Adversarial Regularization for Attention Based End-to-End Robust Speech Recognition

Sining Sun, *Student Member, IEEE*, Pengcheng Guo, Lei Xie, *Senior Member, IEEE*, and Mei-Yuh Hwang, *Fellow, IEEE*

***Abstract*—End-to-end speech recognition, such as attention based approaches, is an emerging and attractive topic in recent years. It has achieved comparable performance with the traditional speech recognition framework. Because end-to-end approaches integrate acoustic and linguistic information into one model, the perturbation in the acoustic level such as acoustic noise, could be easily propagated to the linguistic level. Thus improving model robustness in real application environments for these end-to-end systems is crucial. In this paper, in order to make the attention based end-to-end model more robust against noises, we formulate regulation of the objective function with adversarial training examples. Particularly two adversarial regularization techniques, the fast gradient-sign method and the local distributional smoothness method, are explored to improve noise robustness. Experiments on two publicly available Chinese Mandarin corpora, AISHELL-1 and AISHELL-2, show that adversarial regularization is an effective approach to improve robustness against noises for our attention-based models. Specifically, we obtained 18.4% relative character error rate (CER) reduction on the AISHELL-1 noisy test set. Even on the clean test set, we showed 16.7% relative improvement. As the training set increases and covers more environmental varieties, our proposed methods remain effective despite that the improvement shrinks. Training on the large AISHELL-2 training corpus and testing on the various AISHELL-2 test sets, we achieved 7.0%–12.2% relative error rate reduction. To our knowledge, this is the first successful application of adversarial regularization to sequence-to-sequence speech recognition systems.**

***Index Terms*—Sequence-to-sequence, attention, adversarial training, virtual adversarial training, Listen Attend and Spell, cross entropy.**

I. Introduction

In RECENT years the performance of speech recognition has been improved dramatically due to successful application of deep neural networks (DNNs) [1] and tremendous amount of real-world speech data. The hybrid framework of hidden Markov model (HMM) and DNN (HMM-DNN) is widely adopted by most automatic speech recognition (ASR) systems. Although this hybrid framework can obtain satisfactory performance, the building process is pretty complex. In recent years, researchers tried to simplify the foregoing hybrid HMM-DNN framework by using end-to-end modeling techniques, which can jointly optimize all components of speech recognition systems, including acoustic models (AM) and language models (LM). Currently, two dominated modeling techniques were widely adopted by end-to-end speech recognition systems: the connectionist temporal classification (CTC) based model [2], [3] and the attention based model [4], [5] such as the "Listen, Attend and Spell (LAS)" model. Both approaches regard speech recognition as a sequence-to-sequence (seq-to-seq) task, which maps the speech feature sequences to text label sequences. Recent results show that end-to-end models have achieved comparable performance as HMM-DNN models in some scenarios [6].

Given the fast development and bright prospect of end-to-end framework, model robustness should be taken into account for practical reasons. Although end-to-end modeling techniques can significantly simplify the model building procedure and get comparable performance as conventional hybrid systems, it still suffers performance degradation from noises, accents, etc., because these factors could introduce training-testing mismatch. Different from conventional hybrid systems, end-to-end models jointly learn acoustic and linguistic information simultaneously. Therefore, the perturbation in the acoustic level could be easily propagated into the linguistic level. From this perspective, end-to-end framework is more vulnerable to these mismatches. So in real-world speech recognition tasks, end-to-end models are usually less robust and show performance degradation. For example, when noises exist, the alignment learned form attention mechanism is easily corrupted. Hence improving model robustness for end-to-end systems has attracted a lot of research interests recently [7]–[11].

Among these methods, some researchers have begun to apply *adversarial learning* to robust end-to-end modeling. Although the concept of adversarial learning was proposed only in recent years [12]–[14], especially in the computer vision field, it has been recently applied to speech processing tasks, including speech enhancement [15], [16], speech recognition [17]–[19] and speech synthesis [20], [21]. Generative adversarial network (GAN) [12] and domain adversarial training (DAT) are commonly adopted in speech recognition. Besides from GAN and DAT, another interesting concept is *adversarial examples*, which was first proposed by Szegedy *et al.* [22] in image classification task. The authors found that a formerly correctly classified

Manuscript received February 20, 2019; revised June 23, 2019 and July 27, 2019; accepted July 27, 2019. Date of publication August 5, 2019; date of current version August 21, 2019. This work was supported by the National Natural Science Foundation of China under Grant 61571363. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jinyu Li. *(Sining Sun and Pengcheng Guo contributed equally to this work.) (Corresponding author: Lei Xie.)*

S. Sun, P. Guo, and L. Xie are with the School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: ssning2013@gmail.com; pcguo@nwpu-aslp.org; xielei21st@gmail.com).

M.-Y. Hwang is with the Mobvoi AI Lab, Redmond, WA 98052, USA (e-mail: mhwang@mobvoi.com).

Digital Object Identifier 10.1109/TASLP.2019.2933146

example could be mis-classified by neural network when a very small perturbation is added to the original example, even the perturbation is unnoticeable to human. This kind of perturbation is called "adversarial perturbation". Many experimental results showed that even the state-of-the-art deep learning models are vulnerable to such adversarial examples [23]–[26]. The existence of adversarial examples indicate that there are "blind spots" in the input space of deep learning models. In other words, the models are *un-smooth* because a tiny perturbation in input space could trigger a drastic change in the output space.

Previous work has focused on improving model robustness against adversarial test examples [22], [27]. In our recent work on robust speech recognition [28], we have adopted the idea and augmented training data with adversarial examples to obtain more robust DNN-HMM acoustic models against noise interferences instead of adversarial examples only. In contrast to adversarial training like DAT, where the model is trained to be invariant to specific phenomena represented in the training set, the adversarial examples used here are generated automatically based on inputs and model parameters associated with each mini-batch. In the training stage, we generate adversarial examples dynamically using the fast gradient sign method (FGSM) [27]. For each mini-batch, after the adversarial examples are obtained, the parameters in the model are updated with both the original and the adversarial examples. Experiments on the Aurora-4 and CHiME-4 single-channel tasks have shown improved robustness against noise and channel variation.

