Li *et al. EURASIP Journal on Audio, Speech, and Music Processing* (2021) 2021:26
https://doi.org/10.1186/s13636-021-00215-6

EURASIP Journal on Audio, Speech, and Music Processing

RESEARCH **Open Access**

Adversarial joint training with self-attention mechanism for robust end-to-end speech recognition

Lujun Li*, Yikai Kang†, Yuchen Shi†, Ludwig Kürzinger, Tobias Watzel and Gerhard Rigoll

Abstract
Lately, the self-attention mechanism has marked a new milestone in the field of automatic speech recognition (ASR). Nevertheless, its performance is susceptible to environmental intrusions as the system predicts the next output symbol depending on the full input sequence and the previous predictions. A popular solution for this problem is adding an independent speech enhancement module as the front-end. Nonetheless, due to being trained separately from the ASR module, the independent enhancement front-end falls into the sub-optimum easily. Besides, the handcrafted loss function of the enhancement module tends to introduce unseen distortions, which even degrade the ASR performance. Inspired by the extensive applications of the generative adversarial networks (GANs) in speech enhancement and ASR tasks, we propose an adversarial joint training framework with the self-attention mechanism to boost the noise robustness of the ASR system. Generally, it consists of a self-attention speech enhancement GAN and a self-attention end-to-end ASR model. There are two advantages which are worth noting in this proposed framework. One is that it benefits from the advancement of both self-attention mechanism and GANs, while the other is that the discriminator of GAN plays the role of the global discriminant network in the stage of the adversarial joint training, which guides the enhancement front-end to capture more compatible structures for the subsequent ASR module and thereby offsets the limitation of the separate training and handcrafted loss functions. With the adversarial joint optimization, the proposed framework is expected to learn more robust representations suitable for the ASR task. We execute systematic experiments on the corpus AISHELL-1, and the experimental results show that on the artificial noisy test set, the proposed framework achieves the relative improvements of 66% compared to the ASR model trained by clean data solely, 35.1% compared to the speech enhancement and ASR scheme without joint training, and 5.3% compared to multi-condition training.

Keywords: Self-attention mechanism, Generative adversarial networks, Speech enhancement, Robust speech recognition

1 Introduction

In recent years, attention-based end-to-end neural networks, which subsume the acoustic and language models into a single neural network, have triggered a revolution in the field of automatic speech recognition (ASR) [1, 2] and are challenging the dominance of hidden Markov model-based hybrid systems [3]. Furthermore, the self-attention mechanism has made another breakthrough in the innovation of the attention architecture, which considers the whole sequence at once to model feature interactions that are arbitrarily distant in time, leading to faster convergence and state-of-the-art results in ASR [4–12]. The self-attention system predicts the next output symbol conditioned on the full sequence of the previous predictions. Once a mistake occurs in one estimation step due to noise interference, all the subsequent steps will be

*Correspondence: lujun.li@tum.de
†Yikai Kang and Yuchen Shi contributed equally to this work.
Department of Electrical and Computer Engineering, Technical University of Munich, Munich, Germany

disturbed. As speech signals are inevitably interfered by various background noises in the realistic environment, it is crucial to improve the robustness of the self-attention mechanism for practical application.

The mainstream solution to the noise robustness problem is adding an independent speech enhancement (SE) module as the front-end of ASR. Speech enhancement aims to transform the interfered speech to its original clean version, which is achieved by various approaches, i.e., statistical methods such as Wiener filter [13], time-frequency masking [14–16], signal approximation [17, 18], and spectral mapping [19, 20]. No matter what approach the speech enhancement model adopts to achieve the goal, it is trained separately from the ASR model on different loss functions (i.e., mean squared error [21]) and being evaluated by different objective criteria (i.e., mean opinion score (MOS) prediction of the intrusiveness of background noise [22], segmental SNR [23]). This mismatch between the enhancement training and the final ASR task leads to a sub-optimum easily [24]. Moreover, the handcrafted loss functions tend to generate over-smoothed spectra or introduce unseen distortions, which sometimes even degrade the downstream ASR performance [25].

To obtain the optimum and circumvent introducing unnecessary distortion, the idea of a joint training framework is proposed for robust speech recognition [25–28]. The fundamental concept of the joint training is concatenating the speech enhancement front-end and a downstream ASR model to build an entire neural network and jointly adjust the parameters in each module. The goal here is that the enhancement front-end tends to produce enhanced features desired by the ASR component, and the ASR module can guide the enhancement module to a more discriminative direction. In this way, the joint framework is optimized on the final ASR objectives, i.e., word/character error rate (W/CER).

Generative adversarial networks (GANs) aim at mapping samples $\hat{x}$ from the distribution $\hat{\mathcal{X}}$ to samples x from another distribution $\mathcal{X}$. There are two components within GANs. One is the generator (G), which performs the mapping, and the other is the discriminator (D), which guides the training of the generator. GANs have been applied to various speech signal processing tasks, such as speech enhancement [29, 30], robust speaker verification [31], spoken language identification [32], speech emotion recognition [33], data augmentation [34], and robust speech recognition [35].

Inspired by the advancement of self-attention mechanism and various applications of GAN in speech-related tasks, we propose an adversarial joint training framework with self-attention mechanism to boost the robustness of the self-attention ASR systems, which consists of a self-attention speech enhancement GAN (SA_SEGAN) and a self-attention end-to-end ASR model (SA_ASR), where we experiment with Transformer [36] and Conformer [37]. The discriminant component of SA_SEGAN is first utilized to distinguish the enhanced features from the original clean features, instructing the enhancement module to output the clean distribution. When it comes to the stage of the joint training, the D component acts as the global training guide, and it will shift the direction for the G component to produce more congruous features for the ASR task. As the global guide, the discriminator is expected to remedy the limitation of the separate training and handcrafted loss functions, alleviate the distortion, and lead the speech enhancement component to the global optimum. Meanwhile, the enhancement module is supposed to capture more underlying structural characteristics. With this global guide, the whole framework is expected to learn more robust representations compatible with the ASR task automatically.

In summary, the main contributions of this paper are as follows:

- We propose a self-attention-based jointly trained adversarial framework targeting robust speech recognition. To the best of our knowledge, this is the first joint training scheme that benefits from the advantages of both the self-attention mechanism and adversarial training.
- We conduct the local and global adversarial training simultaneously, where the discriminant component does not concentrate on the enhancement front-end exclusively, but also plays the role of the global training guide.
- The proposed framework yields remarkable results, which achieves relative improvements up to 66% compared to the ASR model trained by clean data solely, 35.1% compared to the scheme without joint training, and 5.3% compared to multi-condition training.

2 Related work

GANs have been applied in speech enhancement tasks without attention [29, 38, 39] and with attention [40, 41]. These works validate the functionality of GAN in the enhancement task on diverse objective criteria; however, they lack proofs of the effectiveness of their work for the downstream ASR task.

GANs have also been employed to improve the robustness of the ASR model [35, 42–44]. A potential limitation lies in the weak matching and communication between the integrated modules. For instance, speech enhancement and speech recognition are often designed independently, and the enhancement system is tuned according to the metrics that are not straightly relative to the final ASR performance.

To address this concern, joint training is a promising approach. An early attempt was proposed in [45], where a feature extraction front-end and a Gaussian mixture model-hidden Markov model back-end are jointly trained on maximum mutual information. Afterwards, other interesting works are published in this field [25, 26, 46–48]. Nevertheless, an effective integration between the various systems has been difficult for many years, mainly due to the different nature of the technologies involved at different steps. For example, in [25, 46], the joint training is actually performed as a fine-tuning procedure. To tackle this problem, this paper deploys the discriminant component of GAN as a global guide, leading the enhancement module to match the downstream ASR module.

3 Self-attention-based SE-ASR scheme

3.1 Overview

Figure 1 illustrates an overview of our proposed joint training framework for robust end-to-end speech recognition pictorially. The system consists of a self-attention enhancement front-end and a self-attention ASR model. Given the raw noisy speech input $\tilde{\boldsymbol{x}}$ and the raw clean input $\boldsymbol{x}^{\boldsymbol{*}}$, we illustrate the entire procedure of the joint training pipeline in the following forms:

$$\hat{\boldsymbol{x}} = \text{Generator}(\tilde{\boldsymbol{x}}), \tag{1}$$

$$\hat{\boldsymbol{f}} = \text{FBank}(\hat{\boldsymbol{x}}), \tag{2}$$

$$P(\boldsymbol{y}|\hat{\boldsymbol{f}}) = \text{SA\_ASR}(\hat{\boldsymbol{f}}), \tag{3}$$

$$P(D|\hat{\boldsymbol{x}}, \boldsymbol{x}^{\boldsymbol{*}}) = \text{Discriminator}(\hat{\boldsymbol{x}}, \boldsymbol{x}^{\boldsymbol{*}}). \tag{4}$$

Here, Generator$(\cdot)$ acts as a speech enhancement front-end realized by the generator component of SA_SEGAN [40], which transforms the noisy raw input $\tilde{\boldsymbol{x}}$ to the enhanced $\hat{\boldsymbol{x}}$. FBank$(\cdot)$ is a function for extracting the normalized log FBank features $\hat{\boldsymbol{f}}$ from the enhancement outputs $\hat{\boldsymbol{x}}$. Subsequently, SA_ASR$(\cdot)$ is an ASR system based on self-attention layers realized by the Transformer [36] or Conformer [37] architecture. $\boldsymbol{y}$ is the outputs of the whole scheme. Discriminator$(\cdot)$ is realized by the discriminator component of SA_SEGAN [40], which distinguishes enhanced outputs from clean data.

3.2 Self-attention mechanism

Self-attention [49] relates the information over different positions of the entire input sequence for computing the attention distribution using scaled dot-product attention:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{QK}^T}{\sqrt{d_k}}\right)\boldsymbol{V}. \tag{5}$$

$\boldsymbol{Q} \in \mathbb{R}^{t_q \times d_q}$, $\boldsymbol{K} \in \mathbb{R}^{t_k \times d_k}$, and $\boldsymbol{V} \in \mathbb{R}^{t_v \times d_v}$ are three inputs of the self-attention layer: queries, keys, and values, where t_q, t_k, and t_v are the element numbers in different inputs while d_q, d_k, and d_v denote the corresponding element dimensions. The scalar $\frac{1}{\sqrt{d_k}}$ prevents the softmax function from falling into regions with tiny gradients. One query's output is computed as a weighted sum of the values, where each weight of the value is computed by a designated function of the query with the corresponding key.

