frame t. $\beta^{-1}(Y) = \{\pi | Y = \beta(\pi)\}$ represents all output paths for sequence Y. The posterior probability of all possible paths is accumulated for the probability of output sequence Y given X_e. The ASR loss function is combined with negative log probability from the decoder and CTC.

$$L_{\text{asr}} = -\alpha \log p_{\text{e2e}}(Y|X_e) - (1-\alpha) \log p_{\text{ctc}}(Y|X_e) \tag{4}$$

where p_{ctc} is posterior probabilities predicted by the CTC module, and α is a hyperparameter to control the ratio of end-to-end and CTC.

During decoding, we compute the sum of log probabilities from the transformer decoder and CTC in the inference stage. We use the decoding method "Rescoring" [19]. Rescoring uses the two-pass method. The first pass uses beam search to obtain a complete set of hypotheses only in consideration of the sequence probability of the transformer decoder in the beam search process. The second pass rescores the hypotheses using the probabilities of the CTC and the transformer decoder. The CTC probabilities are obtained

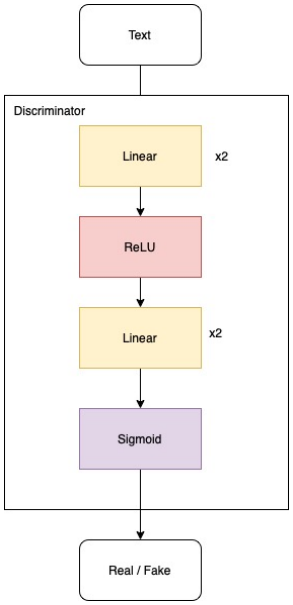

Fig. 4. The architecture of adversarial text discriminator. The adversarial text discriminator is used to determine whether the input text is ground truth or the output recognized by the ASR.

through the CTC forward algorithm. The final result of rescoring is

$$\hat{Y} = \underset{Y \in y^*}{\arg\max} \{\lambda \log p_{\text{e2e}}(Y|X_e) + (1-\lambda) \log p_{\text{ctc}}(Y|X_e)\} \tag{5}$$

where $\lambda \geq 0$ is a hyper-parameter, and y^* is a set of output hypotheses.

2.5 Adversarial Text Discriminator

We use a simple classifier as our adversarial text discriminator, which is shown in Fig. 4, to distinguish whether it is ground truth (natural text) or ASR output. Since the output of the adversarial text discriminator is binary, our loss function is based on the binary cross entropy (BCE) function defined by

$$BCE(O,T) = \frac{1}{n} \sum_{i=1}^{n} (T[i] \cdot \log(O[i]) + (1-T[i]) \cdot \log(1-O[i])) \tag{6}$$

where O and T are the output prediction and the ground-truth label. The generator loss and discriminator loss are defined by

$$Loss_{\text{G}} = BCE(D(Text_{\text{asr}}, 1)) \tag{7}$$

$$Loss_{\text{D}} = 1/2(BCE(D(Text_{\text{asr}}), 0) + BCE(D(Text_{\text{GT}}), 1)) \tag{8}$$

where $Text_{\text{asr}}$ and $Text_{\text{GT}}$ are the discriminator output for ASR output text and ground-truth text, respectively. Note the discriminator output is the probability of ground truth. During training, $Loss_{\text{G}}$ is passed to the ASR model and $Loss_{\text{D}}$ is passed to the adversarial text discriminator.

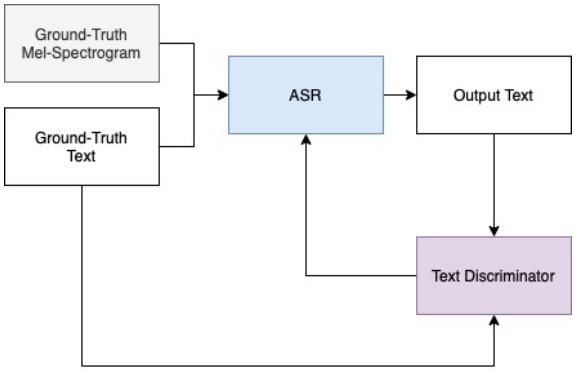

Fig. 5. The architecture of ASR-ADV. In ASR-ADV, we use an adversarial text discriminator, which discriminates whether the input is the output of ASR model or ground-truth text and return the adversarial recognition loss to the ASR model to train.