In this paper, we further explore the potential ability of adversarial examples for robust end-to-end speech recognition. Recent studies have demonstrated that end-to-end speech recognition models, such as LAS, are vulnerable to adversarial attacks [29]–[31]. This is because the LAS model, which unifies the acoustic and language information, predicts words or characters from the acoustic features directly with a simple decoding schedule and the perturbation in the acoustic level is easily propagated to the linguistic outputs, leading to wrong predictions. In this paper, inspired by adversarial examples, we propose to train the attention based end-to-end speech recognition models using *adversarial regularization*. Different from our previous approach that uses adversarial examples directly for training data augmentation [28], in this paper, we use adversarial examples for model regularization. Specifically, in adversarial regularization, the loss function is amended by the loss on adversarial examples, in addition to the loss on the original data. Explicitly introducing adversarial examples into the loss function can make model more robust to minor deviations from the original training data. Regularization, such as L_2 regularization [32], is widely adopted to avoid overfitting a model to the training data. From Bayesian viewpoint, the regularization term is closely related to the prior distribution of the parameters [32]. Another regularization trick is KL-Divergence (KLD) [33] used for DNN acoustic model adaptation to alleviate overfitting during adaptation process, such as speaker adaptation, when available adaptation data is limited. Different from the previous approaches, in our work, we propose to train LAS-based end-to-end speech recognition models using loss functions with adversarial regularization, which simply regularizes the model's prediction to be less sensitive to small perturbations applied to examples. Given a normal observed example, adversarial regularization methods first generate a perturbed version of it, and then enforce the model predictions on the two examples to be similar. A good model should be invariant to any small perturbations that do not change the nature of the example [34].

In this paper, we explore two methods of adversarial regularization. The first is adversarial training (AT) based on the fast gradient sign method (FGSM). FGSM was an effective approach to generate adversarial examples proposed by Goodfellow *et al.* [27]. Given a neural network, a training example and its target label, FGSM tries to generate an adversarial example which increases the difference between the given target label and the network output given the adversarial input. That is, it is trying to generate an adversarial example that has only a minor perturbation of the original input, but the current network parameters are blind to it. Note that FGSM needs label information while generating adversarial examples. The second method is virtual adversarial training (VAT) based on local-distributional-smoothness (LDS) [25], aiming at making the model output to be smooth/invariant with respect to input within a small region. VAT rewards the smoothness of the model distribution with respect to the input's neighborhood space. Unlike AT, the referenced LDS based VAT does not need label information during generation of adversarial perturbation.

Both AT and VAT use adversarial examples to add a regularization term in the objective function. Results in [27] and [25] show that both of them could significantly improve model robustness to adversarial examples in image classification task. We try both AT and VAT during LAS model training to make the model output distribution smoother, with respect to input sequences with minor perturbations. Different from image classification, LAS end-to-end speech recognition task is a sequential mapping task, where the input and output sequences have different lengths and the mapping has strong context dependency. Instead of generating adversarial perturbation frame-by-frame, we generate adversarial perturbation sequence for the whole utterance. Then we use the adversarial sequence to compute the regularization term in the objective function. Our experimental results based on 150-hour AISHELL-1 [35] and 1000-hour AISHELL-2 [36] Chinese Mandarin corpora show that our algorithms of adversarial regularization on LAS models can indeed significantly reduce character error rates (CER) for noisy and clean scenarios.

The rest of the paper is organized as follows: Section II introduces some prior work on end-to-end based speech recognition. Section III gives a brief review on the "listen, attend and spell" (LAS) end-to-end speech recognition paradigm. Section IV explains the basics of FGSM-based AT and the concept of LDS-based VAT. Section V elaborates how we extend AT and VAT to LAS models. Experiments and results are shown in Section VI, followed by concluding remarks in Section VII at the end.

II. Related Work

In this section, we will briefly introduce the related work on end-to-end speech recognition and robust acoustic modeling using adversarial learning.

A. End-to-End Speech Recognition

Graves *et al.* first proposed to tackle the problem of speech recognition using CTC [2] as an end-to-end modeling technique. Different from the HMM-DNN hybrid framework where the input and output lengths are the same, end-to-end systems do not expect input lengths to be identical to output lengths. For example, the input can be acoustic features, with one input vector per 10ms, while its CTC output can be a sequence of words, each of which corresponds to different segments of the input. CTC utilizes a special blank symbol which can be repeated in order to force the input sequence and the output sequence equal in length. The final output is the output sequence with blank symbols removed, hence achieving the effect that the output length does not have to be the same as the input length. CTC provides a simplified and effective end-to-end speech recognition architecture. The output units could be phonemes, characters [37], word pieces [38], or even words themselves. In order to extend CTC framework and overcome CTC's disvantages, Graves *et al.* proposed RNN transducer [39] to jointly model both input-output and output-output dependencies.

Another popular end-to-end modeling is the attention-based model, which originates from the encoder-decoder framework [40] proposed initially for machine translation. For speech recognition an encoder maps a sequence of acoustic features into high-level representations. Then it is followed by an attention-based [4] decoder to map these representations into a sequence of text labels. An attention mechanism between the encoder and decoder is used to learn alignment between acoustic representations and output labels. Transformer proposed in [41] is also a kind of attention based model and Zhou *et al.* proposed to do speech recognition task [42] using transformer. Chan et.al. proposed a model named LAS, which stands for Listen, Attend and Spell [5]. Recently, LAS showed a superior performance to a conventional hybrid system [6]. LAS architecture is also adopted by this paper. Details of LAS modeling will be described in Section III.