3.3 Self-attention speech enhancement GANs

3.3.1 Speech enhancement GANs (SEGAN)

Given a dataset $\mathcal{X} = \{(\boldsymbol{x}_1^*, \tilde{\boldsymbol{x}}_1), (\boldsymbol{x}_2^*, \tilde{\boldsymbol{x}}_2), \cdots, (\boldsymbol{x}_N^*, \tilde{\boldsymbol{x}}_N)\}$ consisting of N pairs of raw signals: clean speech signal $\boldsymbol{x}^*$ and noisy speech signal $\tilde{\boldsymbol{x}}$. Speech enhancement aims to find a mapping $f_\theta(\tilde{\boldsymbol{x}}) : \tilde{\boldsymbol{x}} \rightarrow \hat{\boldsymbol{x}}$ to transform the raw noisy signal $\tilde{\boldsymbol{x}}$ to the enhanced signal $\hat{\boldsymbol{x}}$. θ contains the parameters of the enhancement network.

Conforming to GAN's principle [50], the generator G is for learning an effective mapping that can imitate the real data distribution to generate novel samples related to those of the training set. Hence, G acts as the enhancement function. In contrast, the discriminator D plays the role of a classifier which distinguishes the real sample, coming from the dataset that G is imitating, from the fake samples, made up by G. D guides θ towards the distribution of clean speech signals. To sum up, SEGAN designates the generator G for the enhancement mapping, i.e., $\hat{\boldsymbol{x}} = G(\tilde{\boldsymbol{x}})$, while designates the discriminator D to guide the training of G by classifying $(\boldsymbol{x}^{\boldsymbol{*}}, \tilde{\boldsymbol{x}})$ as real and $(\hat{\boldsymbol{x}}, \tilde{\boldsymbol{x}})$ as fake. Eventually, G learns to produce enhanced signals $\hat{\boldsymbol{x}}$ good enough to fool D such that D classifies $(\hat{\boldsymbol{x}}, \tilde{\boldsymbol{x}})$ as real.

3.3.2 Self-attention speech enhancement GANs (SA_SEGAN)

SA_SEGAN [40] is SEGAN with the adoption of the self-attention layer adapted from non-local attention [51, 52]. Given the feature map $\boldsymbol{F} \in \mathbb{R}^{L \times C}$ output by the 1-dim convolutional layer, where L is the time dimension, C is the number of channels, the query matrix $\boldsymbol{Q}$, the key matrix $\boldsymbol{K}$, and the value matrix $\boldsymbol{V}$ are obtained via transformations:

$$\boldsymbol{Q} = \boldsymbol{FW}^Q, \boldsymbol{K} = \boldsymbol{FW}^K, \boldsymbol{V} = \boldsymbol{FW}^V, \tag{6}$$

where $\boldsymbol{W}^Q \in \mathbb{R}^{C \times \frac{C}{b}}$, $\boldsymbol{W}^K \in \mathbb{R}^{C \times \frac{C}{b}}$, and $\boldsymbol{W}^V \in \mathbb{R}^{C \times \frac{C}{b}}$ represent the learnt weight matrices of the 1×1 convolutional layer of $\frac{C}{b}$ filters. Furthermore, Phan et al. [40] introduce two factors, b and p, for memory efficiency. b reduces the channel dimension, while p reduces the number of keys and values by a max pooling layer with filter width and stride size of p. Therefore, the dimension of the matrices

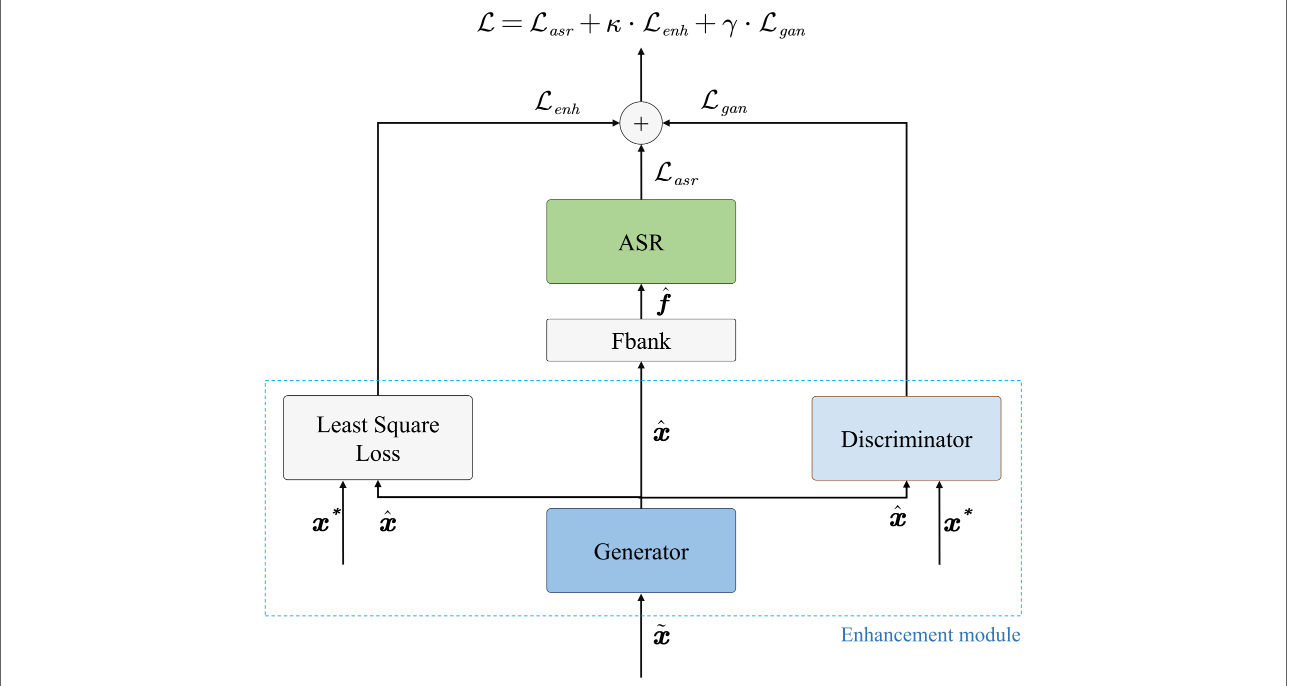

Fig. 1 Overview of the SE_ASR joint training framework

are $Q \in \mathbb{R}^{L\times\frac{C}{b}}$, $K \in \mathbb{R}^{\frac{L}{p}\times\frac{C}{b}}$, and $V \in \mathbb{R}^{\frac{L}{p}\times\frac{C}{b}}$. The attention map A and the attentive output O are then computed as:

$$A = softmax(QK^T), \quad A \in \mathbb{R}^{L\times\frac{L}{p}}, \tag{7}$$

$$O = (AV)W^O, \quad W^O \in \mathbb{R}^{\frac{C}{b}\times C}. \tag{8}$$

Each element $a_{ij} \in A$ indicates the extent to which the model attends to the jth column v_j of V when producing the ith output o_i of O. With the weight matrix W^O realized by a 1 $\times$1 convolution layer of C filters, the shape of O is restored to the original shape $L \times C$.

In the end, SA_SEGAN contains a shortcut connection to facilitate information propagation, and a learnable parameter β is employed to balance the weight between the output O and the input feature map F as:

$$F' = \beta O + F. \tag{9}$$

We illustrate the diagram of a simplified self-attention layer with $L = 9$, $C = 6$, $p = 3$, and $b = 2$ in Fig. 2.

3.3.3 Network architecture

The architectures of the generator G and the discriminator D are depicted in Fig. 3a, b. The G component makes use of an encoder-decoder architecture with fully convolutional layers [53]. The generator's encoder comprises 11 1-dim stridden convolutional layers with a common filter width of 31 and a stride length of 2, followed by parametric rectified linear units (PReLUs) [54]. The encoder receives a 1-s segment of the raw signal sampled at 16 kHz, approximately 16,384 samples as the input. To compensate for the smaller and smaller convolutional output, the number of filters increases along the encoder's depth $\{16, 32, 32, 64, 64, 128, 128, 256, 256, 512, 1024\}$, resulting in output size of the feature map $\{8192 \times 16, 4096 \times 32, 2048 \times 32, 1024 \times 64, 512 \times 64, 256 \times 128, 128 \times 128, 64 \times 256, 32 \times 256, 16 \times 512, 8 \times 1024\}$. At the 11th layer of the encoder, the encoding vector $c \in \mathbb{R}^{8\times1024}$ is stacked with the noise sample $z \in \mathbb{R}^{8\times1024}$, sampled from the distribution $\mathcal{N}(0, I)$, and presented to the decoder.

The decoder component mirrors the encoder architecture with the same number of filters and the filter width to reverse the encoding process through deconvolutions. The same as the encoder, each deconvolutional layer is again followed by a PReLUs. The skip connections are deployed to connect the encoding layer with its corresponding decoding layer to allow the information flow between the encoding stage and the decoding stage.

The discriminator is constructed of a similar architecture to the encoder component of the generator. However, it receives the two-channel input and utilize virtual batch-norm [55] before LeakyReLU [56] activation with α = 0.3. Moreover, the D network is topped up with a 1 $\times$ 1 convolutional layer to reduce the dimension of the output of the last convolutional layer from 8 $\times$ 1024 to 8 for the subsequent classification task with the softmax layer.