2.6 ASR-ADV Architecture

ASR model is trained with adversarial loss incurred by adversarial text discriminator, which determines whether the input is ASR output text or ground-truth text. This allows the ASR model to output hypothesis more correctly. The ASR-ADV model architecture is shown in Fig. 5. In the ASR-ADV architecture, the loss function is

$$L_{\text{asr-adv}} = L_{\text{asr}} + Loss_{\text{G}} \tag{9}$$

where L_{asr} is the ASR loss, and is expressed in Eq. (4), and $Loss_G$ is the adversarial recognition loss, which is returned to the ASR model for training, and is expressed in Eq. (7).

2.7 ASR-TTS Architecture

We use a pre-trained TTS module in the overall model for ASR system. In the TTS part, we mark the pronunciation of Mandarin characters with English pinyin and numbers as the input. In order to convert the input Mandarin character sequence to pinyin form, we include a pinyin block for preprocessing.

The ASR-TTS model is shown in Fig. 6. When adding a pre-trained TTS module to the overall model, we freeze the TTS parameters during training. The ASR output is converted to mel-spectrogram by the pre-trained TTS. We use the TTS loss

$$Loss_{\text{tts}} = MAE(TTS(\text{Pinyin}(Text_{\text{asr}})), M_g) \tag{10}$$

where $Text_{\text{asr}}$ is the ASR output, and M_g is the ground-truth mel-spectrogram. That is, the mean absolute error (MAE) between the synthesized mel-spectrogram and the ground-truth mel-spectrogram is back-propagated to modify the learning of parameters.

Combining the TTS loss, the overall loss function for ASR-TTS architecture is

$$L_{\text{asr-tts}} = L_{\text{asr}} + Loss_{\text{tts}} \tag{11}$$

where L_{asr} is the ASR loss, which is expressed as Eq. (4), and $Loss_{\text{tts}}$ is the TTS loss.

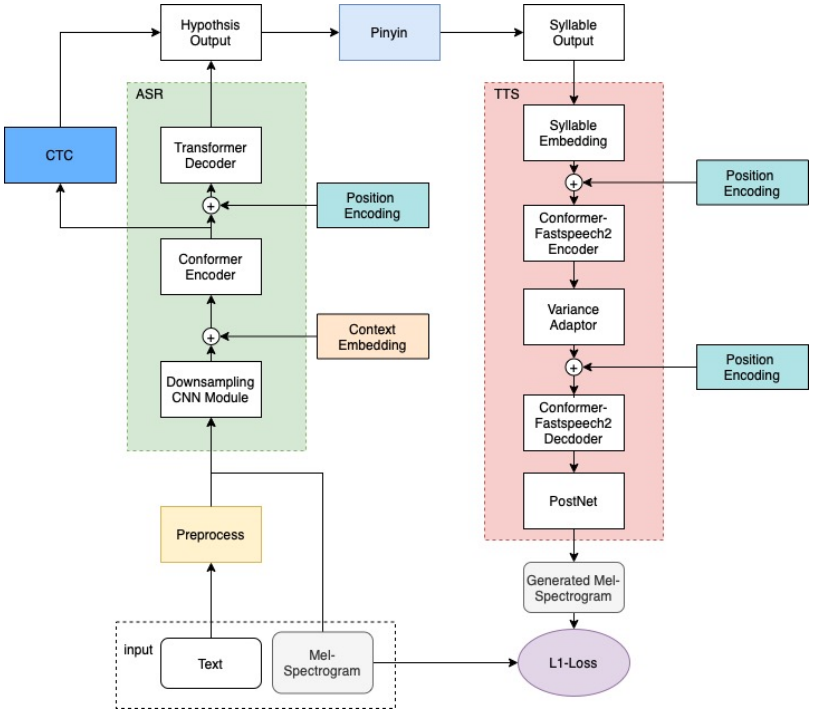

Fig. 6. Block diagram of proposed ASR-TTS system. TTS receives ASR output and synthesizes mel-spectrogram, which is compared with the ground truth mel-spectrogram. Pinyin block, which is just a huge dictionary, converts the output of ASR from Mandarin characters to English Pinyin and numbers to mark the pronunciation.