In order to solidify attention-based models, some work tries to combine CTC with attention [7], [8] to improve model robustness against noisy conditions. Li et al proposed to take advantage of dialect information explicitly during training[9] to improve LAS performance in multi-dialect scenarios. For conversational speech recognition, Weng *et al.* [43] proposed to improve LAS models by using input-feeding architecture [44] and sequential minimum Bayes risk (MBR) training [45]. Karita *et al.* [10] tried to take advantage of text data only (without corresponding audios) using adversarial training, to further improve attention based models. Guo *et al.* also proposed an approach to utilizing text-only data, by training a spelling correction model to explicitly correct recognition errors [46].

B. Robust Acoustic Modeling Using Adversarial Learning

Many adversarial learning techniques, such as GAN, DAT and adversarial examples, were also developed for robust AM training [13], [47]. DAT was initially proposed by Ganin *et al.* to tackle unsupervised domain adaptation for image classification. Later the idea was explored for speech recognition, in both supervised and unsupervised manners. DAT can enhance model robustness by learning domain-invariant hidden representations through a gradient reverse operation. The basic idea is similar to generative adversarial network (GAN) [12]. In speech recognition, DAT can be used to alleviate training-testing mismatch due to noise- [17], [48], [49], speaker- [18], [50], [51] and accent- [52], [53] differences. Adversarial examples can be used to attack speech recognition models, in order for the model to learn to be more robust. Although adversarial examples were initially invented for computer vision, researchers have adopted the methodology to construct audio adversarial examples to attack speech recognition systems, including end-to-end systems [29], [30], [54]. Our recent study has shown that, instead of attacking models, augmenting training data with adversarial examples could also make the acoustic model more robust in noisy environments [28], [55].

III. Listen, Attend and Spell (LAS)

LAS modeling accepts acoustic features as input and emits characters as output. Let $\boldsymbol{x} = (x_1, x_2, ..., x_T)$ be the input sequence of acoustic features, such as filter bank features, and let $\boldsymbol{y} = (\langle sos\rangle, y_1, y_2, ..., y_S, \langle eos\rangle)$ be the output sequence of characters. $\langle sos\rangle$ and $\langle eos\rangle$ are the start-of-sentence and end-of-sentence tokens, respectively. LAS models each character y_i as a conditional distribution over the previous characters y_1^{i-1} and the input feature $\boldsymbol{x}$ using the flowing chain rule:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_i P(y_i|\boldsymbol{x}, y_1^{i-1})$$

LAS modeling consists of two modules: the listener and the speller. The listener is an acoustic encoder, whose key operation is to "listen" and to transform the original input feature $\boldsymbol{x}$ into a high level representation $\boldsymbol{h} = (h_1, h_2, ..., h_T)$[1] The attention-based decoder, whose key operation is to "attend and spell out", is used to transform $\boldsymbol{h}$ into a probability distribution over character sequences:

$$\begin{aligned}\boldsymbol{h} &= \text{Listen}(\boldsymbol{x})\\ P(y_i|\boldsymbol{x}) &= \text{AttendAndSpell}(\boldsymbol{h}, y_1^{i-1})\end{aligned}$$

Fig. 1 shows the LAS model we used in this paper. For the listener, it could be any form of RNN, including LSTM and BLSTM. In [5], the authors adopted a pyramid BLSTM (pBLSTM) to reduce time resolution, layer by layer, and to reduce training time.

A. Attention

For the conventional HMM-DNN speech recognition framework, hard frame-level alignments between acoustic features and HMM states are often needed. For LAS modeling, the model learns the soft alignments through attention mechanism.

Specifically for each decoding step i to generate y_i, the attention mechanism creates a context vector c_i, using the high-level representations $\boldsymbol{h}$ and the decoder's previous hidden

[1] The length of the h_t sequence does not have to be identical to T. But we make it so, as most end-to-end systems do, for simplicity.

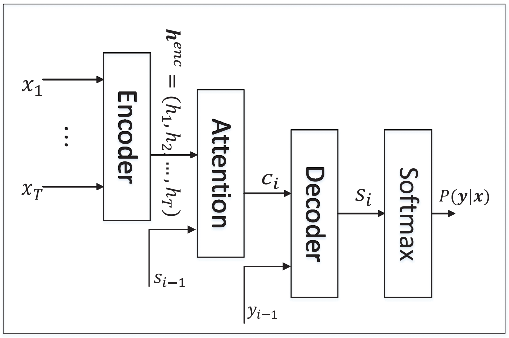

x_1
$\vdots$
x_T
Encoder
$\boldsymbol{h}^{enc} = (h_1, h_2, \ldots, h_T)$
Attention
s_{i-1}
c_i
Decoder
y_{i-1}
s_i
Softmax
$P(\boldsymbol{y}|\boldsymbol{x})$

Fig. 1. Listen, attend and spell (LAS) modeling: The encoder (listener) encodes input acoustic feature sequences into high-level representations. The decoder (speller) generates character sequences. An attention mechanism is used to generate soft alignments between high-level representations and output labels.

state s_{i-1}. To elaborate, we first calculate the similarity between the decoder's current hidden state s_{i-1} and each frame of the encoder, h_t:

$$e_{it} = w^{\intercal} \tanh(W s_{i-1} + V h_t + b)$$

where W and V are matrices and w and b are vectors with appropriate shapes, $\intercal$ represents vector transpose. They are all model parameters to be trained. e_{it} is a scalar similarity score. Then softmax is applied to get normalized scores, which measures the impact of h_t to the new output y_i (hence a soft alignment):

$$a_{it} = \frac{\exp(e_{it})}{\sum_{j=1}^{T} \exp(e_{ij})} \tag{1}$$

Finally, the context vector c_i is a weighted linear combination of the listener representation $\boldsymbol{h}$:

$$c_i = \sum_t a_{it} h_t \tag{2}$$

Notice both c_i and h_t are vectors.

B. Speller

In Fig. 1, decoder accepts the context vector c_i created by attention and the previously predicted (or ground truth) label y_{i-1} as inputs, and outputs the character distribution for the current step. During training, the ground truth label y_{i-1} is fed into the decoder, while during testing the predicted label is used. That said, one could possibly feed the predicted label as additional input during training [43], to possibly make the model more robust.