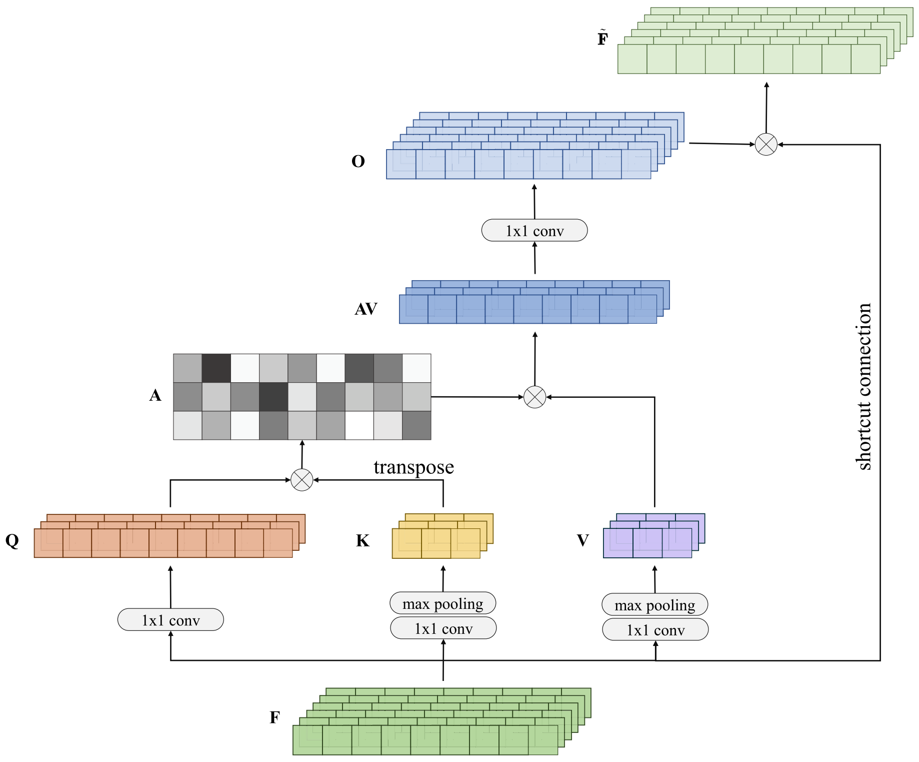

Fig. 2 Illustration of the application of self-attention mechanism in speech enhancement GANs with $L = 9$, $C = 6$, $p = 3$, and $b = 2$

The self-attention layer illustrated in Section 3.3.2 couples with the (de)convolutional layer of both the generator and the discriminator. Figure 3a, b demonstrate an example of the self-attention layer coupling with the lth (de)convolutional layer. As we can see, if we add the self-attention layer to the lth convolutional layer of the encoder, the mirror lth deconvolutional layer of the decoder and the lth layer in the discriminator also couples a self-attention layer. Theoretically, the self-attention layer can be placed in any number, even all, of the (de)convolutional layers.

3.4 FBank extraction network

We extract the normalized log FBank features $\hat{\boldsymbol{f}}$ as the input of the subsequent ASR model, which is computed from the enhanced signals $\hat{\boldsymbol{x}}$:

$$\hat{\boldsymbol{f}} = \text{FBank}(\hat{\boldsymbol{x}}) = \text{Norm}(\log(\text{Mel}(\text{STFT}(\hat{\boldsymbol{x}})))), \qquad (10)$$

where STFT$(\cdot)$ is the operation of short-time Fourier transform (STFT), Mel$(\cdot)$ is the operation of Mel matrix multiplication, and Norm$(\cdot)$ is for normalizing the mean and variance to 0 and 1, separately. Consequently, the FBank feature extraction layer is differentiable.

3.5 Transformer

3.5.1 Multi-head attention mechanism

Multi-head attention mechanism [49], as the terminology implies, contains more than one self-attention module. As the core module of the Transformer [36], it leverages different attending representations jointly. Before performing each attention, three linear projections transform the queries, keys, and values to more discriminated representations, respectively. Afterwards, each dot-product attention is calculated independently, and their outputs are concatenated and fed into another linear projection to obtain the final d_{model}-dimensional outputs:

$$\begin{aligned}&\text{MultiHead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})\\ &\quad = \text{Concat}(head_1, head_2, \cdots, head_h)\boldsymbol{W}^{OUT}, \end{aligned} \qquad (11)$$

where

$$head_i = \text{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V). \qquad (12)$$

h refers to the head numbers, and $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}$ have the same dimensions of d_{model}. Four projection matrices $\boldsymbol{W}_i^Q \in \mathbb{R}^{d_{model}\times d_q}$, $\boldsymbol{W}_i^K \in \mathbb{R}^{d_{model}\times d_k}$, $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{model}\times d_v}$, and $\boldsymbol{W}^{OUT} \in \mathbb{R}^{hd_v\times d_{model}}$. Additionally, $d_q = d_k = d_v = d_{model}/h$.

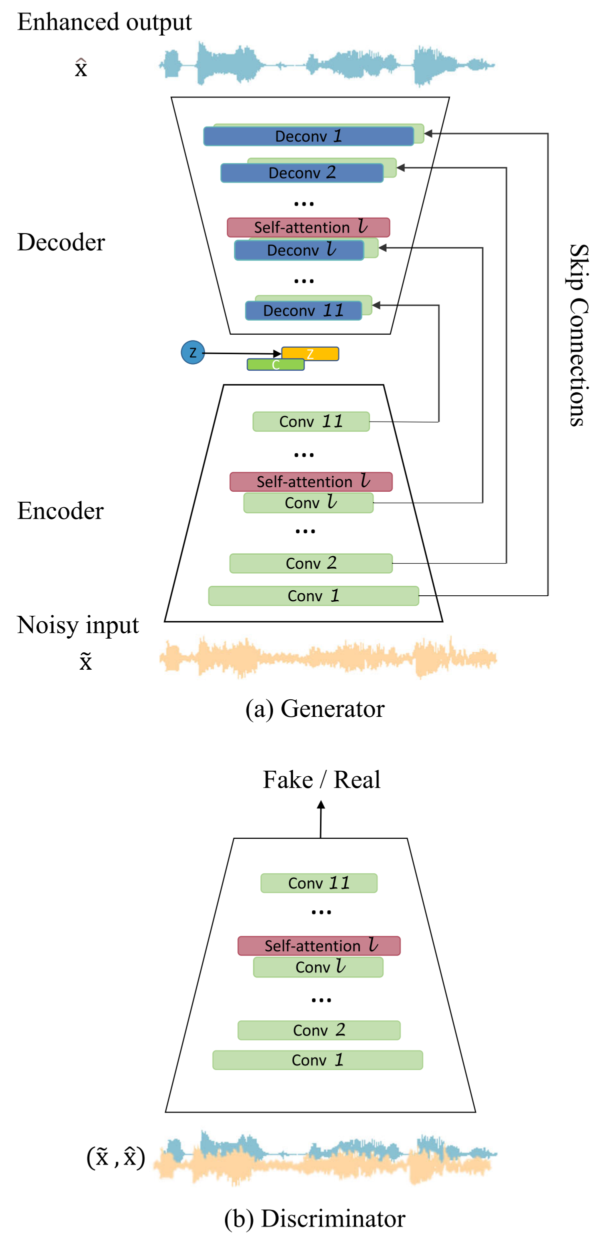

Fig. 3 Illustration of the SA_SEGAN architecture. **a** The generator component. **b** The discriminator component [40]

3.5.2 *Positional encoding*

One obvious limitation of the Transformer model is that the output is invariant to the input order permutation, i.e., the Transformer does not model the order of the input sequence. Vaswani et al. [49] solve this problem by injecting information about absolute positions into the input sequence via sinusoid positional embeddings:

$$PE_{(pos,i)} = \begin{cases} sin(pos/10000^{i/d_{model}}) & \text{if } i \text{ is even} \\ cos(pos/10000^{i/d_{model}}) & \text{if } i \text{ is odd} \end{cases}, \tag{13}$$

where *pos* refers to the position, and i is the dimension. The sinusoidal function allows the model to extrapolate from long sequence lengths.

3.5.3 *Feed-forward network*

The feed-forward network (FFN) is another core module of the Transformer [36]. It is composed of two linear transformations with a ReLU activation in between. The dimensionality of the input and output is d_{model}, and the inner layer has the dimensionality d_{ff}. Specifically,

$$\text{FFN}(\boldsymbol{x}) = \max(0, \boldsymbol{x}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2, \tag{14}$$

where the weights $\boldsymbol{W}_1 \in \mathbb{R}^{d_{model} \times d_{ff}}$, $\boldsymbol{W}_2 \in \mathbb{R}^{d_{ff} \times d_{model}}$ and the biases $\boldsymbol{b}_1 \in \mathbb{R}^{d_{ff}}$, $\boldsymbol{b}_2 \in \mathbb{R}^{d_{model}}$. The linear transformations are the same across different positions.

3.5.4 *Network architecture*

The detailed model architecture of the ASR-Transformer is as follows.

The encoder is shown in Fig. 4a. The input embedding is for extracting expressive representations of dimension d_{model}. Thereafter, to enable the model to attend on the auxiliary position information, the d_{model}-dim positional encoding (Section 3.5.2) is added to the input encoding. Then, the sum of the encoded outputs is fed into a stack of N_e encoder blocks, each of which has two sub-blocks: one is the multi-head attention (Section 3.5.1), receiving queries, keys, and values from the previous block, and the other is the feed-forward networks (Section 3.5.3). In the meanwhile, layer normalization and residual connection are introduced to each sub-block for effective training. Thus, the pipeline of the sub-block is:

$$\boldsymbol{x} + \text{SubBlock}(\text{Layer Norm}(\boldsymbol{x})). \tag{15}$$

The decoder is shown in Fig. 4b. The output-embedding converts the character sequence to dimension d_{model}. Added with the positional encoding, the sum of them is fed into a stack of N_d decoder blocks, which consists of three sub-blocks: The first is a masked multi-head attention, which ensures that the predictions for position j depends only on the known outputs at positions less than j. The second is a multi-head attention whose keys and values come from the encoder outputs while queries come from the previous sub-block outputs. The third is also feed-forward networks. Similar to the encoder, layer normalization and residual connection are also employed to each sub-block of the decoder. Eventually, the output probabilities are acquired by a linear projection and a subsequent softmax function.