2.8 ASR-TTS-ADV Architecture

CycleGAN is proposed for domain adaptation through two generators and two discriminators. Let the two domains be Domain X and Domain Y. Generator G_{X2Y} converts image in Domain X to image in Domain Y and G_{Y2X} does the domain conversion in the opposite direction. Discriminators D_X and D_Y check the quality of the images generated by the generators. CycleGAN wants not only that the image y' generated by G_{X2Y} with x to be an image in Domain Y, but also that the image x' generated by G_{Y2X} with y' to have the same content as the input image x. The same is true for the opposite direction. That is

$$G_{X2Y}(x) = y',\ G_{Y2X}(y') = x'' \approx x, \quad \text{and} \quad G_{Y2X}(y) = x',\ G_{X2Y}(x') = y'' \approx y. \tag{12}$$

We integrate ASR-ADV and ASR-TTS with the ideas of CycleGAN. The integrated method is called ASR-TTS-ADV. Seeing speech as domain X and text as domain Y, we treat ASR as G_{X2Y} and TTS as G_{Y2X}. Fig. 7 is the model architecture of ASR-TTS-ADV. In the ASR-TTS-ADV architecture, our loss function consists of three parts

$$L_{\text{asr-tts-adv}} = Loss_{\text{tts}} + Loss_{\text{asr}} + Loss_{\text{G}} \tag{13}$$

where $Loss_{\text{tts}}$, $Loss_{\text{asr}}$ and $Loss_{\text{G}}$ are respectively TTS loss, ASR loss, and adversarial loss. The cycle consistency loss performs only the calculation from domain X to domain Y and

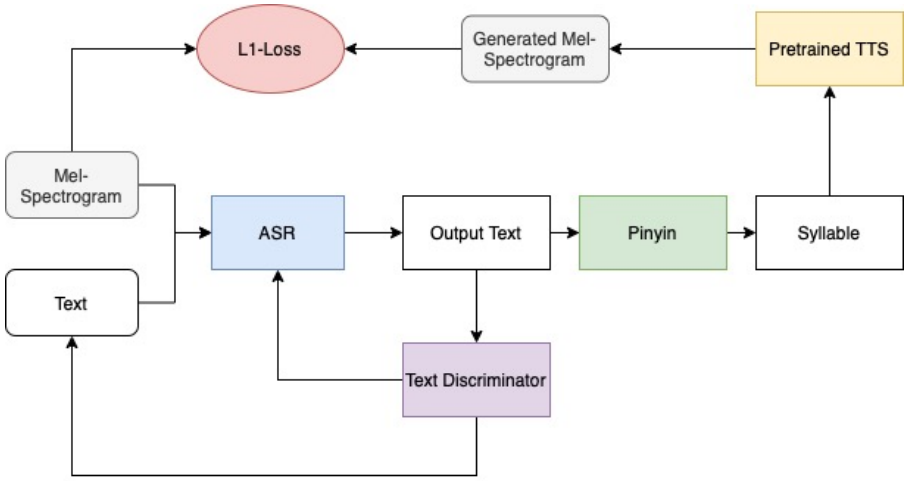

Fig. 7. The architecture of ASR-TTS-ADV. ASR recognizes the text from the input mel-spectrogram, TTS synthesizes the corresponding mel-spectrogram from the output recognized by ASR, and compare the corresponding mel-spectrogram with the groung-truth. The adversarial text discriminator is used to determine whether the input is the output of ASR or the text of ground truth.

Table 1. We divide the Biaobei dataset into training, validation and test sets.

Dataset	Utterances	Hours
Biaobei-train	8,000	9.39
Biaobei-val	1,000	1.22
Biaobei-test	1,000	1.24

back to domain X, which is Eq. (7). The adversarial loss can be calculated by synthesizing the corresponding mel-spectrogram of the ASR output through the pre-trained TTS and calculating the similarity with the ground-truth mel-spectrogram, which is Eq. (10).

3. EXPERIMENTS

We implement the proposed ASR model architectures with the open-source toolkit of ESPnet2 [20]. In this section, we describe the datasets and the parameter settings of ASR, pre-trained TTS and adversarial text discriminator.

3.1 Data

The Biaobei dataset [21] contains 10,000 text-audio pairs for a total duration of 12 hours. It is a Mandarin corpus with a single female speaker. We divide Biaobei data into training, validation, and test sets with 80%, 10%, and 10% ratios, as shown in Table 1. We use a vocabulary of size 4,108 for Biaobei dataset.