IV. Adversarial Regularization

In this section, we review two adversarial regularization techniques from prior work, adversarial training (AT) [27] and virtual adversarial training (VAT) [25]. The essence behind AT and VAT is the same: seek a "worst" spot around the current data point, and then optimize the model using this "worst" data we just found. AT and VAT can make model more robust to adversarial perturbation mentioned in Section I, making the output distribution smoother.

In [27] and [25], each data point x has a hard label y. Thus we first explain how AT and VAT worked with hard alignments. Then in Section V we will explain how we extend them to LAS-based soft alignments in end-to-end speech recognition.

A. Adversarial Examples

As described in Section I, the goal of adversarial examples is to disturb well-trained models in order to make them more robust to small variations in input. We first give a formal definition of adversarial examples.

A neural network is a parameterized function, $f(x; \boldsymbol{\theta})$, where x is the input (usually a multi-dimensional vector) and $\boldsymbol{\theta}$ represents the model parameters. A trained model $f(x; \boldsymbol{\theta})$ is used to predict label y corresponding to input x. An adversarial example related to data point x can be constructed as

$$\hat{x} = x + \delta \tag{3}$$

so that

$$y \neq f(\hat{x}; \boldsymbol{\theta})$$

where

$$\|\delta\| \ll \|x\| \tag{4}$$

and δ is called the adversarial perturbation. Given a well-trained model, if we can generate adversarial examples and "fool" the model easily, it means the model output is un-smooth (un-robust) with respect to the input. Once obtaining the adversarial examples, we can penalize the model's sensitivity with respect to the perturbation in the adversarial direction.

B. Adversarial Training (AT)

In order to generate adversarial examples efficiently, Goodfellow *et al.* [27] proposed a gradient-based approach, named fast gradient-sign method (FGSM). During model training, given input x and corresponding label y, model parameters $\boldsymbol{\theta}$ are trained to minimize the objective function $J(x, y; \boldsymbol{\theta})$ by using optimization methods such as the stochastic gradient descent (SGD) method. Cross entropy is often used as $J(x, y; \boldsymbol{\theta})$ for classification tasks. The objection function serves as a cost measure, and the optimization goal is to minimize the cost. According to the definition of adversarial examples, our goal is to generate a new example $\hat{x}$ ($\approx x$) which increases the value of the cost function $J(\hat{x}, y; \boldsymbol{\theta})$. FGSM provides a method to generate adversarial perturbation:

$$\delta_F = \epsilon \,\mathrm{sign}(\nabla_x J(x, y; \boldsymbol{\theta})) \tag{5}$$

where ϵ is a small constant. Notice the derivative is taken with respect to the input x. Only the direction of the gradient (sign) is used here because it would be easy to satisfy the constraint of Eq. (4). Then the adversarial example can be constructed using Eq. (3).

Fig. 2 illustrates the generation of FGSM-based adversarial examples. This is considered as a supervised generation, because

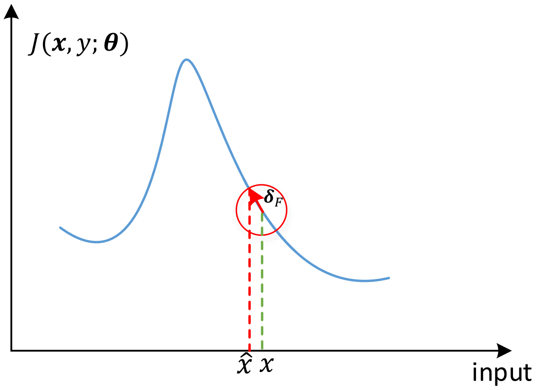


Fig. 2. Generation of adversarial examples via FGSM. It climbs up against the gradient to find adversarial examples.

it needs the true label y. After we generate an adversarial example, the neural network can then be trained using a regularized objective function:

$$J_{AT}(x, y; \boldsymbol{\theta}) = J(x, y; \boldsymbol{\theta}) + \alpha J(\hat{x}, y; \boldsymbol{\theta}) \tag{6}$$

where $\alpha > 0$. Training the neural network with this objective function is called the adversarial training (AT).

C. Virtual Adversarial Training (VAT)

Given the fact that images and time series occurring in nature tend to be smooth with respect to time, a robust neural network should also be smooth with respect to the input. Miyato *et al.* [25] proposed a concept named local distribution smoothness (LDS), which rewards the smoothness of the model distribution with respect to the input around every training data point. LDS is defined as the sensitivity of the model distribution $p(x, \boldsymbol{\theta})$ with respect to the perturbation of x, measured in the sense of KL divergence. The formal definition of LDS is as follows:

$$\Delta(\delta, x, \boldsymbol{\theta}) = \mathbf{KL}[\, p(x, \boldsymbol{\theta}) \| p(x + \delta, \boldsymbol{\theta})] \tag{7}$$

$$\delta_V = \arg\max_{\delta} \{\Delta(\delta, x, \boldsymbol{\theta}) \text{ where } \|\delta\|_2 \leq \epsilon\} \tag{8}$$

where $\Delta(\delta, x, \boldsymbol{\theta})$ is the KL divergence between the model distributions before and after δ perturbation. ϵ is a small positive constant and δ_V is called the "virtual adversarial" perturbation for data x. It is a small perturbation, but it gives you the biggest different output distribution in the neighborhood of x. LDS is simply the negative of KL divergence:

$$\text{LDS}(x, \boldsymbol{\theta}) = -\Delta(\delta_V, x, \boldsymbol{\theta}) \tag{9}$$

The smaller the KL divergence is, the smoother the local neighborhood is (bigger LDS).

The goal is to improve the smoothness of the model in the neighborhood of all the observed inputs. Therefore the regularized objective function becomes:

$$J_{\text{VAT}}(x, y; \boldsymbol{\theta}) = J(x, y; \boldsymbol{\theta}) - \alpha \text{LDS}(x, \boldsymbol{\theta}) \tag{10}$$

where $\alpha > 0$.