3.6 Conformer

Conformer [37] is a state-of-the-art ASR encoder architecture. Different from the Transformer block (as

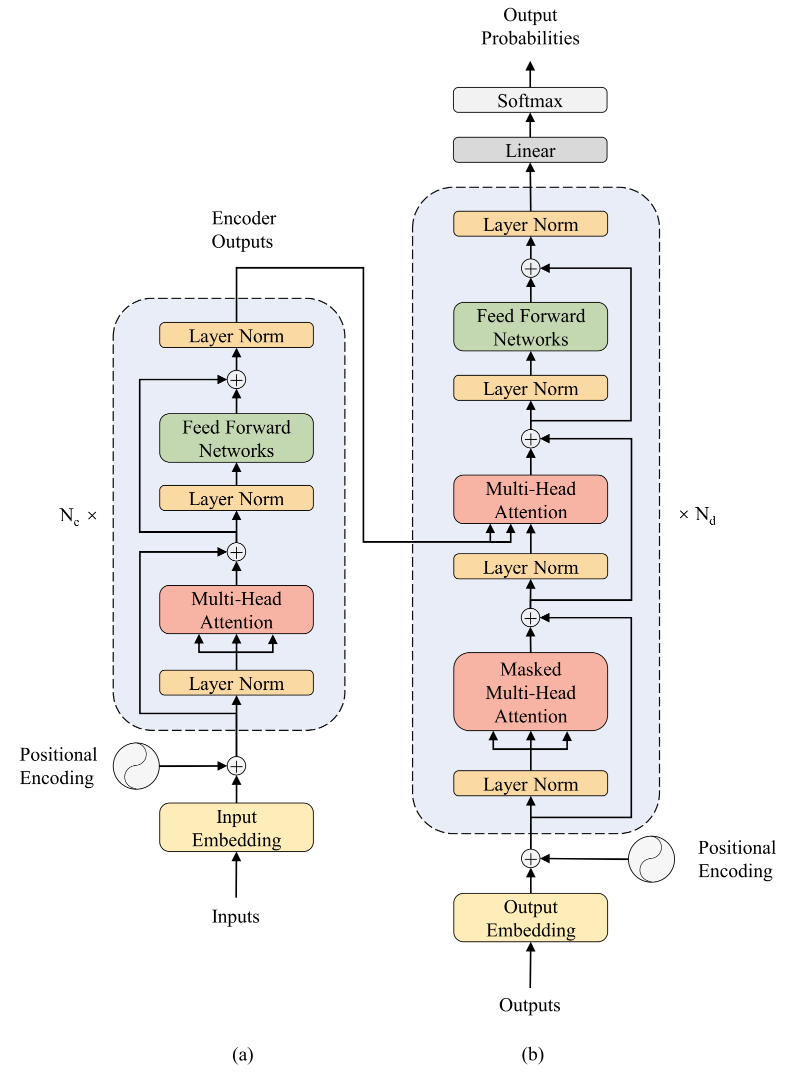

Fig. 4 Model architecture of the Transformer. **a** Encoder. **b** Decoder [36]

described in Section 3.5), it is equipped with a convolution layer to increase the local information modeling capability of the Transformer encoder model [49] and a pair of FFN modules sandwiching the multi-head self-attention module and the integrated convolution module. The Conformer model consists of a Conformer encoder proposed in [37] and a Transformer decoder [36]. The encoder first processes the input with a convolution subsampling layer and then with Conformer blocks, as illustrated in Fig. 5i. The Conformer block (Fig. 5ii) consists of a multi-head self-attention module (MHSA), a convolution module, sandwiched by a pair of macron-feedforward module [57]. The layer normalization is applied before each module, and the dropout is followed by a residual connection afterwards (pre-norm) [58, 59]. Mathematically, let $\boldsymbol{x}_i$ be the input to the ith Conformer block, the output $\boldsymbol{y}_i$ of this block is:

$$\boldsymbol{x}'_i = \boldsymbol{x}_i + \frac{1}{2}\mathrm{FFN}(\boldsymbol{x}_i), \tag{16}$$

$$\boldsymbol{x}''_i = \boldsymbol{x}'_i + \mathrm{MHSA}(\boldsymbol{x}'_i), \tag{17}$$

$$\boldsymbol{x}'''_i = \boldsymbol{x}''_i + \mathrm{Conv}(\boldsymbol{x}''_i), \tag{18}$$

$$\boldsymbol{y}_i = \mathrm{Layer\,Norm}\left(\boldsymbol{x}'''_i + \frac{1}{2}\mathrm{FFN}(\boldsymbol{x}'''_i)\right). \tag{19}$$

FFN(·), MHSA(·), Conv(·), and Layer Norm(·) denote the macron-feedforward module, the multi-head self-attention module, the convolution module, and the layer normalization module, respectively. The multi-head self-attention module is the same as in Section 3.5.1 and is demonstrated in Fig. 5ii-b. Sections 3.6.1 and 3.6.2 introduce the convolution module and the macron-feedforward module, respectively.

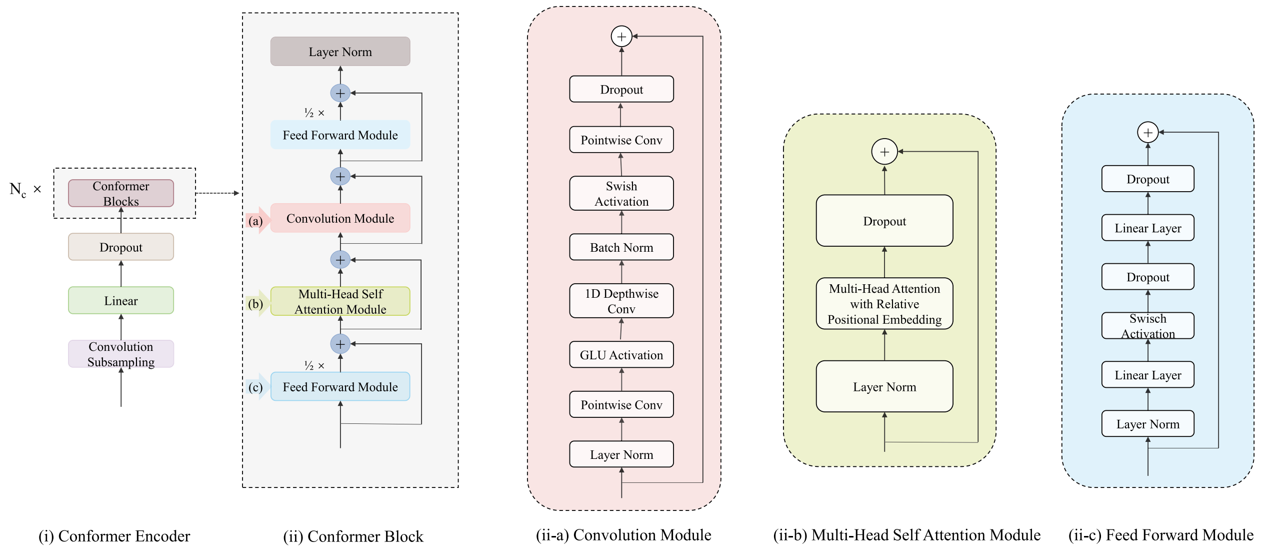

Fig. 5 Illustration of the Conformer encoder model architecture. **i** Conformer encoder architecture. **ii** Conformer block architecture. **ii-a** Convolution module of the Conformer block. **ii-b** Multi-headed self-attention module of Conformer block. **ii-c** Feed forward module of Conformer block

3.6.1 Convolution module

Figure 5ii-a demonstrates the details of the convolution module. The convolution module starts with a 1-dim pointwise convolution layer and a gated linear units (GLU) activation [60]. The 1-dim pointwise convolution layer doubles the input channels, and the GLU activation splits the input along the channel dimension and executes an element-wise product. What follows are a 1-dim depthwise convolution layer, a batch normalization layer, a Swish activation, and another 1-dim pointwise convolution layer. As mentioned before, the layer normalization is applied before each module, and the dropout is followed by a residual connection afterwards (pre-norm).

3.6.2 Macron-feedforward module

Unlike the FFN module in Transformer encoder [36], which comprises two linear transformations with a ReLU activation in between (Eq. 14), Conformer encoder [37] introduces another FFN module and substitutes the ReLU activation with the Swish activation. Furthermore, inspired by Macaron-Net [57], this pair of FFN modules are following a half-step scheme and sandwiching the MHSA and the convolution modules. The detail of the FFN is illustrated in Fig. 5ii-c.

4 Adversarial joint training

GANs aim at mapping samples $\hat{\boldsymbol{x}}$ from the distribution $\hat{\mathcal{X}}$ to samples $\boldsymbol{x}^*$ from another distribution $\mathcal{X}^*$. The generator G is tasked to learn an effective mapping that can imitate the real data distribution to generate novel samples from the manifold defined in the $\mathcal{X}$, by means of an adversarial training exerted by the discriminator D. During back-propagation, D classifies real samples from the fake samples more accurately; in return, G updates its parameters towards the real data manifold, till the mixed Nash equilibria are reached [50]. The GAN training process can be formulated as a minimax game between G and D, with the objective:

$$\min_G \max_D \mathcal{L}(D,G) = \mathbb{E}_{\boldsymbol{x}^*\sim p_{data}(\boldsymbol{x}^*)}[\log D(\boldsymbol{x}^*)] + \mathbb{E}_{\hat{\boldsymbol{x}}\sim p_{\hat{x}}(\hat{\boldsymbol{x}})}[\log(1 - D(G((\hat{\boldsymbol{x}}))))]\,. \tag{20}$$

In our proposed robust end-to-end speech recognition scheme, the discriminant network first acts as the local guide for the enhancement module, where D shifts the training of G towards the distribution of clean data; thereafter, it is deployed as the global guide for the whole scheme, where D instructs G to output pertinent enhanced data for the subsequent ASR task.