In this work, we also apply the proposed ASR-ADV to the Formosa Speech Recognition Challenge 2020 (FSR-2020) Hàn-jī task, a speaker-independent ASR built and

Table 2. The content of the data sets provided by the FSR-2020 challenge. The training sets are TAT-Vol1-train-lavalier and PTS-5.0, and the test set is FSR-2020_final-test.

data set	# of utterances	# of speakers	duration (hours)
TAT-Vol1-train-lavalier	17,516	80	31.44
PTS-5.0	23,363	–	52.71
FSR-2020_final-test	6,991	–	19.57

tested with data from multiple speakers, to improve the end-to-end models we developed for FSR-2020 [22]. FSR-2020 is a competition of speech recognition systems for Taiwanese Hokkien speech. The tasks are divided into three categories according to the symbols for output text, namely Traditional Chinese, Hàn-jī, or Tâi-lô pinyin. FSR-2020 provides TAT-Vol1-train-lavalier training set with Hàn-jī and Tâi-lô pinyin transcription. This dataset is speech data recorded from 80 speakers. It consists of 23,104 pairs of text and audio with total duration of approximately 41.76 hours. FSR-2020 also provides the PTS_TW-train dataset from Taiwan Public Television Service Foundation (PTS). It contains 95 audio files with Hàn-jī transcription. As the audio files in PTS_TW-train tend to be very long, they are automatically segmented, by CTC-Segmentation, to extract short sound clips. The extracted data set is called PTS-5.0, which contains 23,363 pairs of text and audio. FSR-2020_final-test consists of 6,991 pairs of text and audio, with a total duration of approximately 19.57 hours. A summary of the FSR-2020 data sets relevant to this work is given in Table 2. We use a vocabulary of size 4,106 for FSR-2020 Hàn-jī task.

3.2 Experimental Setup

We use speed perturbation [23] and SpecAugment [24] for data augmentation. The speed perturbation [23] changes the speed of the audio signal. We generate three versions of the audio signal with speed factors 0.9, 1.0, and 1.1. SpecAugment [24] performs data augmentation directly on the mel-spectrogram rather than performing on the audio signal. It causes deformation to the mel-spectrogram such as time shift. It also randomly selects a starting point in the frequency or time and mask a segment of mel-spectrgram.

Our end-to-end ASR architecture consists of a 12-layer encoder and a 6-layer of decoder. The kernel size of depth-wise convolution and point-wise convolution in the conformer are 15 and 1. The parameter $\{N_l, N_c, N_r\}$ in the contextual blockwise processing is set to $\{16, 16, 8\}$, overlapping half of the input between adjacent blocks [14, 15]. For the contextual embedding vector, we take the average of all frames in each block as the initial value and use position encoding to distinguish the sequence of the block [14]. The hyperparameter α in ASR loss function is 0.3. The Adam optimizer [25] with square root learning rate scheme is used in training. The hyperparameter λ in decoding is 0.6, and the beam size is 10.

The adversarial text discriminator uses two fully connected layers each with 128 neurons and ReLU for non-linear activation. A final linear combination and sigmoid function is used to obtain the output probability. The parameters are trained with Adam optimizer with square root learning rate scheme. Furthermore, we summarize the system training architectures in this paper in Table 3. Note that TTS is a pre-trained module.

Table 3. Model architectures for training of the proposed systems.

System	Text Discriminator	TTS
ASR	–	–
ASR-ADV	✓	–
ASR-TTS	–	✓
ASR-TTS-ADV	✓	✓

4. RESULTS

First, we present the performance in error rates of the proposed architectures of ASR, ASR-ADV, ASR-TTS and ASR-TTS-ADV on Biaobei dataset. The results show the advantages of the proposed adversarial models. Then, we present the results of proposed methods on the FSR-2020 Hàn-jī task. The proposed systems not only improves over our own non-adversarial models, but also slightly outperforms the participating systems using the hybrid DNN-HMM models, a.k.a. the chain models.

4.1 ASR-ADV

The adversarial text discriminator determines input text as hypotheses or ground truth, and returns an adversarial loss during training. ASR-ADV enhances ASR by text discriminator, so it can improve the problem of spelling errors. A test example is provided in Table 4. In this example, ASR-ADV corrects the original ASR errors. Specifically, the output segment 煮 管部門 in ASR is corrected to 主 管部門 in ASR-ADV. Note the out-of-vocabulary (OOV) rate in our Biaobei-test dataset is 0.40%.