Because δ_V in Eq. (8) is the direction to which the model distribution $p(x, \boldsymbol{\theta})$ is the most sensitive, the smaller the value of $\Delta(\delta_V, x, \boldsymbol{\theta})$ is, the smoother the $p(x, \boldsymbol{\theta})$ distribution is at x.

Miyato *et al.* [24], [25] gave an iterative method to estimate δ_V. Because $\Delta(\delta_V, x, \boldsymbol{\theta})$ equals to 0 at $\delta = 0$, the differentiability assumption means $\nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \mid_{\delta=0}$ is zero. Therefore, we can take the second-order Taylor approximation as

$$\Delta(\delta, x, \boldsymbol{\theta}) \approx \frac{1}{2} \delta^T H(x, \theta) \delta \tag{11}$$

where $H(x, \theta)$ is a Hessian matrix defined by $H(x, \theta) = \nabla\nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \mid_{\delta=0}$. Under this approximation, δ_V emerges as the first dominant eigenvector $u(x, \theta)$ of Hessian matrix $H(x, \theta)$, of magnitude ϵ,

$$\begin{aligned} \delta_V &\approx \arg\max_{\delta} \{\delta^T H(x, \theta) \delta \text{ where } \|\delta\|_2 \leq \epsilon\} \\ &= \epsilon \overline{u(x, \theta)} \end{aligned} \tag{12}$$

where $\overline{u}$ returns a unit vector of u. The eigenvectors of $H(x, \theta)$ require $O(I^3)$ computational time, which becomes unfeasibly large for high dimensional input space, where I is the dimensional of input. Therefore, power iteration method [56] and finite difference method are applied to approximate δ_V, which leads to an iterative estimation method. Starting from a random unit vector d, as long as d is not perpendicular to the dominant eigenvector $u(x, \theta)$, the iterative calculation of

$$d \leftarrow \overline{H(x, \theta) d}$$

will make d converge to $u(x, \theta)$. $H(x, \theta)d$ can be approximated by finite difference without the direct computation of $H(x, \theta)$:

$$\begin{aligned} H(x, \theta) d &\approx \frac{\nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \mid_{\delta=\xi d} - \nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \mid_{\delta=0}}{\xi} \\ &= \frac{\nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \mid_{\delta=\xi d}}{\xi} \end{aligned}$$

with $\xi \neq 0$. Finally, d can be approximated with the repeated application of the following update rule:

$$d \leftarrow \overline{\nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \mid_{\delta=\xi d}}$$

and δ_V could be obtained after using the estimated d

$$\delta_V = \epsilon d \tag{13}$$

The algorithm is depicted in Algorithm 1, where Iters=1 usually can get a good result in practice.

Training model using Eq. 10 as the objective function is called "virtual adversarial training" (VAT). Notice unlike AT, the label y is not needed while generating $\hat{x}$, nor computing the regularization term in J_{VAT}.

D. AT vs. VAT

AT and VAT are similar in many ways. First, both of them attempt to make the model more robust against adversarial perturbation by adding an adversarial regularization term into the loss function. Second, the regularization term is related to the "worst input" around the current training example. Making the model work better in the "worst case" could definitely increase

Algorithm 1: Iterative Estimation of δ_V for Input x.

```
Initialize a random unit vector $d$ in input space
for ($i = 0$; $i <$ Iters; $i$++) do
    $d \leftarrow \overline{\nabla_\delta \Delta(\delta, x, \boldsymbol{\theta}) \,|_{\delta=\xi d}}$
end for
$\delta_V = \epsilon\, d$
return $\delta_V$
```

model robustness. Finally they both take advantage of the local input-output relationship to smooth model output distributions in a small neighborhood.

The main differences are on how adversarial perturbation δ is generated and how to use the generated adversarial examples. In AT, the adversarial perturbation is generated in a supervised way using FGSM, and it tries to minimize the distance between the model prediction given $\hat{x}$ and the ground-truth label. It does not measure the local smoothness. On the other hand, VAT does not know and does not care what the ground-truth label is. It just tries to smooth out the neighborhood. Hence the term "virtual" is used. Generating perturbation δ_V does not need ground truth.

V. Adversarial Regularization for LAS

Similar to other end-to-end speech recognition framework, LAS models acoustic and linguistic information simultaneously. Hence environmental noises could be more easily propagated to the model output than non end-to-end systems. Furthermore the model's current output distribution depends on previous outputs strongly. So the errors in previous outputs could also mislead the current output. Compared with conventional hybrid approaches, LAS could very likely suffer from the input-output un-smooth issue more seriously. Although there is a lot of work which focuses on improving LAS model robustness as mentioned in Section I, no work has tried to improve LAS from the viewpoint of model smoothness using adversarial training, to our knowledge. This paper attempts to understand if adversarial training could make LAS models more invariant to small variations in input.

In Section IV, AT and VAT are introduced given frame-by-frame hard alignment. In sequence-to-sequence modeling such as speech-to-text transcription, input $\boldsymbol{x} = (x_1 x_2\ ...\ x_T)$ and output $\boldsymbol{y} = (\langle sos \rangle y_1 y_2\ ...\ y_S \langle eos \rangle)$ are both sequences and often of different lengths. Here we elaborate how we extend AT and VAT to sequence based training. Our goal is to generate a perturbed input sequence $\hat{\boldsymbol{x}} = (\hat{x}_1 \hat{x}_2\ ...\ \hat{x}_T)$, given $\boldsymbol{x}$. Notice our perturbed sequence has the same length as the original input sequence.