We first train the enhancement module, which contains both the generator and the discriminator. To solve the problem of vanishing gradients caused by sigmoid cross-entropy loss for training, the least-squares GAN (LSGAN) with binary coding (1 for real, 0 for fake) is utilized instead of the cross-entropy loss. Consequently, the loss function of the discriminator component changes to

$$\min_D \mathcal{L}(D) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}^*,\tilde{\boldsymbol{x}}\sim p_{data}(\boldsymbol{x}^*,\tilde{\boldsymbol{x}})}[D(\boldsymbol{x}^*,\tilde{\boldsymbol{x}}) - 1]^2 + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z}),\tilde{\boldsymbol{x}}\sim p_{data}(\tilde{\boldsymbol{x}})}[D(G(\boldsymbol{z},\tilde{\boldsymbol{x}}),\tilde{\boldsymbol{x}})]^2\,, \tag{21}$$

where $\boldsymbol{z}$ is a latent variable. To minimize the distance between its generations and the clean examples, it is beneficial to add a secondary component to the loss of G. Inspired by the effectiveness of L_1 norm in the image manipulation domain [61, 62], we deploy it in G component to gain more fine-grained and realistic results. The magnitude of the L_1 norm is controlled by a new hyper-parameter λ. Hence, the loss function of the generator component becomes:

$$\begin{aligned}\min_G \mathcal{L}(G) =& \frac{1}{2}\mathbb{E}_{z\sim p_z(z),\tilde{x}\sim p_{data}(\tilde{x})}[D(G(z,\tilde{x}),\tilde{x})-1]^2 \\ &+\lambda\left\|G(z,\tilde{x})-x^*\right\|_1.\end{aligned} \tag{22}$$

In the joint training, the enhancement module is initialized from the trained G component, while the global discriminant module is initialized from the trained D component. The training of the ASR component is based on the cross entropy criterion, namely:

$$\mathcal{L}_{asr} = -\ln P(y^*|f) = -\sum_n \ln P(y_n^*|f, y_{1:n-1}^*), \tag{23}$$

where y^* is the ground truth of a whole sequence of output labels, and $y_{1:n-1}^*$ is the ground truth from output step 1 to $n-1$. In the proposed framework, the parameters of all procedures, enhancement, feature extraction, ASR, and the discriminant network, are updated by stochastic gradient descent calculated by the loss function of the whole scheme. It is composed of three losses: $\mathcal{L}_{asr}$, $\mathcal{L}_{enh}$, and $\mathcal{L}_{gan}$, which correspond to Eqs. 23, 22, and 21, i.e.:

$$\mathcal{L} = \mathcal{L}_{asr} + \kappa\mathcal{L}_{enh} + \gamma\mathcal{L}_{gan}, \tag{24}$$

where κ and γ are two hyper-parameters weighting the magnitude of the enhancement loss and adversarial loss, respectively. Notably, the scheme targets the recognition performance, and the loss function of the discriminant network adapts the enhancement module implicitly. As a result, the discriminant network guides the enhancement module to serve the subsequent ASR task more properly. Accordingly, the unnecessary speech distortion caused by the enhancement process is alleviated.

5 Experimental setups

We systematically evaluate the robustness of the adversarial joint training framework, and ablation tests are conducted to validate the effects of (i) the enhancement front-end on the ASR task, (ii) the joint training on the whole scheme, and (iii) the GAN on the joint training.

5.1 Corpus

All experiments are executed on the open source Mandarin speech corpus, AISHELL-1 [63]. This corpus is 178-h-long, and its utterances contain 11 domains, e.g., smart home, autonomous driving, industrial production. A total of 400 speakers from different accent areas in China participate in the recording. The corpus is divided into training, development, and test sets. The training dataset contains 120,098 utterances from 340 speakers, the development dataset contains 14,326 utterances from 40 speakers, and the test dataset contains 7176 utterances from 20 speakers.

For the noisy data, we contaminate clean utterances in AISHELL-1 with 9 sorts of intrusions from the NOISEX-92 dataset [64] artificially as noisy utterances. We create noisy training, development, and test sets in the same manner. Note that besides the "matched" noisy test set, which is contaminated by the same intrusions as the training dataset, we also corrupt the test set with the rest 5 sorts of intrusions in the NOISEX-92 dataset as "unmatched" test materials. Table 1 exhibits the sorts the intrusions mixed in "match" and "unmatch" cases. All utterances are mixed with the intrusions at SNRs randomly sampled between [0 dB, 20 dB]. To sum up, we have two sorts of datasets for training:

- Clean: clean utterances from the training dataset of AISHELL-1
- Match: contaminated clean utterances (training dataset) with "matched" noises of Table 1

For test datasets, we have the following:

- Clean: clean utterances from the test dataset of AISHELL-1
- Match: contaminated clean utterances of the test set with the same intrusions ("matched" noises of Table 1) as "matched" training set
- Unmatch: contaminated clean utterances of the test set with different intrusions ("unmatched" noises of Table 1) from "matched" training set

5.2 Baseline

For the comparison purpose, we take the work from [28] as the baseline model.

In [28], the mask-based enhancement network is deployed as the front-end. It estimates a masking function to multiply the frequency domain feature of the noisy speech to form an estimate of the clean speech. For the ASR task, Liu et al. employ the ESPnet model [65]. It consists of an encoder network that maps the input feature sequence into a higher-level representation. Then, a location-based attention layer integrates the representation into a context vector with the attention weight vector. In the end, the decoder predicts the next output conditioned on the full sequence of previous predictions. Besides, there is an extra discriminant network, whose loss is weighted in the loss function of the whole scheme to optimize the joint training.

Importantly, the baseline model does not contain any self-attention layer. Furthermore, the discriminant work in the baseline model is an extra auxiliary module, which does not participate in the enhancement training directly.

Table 1 The demonstration of categories of intrusions utilized in "match" and "unmatch" cases

Intrusion	Description
Match	
White noise	Analog noise generator
Factory floor noise 1	Plate-cutting and electrical welding
Cockpit noise 1	Buccaneer jet traveling at 190 knots
Cockpit noise 3	F-16
Engine room noise	Destroyer
Military vehicle noise	Leopard 1 vehicle
Machine gun noise	Gun
Vehicle interior noise	Volvo 340
HF channel noise	HF radio channel
Unmatch	
Pink noise	Analog noise generator
Factory floor noise 2	Car production hall
Cockpit noise 2	Buccaneer jet traveling at 450 knots
Operations room background noise	Destroyer
Military vehicle noise	M109

By contrast, our work benefit from self-attention mechanism and the discriminant module exits innately, which is a component of the enhancement front-end. It acts as the local guide for the enhancement training, leading the enhancement network to output towards the distribution of the clean samples. Simultaneously, it also plays the role of the global guide, instructing the enhancement module and the ASR module better matched.

5.3 Configurations

5.3.1 Baseline

For the enhancement front-end, the input is the 257-dim logarithmic STFT features, and all input vectors are normalized to have the zero mean and the unit variance. The network is composed of 3-layer long short-term memory (LSTM) with 128 nodes, followed by a linear layer with the sigmoid activation function. The network outputs the masking estimate, whose size is equal to the input size, multiplying by the STFT feature of the noisy speech to estimate the clean speech.

For the ASR network, the input is the 80-dim normalized log FBank features transformed from the enhanced STFT features. The encoder is composed of 4-layer bidirectional LSTM (BLSTM) with 320 cells, while the decoder is composed of 1-layer unidirectional LSTM with 320 cells. After each BLSTM layer, a linear projection layer with 320 nodes is used to combine the forward and backward LSTM outputs. The location-based attention mechanism comprises 10 centered convolution filters of width 100. Besides, We also adopt a joint connectionist temporal classification (CTC)-attention multitask loss function [66] with the CTC loss weight as 0.1.

The discriminant network consists of a 4-layer convolution network, each of which is followed by the ReLU activation function [67].

For decoding, we use a beam search algorithm with the beam size 12. CTC rescores the hypotheses with 0.1 weight [66]. Besides, an external recurrent neural network (RNN) language model is also adopted with 0.2 weight during decoding.

5.3.2 The proposed joint training scheme

SA_SEGAN The SA_SEGAN is trained for 86 epochs with RMSprop [68] and a learning rate of 0.0002. The batch size is 50. During training, we extract 1-s chunks of raw waveforms ($L = 16,384$ samples) with a 50% overlap. During the test, we slide the window without overlapping through the whole duration of our test utterances and concatenate the outputs at the end of the stream. During both training and test, we employ a high-frequency preemphasis filter with a coefficient of 0.95 to all inputs. For the self-attention layer in SA_SEGAN, we use $b = 8$ and $p = 4$ for memory reduction. Phan et al. [40] suggest that the placement of the self-attention layer does not show a clear difference on the performance, which indicates that applying the self-attention to the higher-level (de)convolutional layer is expected to be as good as to a lower layer. Compromising between the computation time and memory requirement and the performance, we place the self-attention layer in the 10th layer (l=10).

FBank extraction network The FBank feature extraction network is a linear layer to transform the raw outputs from the upstream SA_SEGAN to the downstream ASR procedure. We extract 80-dim filterbanks with the window size of 25 ms and the window shift of 10 ms, extended with the temporal first- and second-order differences. Thereafter, we do the logarithmic calculation and global mean and variance normalization according to Eq. 10.

Transformer For training the Transformer, we adopt Adam optimizer [69] with $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$, and vary the learning rate over the course of training according to the formula:

$$lr = k' \cdot d_{model}^{-0.5} \cdot \min(n^{-0.5}, n \times warmup_n^{-1.5}), \qquad (25)$$

where n denotes the step number. k' is a tunable scalar, which is set to be 10 initially and is declined to 1 when the model converges. The learning rate increases linearly during the fist $warmup_n = 25,000$ steps, and afterwards, it decreases proportionally to the inverse square root of the step number. We apply the residual dropout to each sub-block before adding the residual information, while the attention dropout is performed on the softmax activations in each attention. Both of these aforementioned dropouts are set to be 0.1. Additionally, we guide the system to be more attentive on closer positions by punishing the attention weights of more distant position pairs. Similar to the baseline model, we also adopt a joint CTC-attention multi-task loss function [66], with the CTC loss weight as 0.3. In the decoding, we set the beam size to 12 and length penalty $\alpha = 1.0$ [70]. Besides, we also integrate an external RNN language model with 0.3 weight. The training procedure is stopped after 30 epochs.