4.2 ASR-TTS

We also propose to use TTS in ASR model architecture, which is ASR-TTS, to reduce the recognition errors caused by the pronunciation and pause problems. The goal of ASR-TTS is for the TTS part to synthesize the same mel-spectrogram as the ground truth. The idea is to train ASR to better output ground-truth pronunciation and segmentation. ASR-TTS produces output with correct pauses, which enable the decoder to decode words with correct semantics. At the same time, it also helps to output correct tonal pronunciation. In Table 4, the outputs of ASR and ASR-ADV are both 籠長的 (lǒng cháng), while the output of ASR-TTS is 冗長的 (rǒng cháng). It can be seen that the addition of pre-trained TTS allows the original output of ASR: lǒng cháng to be correctly identified as rǒng cháng. Not only does this fix the typo in the output, but also makes the semantics correct and avoids the appearance of incomprehensible words. In Table 5 the output of ground truth and ASR-TTS in the second column are both 起火 (qǐ huǒ), which is a falling-rising tone, while the output of ASR identifies incorrect output 齊火 (qí huǒ), which is a rising tone. We observe that the mel-spectrogram synthesized by pre-trained TTS is more capable of correcting the pronunciation errors of similar initials and solving the problem of tone recognition.

Table 4. Comparison of system outputs.

System	Text Output
ground truth	如此，便能繞開政府主管部門冗長的審批。
ASR	如此，便能繞開政府煮管部門籠長的審批。
ASR-ADV	如此，便能繞開政府主管部門籠長的審批。
ASR-TTS	如此，便能繞開政府主管部門冗長的審批。
ASR-TTS-ADV	如此，便能繞開政府主管部門冗長的審批。

Table 5. Comparison of system outputs.

System	Text Output
ground truth	起火點位於河邊。
ASR	齊火點位於河邊。
ASR-ADV	起火點位於河邊。
ASR-TTS	起火點位於河邊。
ASR-TTS-ADV	起火點位於河邊。

Table 6. Comparison of four systems of character and word error rate.

System	CER(%)	WER(%)
ASR	12.6	31.7
ASR-ADV	12.3	31.4
ASR-TTS	10.8	24.4
ASR-TTS-ADV	9.9	22.7

4.3 ASR-TTS-ADV

ASR-ADV system initially improves the problem of typos from the ASR system. In the ASR-TTS system, not only pre-trained TTS can solve the pronunciation problem, but also the ASR system can have better word segmentation performance. In the end, we integrate the pre-trained TTS and adversarial text discriminator into the ASR system and turn it into an ASR-TTS-ADV system, which retains the advantages of ASR-TTS and ASR-ADV, such as the ability to solve the problems of pronunciation, word segmentation, and spelling errors. While Tables 4 and 5 show the outputs of ASR-TTS-ADV and ASR-TTS are the same as ground truth, Table 6 shows consistent decreases in CER and WER as the system continues to improve.

4.4 FSR-2020 Hàn-jī Task

We also apply the proposed method on FSR-2020 Hàn-jī task. The performance is presented in Fig. 8. The blue bars are the official results of the participating teams on the final test dataset. The red bar is the performance of the proposed ASR-ADV on the same dataset. Achieving a character error rate of 35.8%, the proposed system has outperformed our system (P22) without adversarial training [22]. Furthermore, the improved CER is slightly better than the first-place system (T21 [26]) based on hybrid DNN-HMM chain model using a Kaldi recipe at that time. Note the OOV rate in this task is 0 as we used characters as units for ASR output tokens.

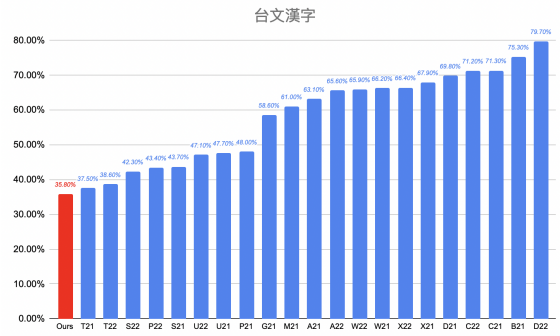

Fig. 8. The ASR-ADV system is compared with the character error rates of participating teams of FSR-2020 Hàn-jī task.