A. Finding FGSM-Based Adversarial Perturbation for LAS Modeling

For FGSM-based adversarial perturbation given one training utterance $\boldsymbol{x}$, all gradients back propagated from all output steps $\boldsymbol{y}$ are accumulated. After $\langle eos \rangle$ is output, we finally generate the adversarial example $\hat{\boldsymbol{x}} = (\hat{x_1} \hat{x_2}\ ...\ \hat{x_T})$ and compute the regularized objective function as follows, for AT:

$$\delta_F(x_t) = \epsilon \text{ sign}\left(\nabla_{x_t} \sum_i J(\boldsymbol{x}, y_i; \boldsymbol{\theta})\right) \tag{14}$$

$$\hat{x}_t = x_t + \delta_F(x_t)$$

$$\hat{\boldsymbol{x}} = (\hat{x}_1\ \hat{x}_2\ ...\ \hat{x}_T)$$

$$J_{AT}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = \sum_i \left(J(\boldsymbol{x}, y_i; \boldsymbol{\theta}) + \alpha J(\hat{\boldsymbol{x}}, y_i; \boldsymbol{\theta})\right) \tag{15}$$

Notice that the length of $\hat{\boldsymbol{x}}$ is the same as $\boldsymbol{x}$, and they both generate the same output sequence $\boldsymbol{y}$. The error signals in Eq. 15 are accumulated across all output steps y_i for $\boldsymbol{x}$, and then all output steps for $\hat{\boldsymbol{x}}$, and further accumulated for an entire mini-batch. Finally model parameters are updated per mini-batch. Note that Eq. 15 is similar to the previous adversarial example based data augmentation approach in [28], in which for every mini-batch, the model was trained using $J(\boldsymbol{x}, y_i; \boldsymbol{\theta})$ first, and then Eq. 14 was used to generate δ_F and the model was trained again using $J(\hat{\boldsymbol{x}}, y_i; \boldsymbol{\theta})$. Similar to hard-alignment AT, we also require the ground truth $\boldsymbol{y}$ for LAS models to calculate adversarial perturbation.

B. Finding Virtual Adversarial Perturbation for LAS Modeling

For VAT, we can not apply Algorithm 1 directly to our LAS model because LAS is a sequence-to-sequence model. We need to calculate the optimal perturbation sequence $\boldsymbol{\delta}_V = (\delta_1^* \delta_2^*\ ...\ \delta_T^*)$ for an input sequence of length T. Algorithm 2 modifies Algorithm 1 to generate virtual adversarial perturbation for sequence-to-sequence models. Note that at Step 4 and 5 of Algorithm 2, the length of $\boldsymbol{P}$ and $\boldsymbol{Q}$ is $S + 1$ because $\langle eos \rangle$ is considered ($y_{S+1} = \langle eos \rangle$).

After generating virtual adversarial perturbation, $\boldsymbol{\delta}_V = (\delta_1^* \delta_2^* ... \delta_T^*)$, the LDS for each output step y_i is the negative of the KL divergence between the output distribution (p_i^*) at output step i given the original input sequence, and the output distribution (q_i^*) at output step i given the perturbed input sequence $\hat{\boldsymbol{x}} = (x_1 + \delta_1^*, x_2 + \delta_2^*, ..., x_T + \delta_T^*)$, as indicated by Eq. 16:

$$\text{LDS}_i(\boldsymbol{x}, \boldsymbol{\theta}) = -\mathbf{KL}\,[\, p_i^* \| q_i^* \,]. \tag{16}$$

The loss function at output step i is then

$$J_{\text{VAT}}(\boldsymbol{x}, y_i; \boldsymbol{\theta}) = J(\boldsymbol{x}, y_i; \boldsymbol{\theta}) - \alpha\, \text{LDS}_i(\boldsymbol{x}, \boldsymbol{\theta}). \tag{17}$$

Similar to the FGSM-based AT, the length of $\hat{\boldsymbol{x}}$ is the same as $\boldsymbol{x}$. The loss in Eq. 17 is accumulated across $(\boldsymbol{x}, \hat{\boldsymbol{x}})$ per batch, and then model parameters are updated.

Unlike hard-alignment VAT, we now do need ground truth $\boldsymbol{y}$ at Step 5 of Algorithm 2 to generate adversarial perturbations.

C. Training LAS Models With Adversarial Regularization

Algorithm 3 outlines LAS model training, integrated with adversarial regularization. For every mini-batch training, we need to feed the data into the neural network first. For the first N epochs, only cross-entropy loss with the original data is used, in order to get a reliable model for generating adversarial examples later. For both AT and VAT, they need a relatively

Algorithm 2: Iterative Estimation of $\boldsymbol{\delta}_V$ for LAS Models, Given Input Sequence $\boldsymbol{x} = (x_1 x_2 \ldots x_T)$ and Output Sequence $\boldsymbol{y} = (\langle sos \rangle y_1 y_2 \ldots y_S, \langle eos \rangle)$.

```
1:  Initialize a random unit vector sequence
    $\boldsymbol{d} = (d_1 d_2 \ldots d_T)$ in input space, where
    $\|d_t\|_2 = 1\ \forall t = 1..T$
2:  $\boldsymbol{\delta} = (\delta_1 \delta_2 \ldots \delta_T) = \xi \boldsymbol{d} = (\xi d_1\ \xi d_2 \ldots \xi d_T)$
3:  for ($i = 0$; $i <$ Iters; $i$++) do
4:      Feed $\boldsymbol{x}$ into LAS model and force output sequence
        $\boldsymbol{y}$. Denote the output distribution at each output step
        $i$ as $p_i$ and $\boldsymbol{P} = (p_1 p_2 \ldots p_S p_{S+1})$.
5:      Feed $\hat{\boldsymbol{x}} = (x_1 + \delta_1, x_2 + \delta_2, \ldots, x_T + \delta_T)$ into LAS
        model and force output sequence $\boldsymbol{y}$ as well. Denote
        the output distribution at each output step $i$ as $q_i$
        and $\boldsymbol{Q} = (q_1 q_2 \ldots q_S q_{S+1})$.
6:      $\Delta(\boldsymbol{\delta}, \boldsymbol{x}, \boldsymbol{\theta}) = \sum_{i=1}^{S+1} \mathbf{KL}\,[\,p_i \| q_i\,]$
7:      $d_t \leftarrow \overline{\nabla_{\delta_t} \Delta(\boldsymbol{\delta}, \boldsymbol{x}, \boldsymbol{\theta})\,|_{\delta_t = \xi d_t}},\ \forall t = 1...T$
8:      $\boldsymbol{d} = (d_1 d_2 \ldots d_T)$
9:      $\boldsymbol{\delta} = \xi \boldsymbol{d}$
10: end for
11: $\boldsymbol{\delta}_V = (\delta_1^* \delta_2^* \ldots \delta_T^*) = \epsilon \boldsymbol{d}$
12: return $\boldsymbol{\delta}_V$
```