Conformer The model hyper-parameters of the Conformer are N_e=12, N_d=6, H=4, d_k=256, and d_{ff}=2048. The convolution subsampling layer possesses a 2-layer convolutional neural network (CNN) with 256 channels, stride with 2, and kernel size with 3. The kernel size of the convolution module is 31. We apply dropout in each residual unit of the Conformer with the weight 0.1. The same as the Transformer, we train the network with Adam optimizer [69] with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-9}$ and a Transformer learning rate schedule [49] with 10,000 warm-up steps. The learning rate is peaked at $0.05/\sqrt{d}$, where d is the model dimension in the Conformer encoder. Note that we do not apply speed perturbation [71] or SpecAugment [72] for the data augmentation to exclude extra tricks that could cause performance improvements. The training procedure is stopped after 30 epochs.

6 Results

We use character error rate (CER) to quantify the system performance in all experiments. We report CER of the AISHELL-1 test set on three conditions: "clean" refers to the original clean test dataset of the corpus, "match" denotes the noisy test dataset contaminated by "matched" sorts of intrusions in Table 1, and "unmatch" means the noisy test set corrupted by the rest of "unmatched" sorts of intrusions in Table 1. To validate the efficacy of the enhancement front-end, we also introduce multi-condition training (MCT), a popular training strategy for robust speech recognition, for comparison. Different from the training data generation of the speech enhancement front-end, the training data of MCT contains 10% clean utterances, which are chosen randomly from the training set of AISHELL-1. Except for the 10% partition, the remaining 90% of the training data is generated in the same manner as that of the enhancement front-end, namely being corrupted by the "matched" intrusions from Table 1 at an SNR in the range [0 dB, 20 dB].

Firstly, we train the ASR network with the original clean utterance and multi-condition training strategy. The results are shown in Table 2.

Ranking these three models from the aspect of ASR performance, the first is Conformer, then Transformer, and the last is the baseline model, consistent with observations in [36, 37]. However, their performance deteriorates rapidly in the noisy test set, demonstrating the necessity of the robustness investigation. The MCT training considerately improves the system's robustness. Its performance on the "matched" test set outperforms the clean training by 63.2% and 62.9% relative, in cases of Transformer and Conformer model respectively, while on the "unmatched" dataset, it outperforms the clean training by 56.3% and 56.8% relatively, in cases of Transformer and Conformer model, respectively.

Table 2 CER [%] results of the ASR system trained by clean data and multi-condition training (MCT) without the enhancement

ASR	Training	CER [%]		
		Clean	Match	Unmatch
Baseline	Clean	14.0	60.0	61.6
	MCT	14.6	20.8	27.3
Transformer	Clean	8.0	35.6	37.1
	MCT	7.8	13.1	16.2
Conformer	Clean	6.5	32.6	31.7
	MCT	6.9	12.1	13.7

Secondly, we train SA_SEGAN with the training data contaminated by "matched" intrusions in Table 1 to enhance the noisy speech. Then, the enhanced features are used for the downstream ASR task. Importantly, the ASR models are taken over from the same well-trained model as in Table 2, which means that the enhancement front-end and the ASR back-end are trained separately by different objectives. As exhibited in Table 3, the enhancement module tremendously improves the performance of the ASR component, which is trained by the clean data merely. Compared to Table 2, it outperforms all of the three ASR modules (baseline, Transformer, Conformer) without the enhancement front-end. The improvement achieved in the "matched" dataset is more remarkable than that achieved on the "unmatched" test set. For instance, it outperforms the Conformer without the enhancement module by 46.0% in the "matched" test set while 9.1% in the "unmatched" test set. This difference is due to the fact that the SA_SEGAN is trained with the "matched" intrusions and can enhance the data contaminated by the same intrusions better during the test. All these improvements confirm the efficacy of the enhancement module for improving the robustness of the ASR system. Nevertheless, improving the robustness of the framework in unseen noisy environments still remains to be a challenge. Additionally, the speech enhancement module deteriorates the performance of the ASR_MCT network, which stays in accordance with the observations in [35] and [73]. Donahue et al. [35] and Narayanan

Table 3 The impacts of the enhancement front-end on the ASR systems trained by clean data and multi-condition training (MCT). The results are in CER [%]

ASR	Training	CER [%]		
		Clean	Match	Unmatch
Baseline	Clean	13.9	25.8	53.8
	MCT	14.9	23.5	34.3
Transformer	Clean	8.1	19.1	33.7
	MCT	7.9	14.9	20.7
Conformer	Clean	6.5	17.6	28.8
	MCT	7.0	14.2	17.9

and Wang [73] hypothesize that the enhancement front-end might be introducing hitherto-unseen distortions that compromise performance. Furthermore, we believe that this latent distortion is derived from the independent training of the enhancement module, demonstrating the necessity of the joint training strategy.

To remedy the deterioration of the performance of the ASR_MCT, we retrain the network with the enhanced features. Assuming that the network may also benefit from the knowledge of the noisy features, we also experimented with ingesting both enhanced and noisy features. The results are displayed in Table 4. Either the Transformer_MCT model or the Conformer_MCT model is initialized from the existing well-trained MCT checkpoints respectively, setting the additional parameters to zero to ensure the fair training start. As presented in Table 4, the retraining with the enhanced features improves the performance in both “matched” and “unmatched” cases, and the retraining with both enhanced and noisy features improves the performance slightly further.

Lastly, we jointly train the whole scheme with and without adversarial training according to Eq. 24. In the framework, the enhancement front-end is initialized from the generator (G component) of SA_SEGAN, the ASR back-end is initialized from the ASR_MCT checkpoint (without retraining), and the adversarial module is initialized from the discriminator (D component) of SA_SEGAN. When the adversarial module participates in the training, we set the magnitude of the loss function by κ=6.0 and γ=0; by contrast, when it participates in the training, we set κ=6.0 and γ=3.0. The results are presented in Table 5. Compared to Table 2, the joint training mitigates the distortion problem existing in the MCT strategy. Additionally, the participance of the adversarial training improves the performance further and exceeds the performance of retraining with both enhanced and noisy features in either Transformer or Conformer case. Taking Conformer for example, compared to Conformer trained with clean data merely, the adversarial joint training yields 63.8% relative and 59.0% relative improvements on “matched" and “unmatched" datasets, respectively; meanwhile, the adversarial joint training outperforms the MCT strategy by 2.5% relative and 5.2% relative on “matched" and “unmatched" datasets, separately. These results indicate the efficacy of the adversarial joint training in improving the robustness of the end-to-end ASR scheme.

Table 4 CER [%] results of the SE_ASR system retraining with and without noisy features

ASR	Retraining	CER [%]		
		Clean	Match	Unmatch
Transformer_MCT	No	7.8	13.1	16.2
	Enhanced	7.8	12.9	15.8
	Enhanced + noisy	7.8	12.9	15.6
Conformer_MCT	No	6.9	12.1	13.7
	Enhanced	6.7	12.0	13.5
	Enhanced + noisy	6.7	11.8	13.3

7 Discussion

To analyze the difference between these enhancement modules that are trained independently, jointly without GANs, and jointly with GANs, we quantify their performance on the following five objective criteria (the higher the better):

- SSNR: segmental SNR [23] (in the range of $[0, +\infty)$)
- CBAK: MOS prediction of the intrusiveness of background noises [22] (in the range of $[1, 5]$)
- CSIG: MOS prediction of the signal distortion attending only to the speech signal [22] (in the range of $[1, 5]$)
- COVL: MOS prediction of the overall effect [22] (in the range of $[1, 5]$)
- PESQ: perceptual evaluation of speech quality, using the wide-band version recommended in ITU-T P.862.2 [74] (in the range of $[-0.5, 4.5]$)

All criteria are computed based on the implementation in [75], available at the publisher website[1]. We quantify the performance of the enhancement front-end that is trained independently, trained jointly with and without GANs in cases of baseline, Transformer, and Conformer schemes. As exhibited in Figs. 6, 7, and 8, the joint training slightly degrades the enhancement module’s performance on SSNR, CBAK, COVL, CSIG, and PESQ generally. These results suggest that these objective criteria cannot indicate the suitability of the enhanced data for ASR task, which verifies that it is hard for independent training to lead the enhancement module to the global optimum.

[1]https://www.crcpress.com/downloads/K14513/K14513_CD_Files.zip

Table 5 The impacts of the joint training with and without GAN on SA-ASR pipeline. The results are in CER [%]

SE	ASR	joint training with GANs	CER[%]		
			Clean	Match	Unmatch
SA_SEGAN	Baseline	No	12.8	18.7	25.3
		Yes	**12.8**	**18.7**	**24.8**
	Transformer	No	**7.0**	12.4	15.6
		Yes	7.2	**12.4**	**15.5**
	Conformer	No	**6.8**	11.9	13.3
		Yes	6.9	**11.8**	**13.0**

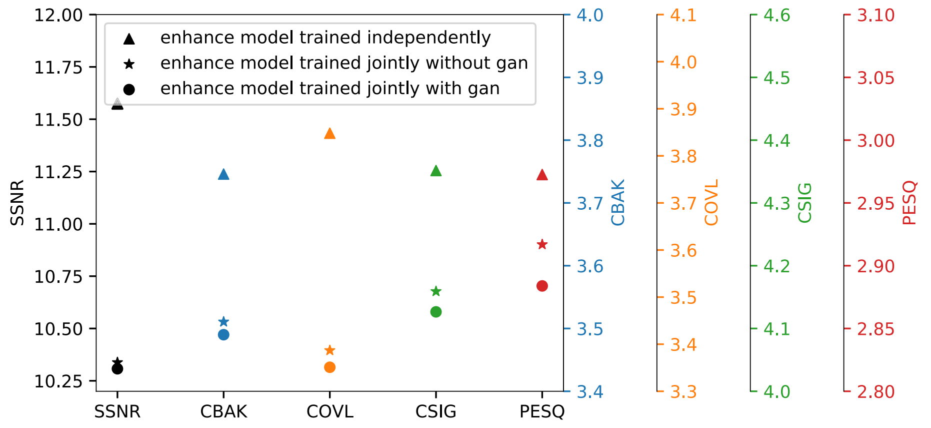

Fig. 6 The performance comparison of the enhancement model trained independently and the enhancement models trained jointly with the baseline ASR model without and with GAN