5. CONCLUSION

We apply ideas inspired by CycleGAN to improve ASR system. We introduce the adversarial text discriminator and TTS to incur adversarial loss and consistency loss to guide the training of ASR model parameters. Experiments of the proposed methods on Mandarin and Taiwanese speech recognition show significant improvements on error rates.

In the future, we want to explore variants of GANs in order to improve the ASR-TTS-ADV system. Our next step might be to try out dual training [27] to make ASR and TTS systems compete against each other during training to make both modules better.

REFERENCES

1. L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, Vol. 77, 1989, pp. 257-286.
2. D. Povey *et al.*, "The subspace Gaussian mixture model – A structured model for speech recognition," *Computer Speech & Language*, Vol. 25, 2011, pp. 404-439.
3. A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369-376.
4. W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *arXiv Preprint*, 2015, arXiv:1508.01211.
5. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, Vol. 30, 2017.
6. A. Gulati *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *arXiv Preprint*, 2020, arXiv:2005.08100.
7. Y. Peng, S. Dalmia, I. Lane, and S. Watanabe, "Branchformer: Parallel MLP-attention architectures to capture local and global context for speech recognition and understanding," in *Proceedings of International Conference on Machine Learning*, 2022, pp. 17 627-17 643.

8. A. Graves, "Sequence transduction with recurrent neural networks," *arXiv Preprint*, 2012, arXiv:1211.3711.
9. S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid CTC/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, Vol. 11, 2017, pp. 1240-1253.
10. S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017, pp. 4835-4839.
11. T. Nakatani, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *Proceedings of Interspeech*, 2019.
12. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, Vol. 27, 2014.
13. J.-Y. Zhu, T. Park, P. Isola, and A. 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.
14. E. Tsunoo, Y. Kashiwagi, T. Kumakura, and S. Watanabe, "Transformer ASR with contextual block processing," in *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop*, 2019, pp. 427-433.
15. E. Tsunoo, Y. Kashiwagi, and S. Watanabe, "Streaming transformer ASR with blockwise synchronous beam search," in *Proceedings of IEEE Spoken Language Technology Workshop*, 2021, pp. 22-29.
16. M. K. Baskar *et al.*, "EAT: Enhanced ASR-TTS for self-supervised speech recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2021, pp. 6753-6757.
17. J. Sun, "Jieba Chinese word segmentation tool," https://github.com/fxsjy/jieba, 2012.
18. Y. Lu, Z. Li, D. He, Z. Sun, B. Dong, T. Qin, L. Wang, and T.-Y. Liu, "Understanding and improving transformer from a multi-particle dynamic system point of view," *arXiv Preprint*, 2019, arXiv:1906.02762.
19. T. Hori, S. Watanabe, and J. R. Hershey, "Joint CTC/attention decoding for end-to-end speech recognition," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, Vol. 1, 2017, pp. 518-529.
20. S. Watanabe *et al.*, "ESPNet: End-to-end speech processing toolkit," *arXiv Preprint*, 2018, arXiv:1804.00015.
21. BiaoBei (Beijing) Technology, "Chinese standard Mandarin speech corpus," https://www.data-baker.com/open source.html, 2017.
22. H.-P. Lin, "Improving speech recognition systems for low-resource languages with hidden speaker information," https://hdl.handle.net/11296/p2yng3, 2021.
23. T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proceedings of Interspeech*, 2015, pp. 3586-3589.
24. D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *arXiv Preprint*, 2019, arXiv:1904.08779.
25. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv Preprint*, 2014, arXiv:1412.6980.

26. H.-B. Liang, C.-Y. Li, and H.-Y. Lee, "The NTU ASR system for formosa speech recognition challenge 2020," in *Proceedings of Speech Signal Processing Workshop*, 2021.
27. Y. Ren, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Almost unsupervised text to speech and automatic speech recognition," in *Proceedings of International Conference on Machine Learning*, 2019, pp. 5410-5419.

Hou-An Lin obtained the BS degree in Applied Mathematics from Feng Chia University and earned the MS degree in Computer Science and Engineering from National Sun Yat-sen University in 2020 and 2023. His research focuses on speech recognition, including code-switching speech recognition systems and training speech recognition systems using speech synthesis and text 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 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.