Algorithm 3: Training LAS Models Using AT or VAT.

```
Initialize model parameters $\boldsymbol{\theta}$, let epoch=0
Given hyper parameters
    • $\epsilon$ for Eq. (14) and Algorithm 2 Line 11.
    • $\alpha$ for Eq. (15) and (17)
    • Iters and $\xi$ for Algorithm 2
    • $p_a$, adversarial probability
    • $N \geq 1$, starting epoch number for AT or VAT
Prepare training data $\{\boldsymbol{x}_k, \boldsymbol{y}_k\}_{k=1}^{I}$
while not converge do
    Get a mini-batch training data $B = \{\boldsymbol{x}_m, \boldsymbol{y}_m\}_{m=1}^{M}$
    Forward propagate the network using data $B$
    if epoch $> N$ && use AT && random() $< p_a$ then
        Generate $\hat{B}$ using Eq. (14) from $B$
        Train $\boldsymbol{\theta}$ using Eq. (15) with $B \cup \hat{B}$
    else if epoch $> N$ && use VAT && random() $< p_a$
    then
        Generate $\hat{B}$ using Algorithm 2 from $B$
        Train $\boldsymbol{\theta}$ using Eq. (17) with $B \cup \hat{B}$
    else
        Train $\boldsymbol{\theta}$ using cross-entropy (CE) loss with $B$
    end if
    epoch = epoch + 1
end while
return $\boldsymbol{\theta}$
```

reliable model to calculate gradients with respect to input data, regardless of using hard alignments in [27] and [25], or using soft alignments in LAS models. Furthermore, because AT and VAT can introduce extra computational cost, training model only using the CE loss at the first N epochs can reduce training time without performance degradation. On the other hand, this strategy can also provide a well-trained initial model for AT and VAT training. In our experiments, we found that the choice of N was not crucial for the final results.

We also introduce an adversarial probability p_a during AT and VAT training, which is used to control the proportion of training data used for AT and VAT. In Algorithm 3, random() returns a random float number in range [0,1). When $p_a < 1$, AT or VAT is randomly applied to the training data. When $p_a = 1$, all training data will be used for AT or VAT. For AT and VAT, the main extra computational cost comes from adversarial perturbation calculation. For FGSM-based AT, only one extra back-propagation is needed to calculate δ_F in Eq. 14. For LDS-based VAT, one forward propagation is needed to calculate $\boldsymbol{Q}$ in line 5, Algorithm 2 and one back propagation is needed for d_t calculation in line 7, Algorithm 2. In summary, it needs no more than three pairs of forward and back propagation operations for VAT and more details can be found in [25].

D. Comparison With Data Augmentation Using Adversarial Examples

Training augmentation using adversarial examples was first proposed in our recent study [28] for DNN-HMM acoustic model training, which can be briefly summarized as follows:

- Train the model using a mini-batch of training data, B, using CE loss function and back propagation;
- Freeze model parameters and generate a batch of adversarial examples, $\hat{B}$ using FGSM or LDS;
- Resume the training using the generated adversarial examples and CE loss function.

Similar to Algorithm 3, at the first N epochs, the model is trained on the original data only. Starting from Epoch $N + 1$, the data augmentation strategy trains on B followed by $\hat{B}$ for each batch, always using CE as the objective function.

In [28], we only tried FGSM-based training augmentation. In the experiments in Section VI, LDS-based training augmentation is also studied as a comparison. Actually, FGSM-based augmentation and FGSM-based regularization are similar in nature because both $J(\boldsymbol{x}, y_i; \boldsymbol{\theta})$ and $J(\hat{\boldsymbol{x}}, y_i; \boldsymbol{\theta})$ in Eq. 15 are CE loss function between neural network's output and target; the only difference lies in the training strategies. Comparing with data augmentation (first generate data, and the train the model), regularization technique generates the data and training the model simultaneously.

As a contrast, LDS-based regularization is apparently different from the augmentation idea. Although both LDS-based augmentation and regularization use Algorithm 2 to find the perturbation $\boldsymbol{\delta}_V$, the regularization approach explicitly considers LDS as part of the loss function to improve model smoothness in Eq. 17, instead of using the CE loss function to measure the distribution difference between the perturbed output and the ground truth target.

TABLE I
AISHELL-1 MANDARIN SPEECH CORPUS. HF REPRESENTS HIGH-FIDELITY RECORDING

	#utters	#speakers	#hrs	channels
training	120,098	340	150	HF
dev	14,326	40	10	HF
test	7,176	20	5	HF

TABLE II
AISHELL-2 MANDARIN SPEECH CORPUS. IP=IPHONE, AP = ANDROID PHONE

	#utters	#speakers	#hrs	channels
training	1,008,834	1347	1000	iP
dev	2,500x3	5	6	HF, iP, aP
test	5,000x3	10	12	HF, iP, aP

VI. EXPERIMENTS

A. Datasets

Our experiments are conducted based on two Mandarin Chinese corpora: AISHELL-1 and AISHELL-2. Both of them are public available and one can access and download them online.[2,3] The two corpora were originally recorded in quiet environments. In order to test model robustness to noises, we further add noises to all data sets, by ranging signal-to-noise (SNR) between 5 db and 20 db.