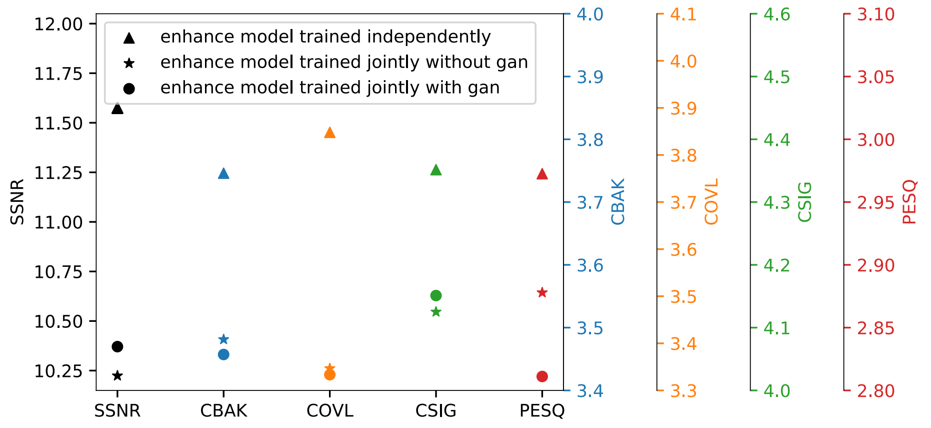

Fig. 7 The performance comparison of the enhancement model trained independently and the enhancement models trained jointly with Transformer ASR model without and with GAN

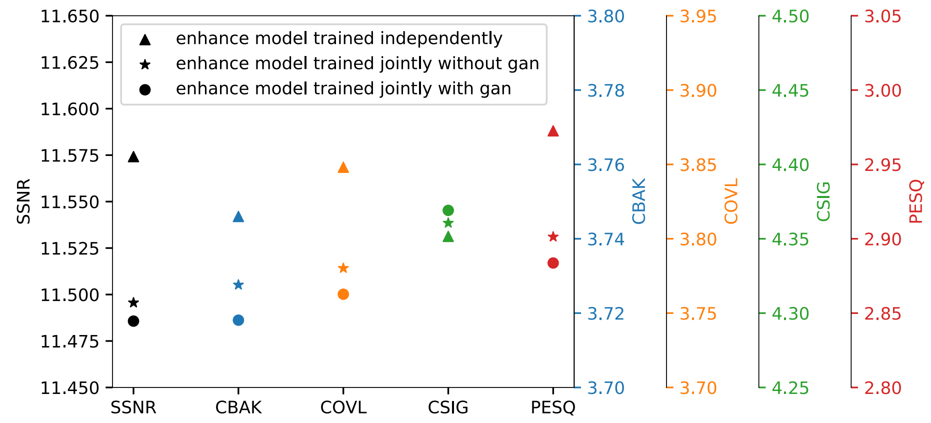

Fig. 8 The performance comparison of the enhancement model trained independently and the enhancement models trained jointly with Conformer ASR model without and with GAN

Another phenomenon which is worth noting is that the discrepancies on CBAK and SSNR suggests that there are conflicts between erasing the noise contamination and averting the speech distortion. Therefore, an equilibrium between these two goals should be sought. The experimental results in Section 6 validate the efficacy of the adversarial joint training with a global discriminant guide for reaching the equilibrium point.

8 Conclusion

In this paper, we propose an adversarial joint training framework with the self-attention mechanism to boost the noise robustness of the end-to-end ASR system. The jointly compositional scheme consists of an enhancement front-end, a recognition back-end, and the discriminant network. A highlight of this proposed framework is the discriminant component first acts as the guide of the enhancement front-end training; afterwards, it participates in the adversarial joint training as the global instructor, which leads the enhancement front-end to output appropriate enhanced features for the downstream ASR task. Experimental results validate the efficacy of the proposed adversarial joint training strategy. The next work plan is to investigate different framework architectures and training strategies for further improved performance.

Abbreviations

ASR: Automatic speech recognition; SA_ASR: Sela-attention automatic speech recognition; GANs: Generative adversarial networks; LSGAN: Least-squares generative adversarial networks; SE: Speech enhancement; SEGAN: Speech enhancement generative adversarial networks; SA_SEGAN: Self-attention speech enhancement generative adversarial networks; CER: Character error rate; PReLUs: Parametric rectified linear units; FFN: Feed-forward network; MHSA: Multi-head self-attention module; GLU: Gated linear units; STFT: Short-time Fourier transform; LSTM: Long short-term memory; BLSTM: Bidirectional long short-term memory; CTC: Connectionist temporal classification; RNN: Recurrent neural network; CNN: Convolutional neural network; MCT: Multi-conditional training

Authors' contributions

Li, L. conceptualized the study. Li, L.; Kang, Y.; and Shi, Y. executed the experiments. All authors did the literature analysis, manuscript preparation, editing, and proofreading, and approved the final manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Availability of data and materials

The dataset is the open source Mandarin speech corpus, AISHELL-1 [63], and can be found under the following link: http://www.aishelltech.com/kysjcp.

Declarations

Competing interests

The authors declare that they have no competing interests.

Received: 4 February 2021 Accepted: 9 June 2021
Published online: 05 July 2021

References

1. W. Chan, N. Jaitly, Q. V. Le, O. Vinyals, Listen, attend and spell (2015). arXiv preprint arXiv:1508.01211
2. J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio, Attention-based models for speech recognition. Adv. Neural Inf. Process Syst. **28**, 577–585 (2015)
3. G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al*, Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process Mag. **29**(6), 82–97 (2012)
4. C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, *et al*, in *Proc. ICASSP*. State-of-the-art speech recognition with sequence-to-sequence models (IEEE, 2018), pp. 4774–4778. https://doi.org/10.1109/icassp.2018.8462105
5. D. Povey, H. Hadian, P. Ghahremani, K. Li, S. Khudanpur, in *Proc. ICASSP*. A time-restricted self-attention layer for ASR (IEEE, 2018), pp. 5874–5878. https://doi.org/10.1109/icassp.2018.8462497
6. Z. Tian, J. Yi, J. Tao, Y. Bai, Z. Wen, Self-attention transducers for end-to-end speech recognition (2019). arXiv preprint arXiv:1909.13037
7. J. Salazar, K. Kirchhoff, Z. Huang, in *Proc. ICASSP*. Self-attention networks for connectionist temporal classification in speech recognition (IEEE, 2019), pp. 7115–7119. https://doi.org/10.1109/icassp.2019.8682539
8. K. J. Han, J. Huang, Y. Tang, X. He, B. Zhou, in *Proc. INTERSPEECH*. Multi-stride self-attention for speech recognition, (2019), pp. 2788–2792. https://doi.org/10.21437/interspeech.2019-1973

9. K. J. Han, R. Prieto, T. Ma, in *Proc. ASRU*. State-of-the-art speech recognition using multi-stream self-attention with dilated 1D convolutions (IEEE, 2019), pp. 54–61. https://doi.org/10.1109/asru46091.2019.9003730
10. N.-Q. Pham, T.-S. Nguyen, J. Niehues, M. Müller, S. Stüker, A. Waibel, Very deep self-attention networks for end-to-end speech recognition (2019). arXiv preprint arXiv:1904.13377
11. C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar, Y. Wang, D. Le, M. Jain, K. Schubert, C. Fuegen, M. L. Seltzer, Transformer-transducer: End-to-end speech recognition with self-attention (2019). arXiv preprint arXiv:1910.12977
12. H. Luo, S. Zhang, M. Lei, L. Xie, Simplified self-attention for transformer-based end-to-end speech recognition (2020). arXiv preprint arXiv:2005.10463
13. J. Lim, A. Oppenheim, All-pole modeling of degraded speech. IEEE Trans. Acoust. Speech Sig. Process. **26**(3), 197–210 (1978)
14. A. Narayanan, D. Wang, in *Proc. ICASSP*. Ideal ratio mask estimation using deep neural networks for robust speech recognition (IEEE, 2013), pp. 7092–7096. https://doi.org/10.1109/icassp.2013.6639038
15. Y. Wang, A. Narayanan, D. Wang, On training targets for supervised speech separation. IEEE/ACM Trans. Audio Speech Lang. Process. **22**(12), 1849–1858 (2014)
16. S. Nie, S. Liang, W. Xue, X. Zhang, W. Liu, et al., in *Proc. INTERSPEECH*. Two-stage multi-target joint learning for monaural speech separation (International Speech Communication Association (ISCA), Dresden, 2015), pp. 1503–1507
17. F. Weninger, J. R. Hershey, J. Le Roux, B. Schuller, in *Proc. GlobalSIP*. Discriminatively trained recurrent neural networks for single-channel speech separation (IEEE, 2014), pp. 577–581. https://doi.org/10.1109/globalsip.2014.7032183
18. H. Erdogan, J. R. Hershey, S. Watanabe, J. Le Roux, in *Proc. ICASSP*. Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks (IEEE, 2015), pp. 708–712. https://doi.org/10.1109/icassp.2015.7178061
19. Y. Xu, J. Du, L.-R. Dai, C.-H. Lee, A regression approach to speech enhancement based on deep neural networks. IEEE/ACM Trans. Audio Speech Lang. Process. **23**(1), 7–19 (2014)
20. S. Nie, S. Liang, W. Liu, X. Zhang, J. Tao, Deep learning based speech separation via NMF-style reconstructions. IEEE/ACM Trans. Audio Speech Lang. Process. **26**(11), 2043–2055 (2018)
21. Y. Ephraim, in *Proc. ICASSP*. A minimum mean square error approach for speech enhancement (IEEE, 1990), pp. 829–832. https://doi.org/10.1109/icassp.1990.115960
22. Y. Hu, P. C. Loizou, Evaluation of objective quality measures for speech enhancement. IEEE Trans. Audio Speech Lang. Process. **16**(1), 229–238 (2007)
23. S. R. Quackenbush, T. P. Barnwell, M. A. Clements, *Objective measures of speech quality*, Ellis Horwood Series in Artificial Intelligence. (Prentice Hall, 1988)
24. M. L. Seltzer, in *Hands-Free Speech Communication and Microphone Arrays*. Bridging the gap: towards a unified framework for hands-free speech recognition using microphone arrays (IEEE, 2008), pp. 104–107. https://doi.org/10.1109/hscma.2008.4538698
25. Z.-Q. Wang, D. Wang, A joint training framework for robust automatic speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. **24**(4), 796–806 (2016)
26. Z.-q. Wang, D. Wang, in *Proc. INTERSPEECH*. Joint training of speech separation, filterbank and acoustic model for robust automatic speech recognition (International Speech Communication Association (ISCA), Dresden, 2015)
27. T. Ochiai, S. Watanabe, T. Hori, J. R. Hershey, Multichannel end-to-end speech recognition (2017). arXiv preprint arXiv:1703.04783
28. B. Liu, S. Nie, S. Liang, W. Liu, M. Yu, L. Chen, S. Peng, C. Li, in *Proc. INTERSPEECH*. Jointly adversarial enhancement training for robust end-to-end speech recognition (IEEE, 2019), pp. 491–495. https://doi.org/10.21437/interspeech.2019-1242
29. S. Pascual, A. Bonafonte, J. Serra, SEGAN: speech enhancement generative adversarial network (2017). arXiv preprint arXiv:1703.09452
30. M. H. Soni, N. Shah, H. A. Patil, in *Proc. ICASSP*. Time-frequency masking-based speech enhancement using generative adversarial network (IEEE, 2018), pp. 5039–5043. https://doi.org/10.1109/icassp.2018.8462068
31. D. Michelsanti, Z.-H. Tan, Conditional generative adversarial networks for speech enhancement and noise-robust speaker verification (2017). arXiv preprint arXiv:1709.01703
32. P. Shen, X. Lu, S. Li, H. Kawai, in *Proc. INTERSPEECH*. Conditional generative adversarial nets classifier for spoken language identification (IEEE, 2017), pp. 2814–2818. https://doi.org/10.21437/interspeech.2017-553
33. S. Sahu, R. Gupta, C. Espy-Wilson, On enhancing speech emotion recognition using generative adversarial networks (2018). arXiv preprint arXiv:1806.06626
34. H. Hu, T. Tan, Y. Qian, in *Proc. ICASSP*. Generative adversarial networks based data augmentation for noise robust speech recognition (IEEE, 2018), pp. 5044–5048. https://doi.org/10.1109/icassp.2018.8462624
35. C. Donahue, B. Li, R. Prabhavalkar, in *Proc. ICASSP*. Exploring speech enhancement with generative adversarial networks for robust speech recognition (IEEE, 2018), pp. 5024–5028. https://doi.org/10.1109/icassp.2018.8462581
36. L. Dong, S. Xu, B. Xu, in *Proc. ICASSP*. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition (IEEE, 2018), pp. 5884–5888. https://doi.org/10.1109/icassp.2018.8462506
37. A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, *et al*, Conformer: convolution-augmented transformer for speech recognition (2020). arXiv preprint arXiv:2005.08100
38. H. Phan, I. V. McLoughlin, L. Pham, O. Y. Chén, P. Koch, M. De Vos, A. Mertins, Improving GANs for speech enhancement. IEEE Sig. Process Lett. **27**, 1700–1704 (2020)
39. D. Baby, isegan: improved speech enhancement generative adversarial networks (2020). arXiv preprint arXiv:2002.08796
40. H. Phan, H. L. Nguyen, O. Y. Chén, P. Koch, N. Q. Duong, I. McLoughlin, A. Mertins, Self-attention generative adversarial network for speech enhancement (2020). arXiv preprint arXiv:2010.09132
41. Y. Koizumi, K. Yatabe, M. Delcroix, Y. Masuyama, D. Takeuchi, in *Proc. ICASSP*. Speech enhancement using self-adaptation and multi-head self-attention (IEEE, 2020), pp. 181–185. https://doi.org/10.1109/icassp40776.2020.9053214
42. A. Sriram, H. Jun, Y. Gaur, S. Satheesh, in *Proc. ICASSP*. Robust speech recognition using generative adversarial networks (IEEE, 2018), pp. 5639–5643. https://doi.org/10.1109/icassp.2018.8462456
43. K. Wang, J. Zhang, S. Sun, Y. Wang, F. Xiang, L. Xie, in *Proc. INTERSPEECH*. Investigating generative adversarial networks based speech dereverberation for robust speech recognition (IEEE, 2018), pp. 1581–1585. https://doi.org/10.21437/interspeech.2018-1780
44. B. Liu, S. Nie, Y. Zhang, D. Ke, S. Liang, W. Liu, in *Proc. ICASSP*. Boosting noise robustness of acoustic model via deep adversarial training (IEEE, 2018), pp. 5034–5038. https://doi.org/10.1109/icassp.2018.8462093
45. J. Droppo, A. Acero, in *Proc. INTERSPEECH*. Joint discriminative front end and back end training for improved speech recognition accuracy, vol. 1 (IEEE, 2006), pp. 281–284. https://doi.org/10.1109/icassp.2006.1660012
46. T. Gao, J. Du, L. Dai, C. Lee, in *Proc. ICASSP*. Joint training of front-end and back-end deep neural networks for robust speech recognition (IEEE, 2015), pp. 4375–4379. https://doi.org/10.1109/icassp.2015.7178797
47. M. Ravanelli, P. Brakel, M. Omologo, Y. Bengio, in *Proc. SLT*. Batch-normalized joint training for DNN-based distant speech recognition (IEEE, 2016), pp. 28–34. https://doi.org/10.1109/slt.2016.7846241
48. Y. Qian, T. Tan, D. Yu, Neural network based multi-factor aware joint training for robust speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. **24**(12), 2231–2240 (2016)
49. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need. Adv. Neural Inf. Process Syst. **30**, 5998–6008 (2017)
50. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets. Adv. Neural Inf. Process Syst. **27**, 2672–2680 (2014)
51. X. Wang, R. Girshick, A. Gupta, K. He, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Non-local neural networks (IEEE Computer Society, Salt Lake City, 2018), pp. 7794–7803
52. H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, in *International Conference on Machine Learning*. Self-attention generative adversarial networks (Association for Computing Machinery, New York, 2019), pp. 7354–7363
53. A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks (2015). arXiv preprint arXiv:1511.06434

54. K. He, X. Zhang, S. Ren, J. Sun, in *Proceedings of the IEEE International Conference on Computer Vision*. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, (2015), pp. 1026–1034. https://doi.org/10.1109/iccv.2015.123
55. T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs (2016). arXiv preprint arXiv:1606.03498
56. A. L. Maas, A. Y. Hannun, A. Y. Ng, in *Proc. Icml*. Rectifier nonlinearities improve neural network acoustic models, vol. 30 (JMLR.org, Atlanta, 2013), p. 3
57. Y. Lu, Z. Li, D. He, Z. Sun, B. Dong, T. Qin, L. Wang, T.-Y. Liu, Understanding and improving transformer from a multi-particle dynamic system point of view (2019). arXiv preprint arXiv:1906.02762
58. A. Zeyer, P. Bahar, K. Irie, R. Schlüter, H. Ney, in *Proc. ASRU*. A comparison of transformer and LSTM encoder decoder models for ASR (IEEE, 2019), pp. 8–15. https://doi.org/10.1109/asru46091.2019.9004025
59. Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, L. S. Chao, Learning deep transformer models for machine translation (2019). arXiv preprint arXiv:1906.01787
60. Y. N. Dauphin, A. Fan, M. Auli, D. Grangier, in *International Conference on Machine Learning*. Language modeling with gated convolutional networks (International Conference on Machine Learning (IOML), Sydney, 2017), pp. 933–941
61. P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Image-to-image translation with conditional adversarial networks, (2017), pp. 1125–1134. https://doi.org/10.1109/cvpr.2017.632
62. D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, A. A. Efros, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Context encoders: feature learning by inpainting, (2016), pp. 2536–2544. https://doi.org/10.1109/cvpr.2016.278
63. H. Bu, J. Du, X. Na, B. Wu, H. Zheng, in *20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. Aishell-1: an open-source mandarin speech corpus and a speech recognition baseline, (2017), pp. 1–5. https://doi.org/10.1109/icsda.2017.8384449
64. A. Varga, H. Steeneken, D. Jones, The noisex-92 study on the effect of additive noise on automatic speech recognition system. Speech Commun. **12**(3), 247–251 (1992)
65. S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, T. Ochiai, in *Proc. INTERSPEECH*. ESPnet: end-to-end speech processing toolkit (IEEE, 2018), pp. 2207–2211. https://doi.org/10.21437/interspeech.2018-1456
66. S. Kim, T. Hori, S. Watanabe, in *Proc. ICASSP*. Joint CTC-attention based end-to-end speech recognition using multi-task learning (IEEE, 2017), pp. 4835–4839. https://doi.org/10.1109/icassp.2017.7953075
67. V. Nair, G. E. Hinton, in *ICML*. Rectified linear units improve restricted boltzmann machines (Omnipress, Madison, 2010)
68. T. Tieleman, G. Hinton, Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA Neural Netw. Mach. Learn. **4**(2), 26–31 (2012)
69. D. P. Kingma, J. Ba, Adam: a method for stochastic optimization (2014). arXiv preprint arXiv:1412.6980
70. Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al*, Google's neural machine translation system: bridging the gap between human and machine translation (2016). arXiv preprint arXiv:1609.08144
71. T. Ko, V. Peddinti, D. Povey, S. Khudanpur, in *Sixteenth Annual Conference of the International Speech Communication Association*. Audio augmentation for speech recognition (International Speech Communication Association (ISCA), Dresden, 2015)
72. D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, Q. V. Le, Specaugment: a simple data augmentation method for automatic speech recognition (2019). arXiv preprint arXiv:1904.08779
73. A. Narayanan, D. Wang, in *Proc. ICASSP*. Joint noise adaptive training for robust automatic speech recognition (IEEE, 2014), pp. 2504–2508. https://doi.org/10.1109/icassp.2014.6854051
74. ITU, Recommendation ITU-T P, 862.2: Wideband Extension to Recommendation P. 862 for the Assessment of Wideband Telephone Networks and Speech Codecs. ITU-Telecommunication Standardization Sector, 2007
75. P. C. Loizou, *Speech enhancement: theory and practice*. (CRC press, Boca Raton, 2013)

Publisher's Note