1) AISHELL-1: AISHELL-1 were recorded via a high-fidelity (HF) microphone, an android phone (aP) and an iPhone (iP), but only high-fidelity data was released. The recorded audio was re-sampled into 16 kHz and 16-bit WAV format. 400 speakers participated in the recording. The text transcriptions are chosen from 11 domains, including finance, science and technology, sports, entertainment and so on. The training set contains 120,098 utterances from 340 speakers, for a total of 150 hours; the development set contains 14,326 utterances from 40 speakers, for a total of 10 hours; the test set contains 7,176 utterances from 20 speakers, for a total of 5 hours. Table I summarizes AISHELL-1. More details can be found in [35].

2) AISHELL-2: For AISHELL-2, the recording devices are the same as AISHELL-1. For the training data, only iPhone recording was released. There are 1991 speakers participated in the recording. The recording environments include a recording studio and a living room with natural reverberation without extra noises. The text transcriptions cover 12 domains, including keywords, voice commands, points of interest, entertainment, finance, free speaking without specific topics, etc. The training set contains 1000-hour iPhone recorded utterances; the development set consists of data from 3 acoustic channels: iPhone, android phone and high-fidelity microphone, with 2500 utterances per channel from 5 speakers; the test set also contains 3 acoustic channels, with 5000 utterances per channel from 10 speakers. Table II summarizes AISHELL-2 corpus. Compared with AISHELL-1, AISHELL-2 covers more topics and speaker diversities and it is a more practical Mandarin Chinese corpus.

3) Noisy Data Simulation: AISHELL-1 and AISHELL-2 were recorded in clean environments. In order to make model more robust, a widely adopted approach is to simulate noisy training data using clean speech and extra noises. In this work, because we mainly focus on noise robust speech recognition, we also simulate noisy training and test data sets. Musan [57] corpus is used as our noise data source. It consists of music, speech, and noises, in total about 109 hours. For every utterance, we randomly choose a noise file from Musan corpus with an SNR value randomly from {5, 10, 15, 20} db. We construct noisy training sets based on AISHELL-1 and AISHELL-2 using this strategy, which are then used to train LAS models in this work. However for dev and test sets, we report error rates on both clean and noisy data sets. We will see even on clean test data, out proposed methods can also increase the robustness of the model.

B. LAS Model Details

We adopt a relative simple network structure. For the encoder, 2-layer BLSTM is used. There are 256 LSTM units in each direction of each layer. For the decoder, one LSTM layer with 512 units is used. Several effective training strategies are adopted during model training:

- Input-feeding [43] is used. It has been proven to be effective in improving LAS model performance.
- Dropout rate of 0.1 is applied to avoid overfitting.
- Small noises are added to model weights to further avoid overfitting.
- Schedule sampling [58] is adopted to alleviate training-testing mismatch.
- SpecAugment [59] from Google is applied to model training. We found that SpecAugment can give at most 2% absolute CER reduction in our experiments, thus leading to a strong LAS baseline for comparison.

As for SpecAugment, we adopt the time and frequency masking strategies proposed in [59]. Specifically, for every training utterance, time or/and frequency masking is applied to the original fbank feature. Only masked features are used to train the model as suggested in [59]. Thus SpecAugment will not increase the amount of training data.

Our acoustic feature is 80-dim filter banks with delta and delta-delta, for a total of 240 dimensions per input frame. Our output units are Chinese characters. For AISHELL-1 experiments, 4223 frequently-used characters with three extra tokens $\langle sos\rangle$, $\langle eos\rangle$ and $\langle unk\rangle$ are used. For AISHELL-2 experiments, 5257 characters are modeled. During decoding, beam search without any extra language model is used.

C. Experimental Results

In this section, we will show our results on AISHELL-1 and AISHELL-2. As a contrast, apart from the adversarial regularization methods, we also report the results of adversarial data augmentation proposed by our previous work [28]. Experimental results show that adversarial regularization is more effective than adversarial augmentation for LAS modeling.

There are several hyper-parameters needed to be tuned, such as ϵ and α. We use the development sets to tune these hyper-parameters and the best hyper-parameters are then applied to the test sets directly. In all of our VAT experiments, we set

[2] http://www.aishelltech.com/kysjcp
[3] http://www.aishelltech.com/aishell_2

TABLE III
Speech Recognition Results on the AISHELL-1 Dev Sets With Data Augmentation: Adversarial Examples are Generated via Random Perturbation, FGSM-Based Perturbation and LDS-Based Perturbation

Training	p_a	ϵ	α	CER / Relative improvement (%)	
				Clean (dev)	Noisy (dev)
Baseline (Clean)	-	-	-	10.65 / -	31.09 / -
Baseline (Noisy)	-	-	-	11.10 / -	14.09 / -
RAND-AUG	1.0	0.3	1.0	10.39 / 6.4	13.28 / 5.7
FGSM-AUG	1.0	0.15	1.0	10.25 / 7.7	12.76 / 9.4
	1.0	0.15	0.3	10.30 / 7.2	12.94 / 8.2
LDS-AUG	1.0	0.15	0.3	10.45 / 5.9	13.04 / 7.5
	1.0	0.3	0.3	10.53 / 5.1	13.21 / 6.2

TABLE IV
Speech Recognition Results on AISHELL-1 Test Sets Using Adversarial Data Augmentation. The Hyper-Parameters are Selected Based on the Tuning on the Development Sets

Training	p_a	ϵ	α	CER / Relative improvement (%)	
				Clean (test)	Noisy (test)
Baseline (Clean)	-	-	-	12.12 / -	35.17 / -
Baseline (Noisy)	-	-	-	12.68 / -	16.41 / -
RAND-AUG	1.0	0.15	1.0	11.95 / 5.8	15.53 / 5.4
FGSM-AUG	1.0	0.15	1.0	11.60 / 8.5	14.82 / 9.7
LDS-AUG	1.0	0.15	0.3	11.91 / 6.1	15.07 / 8.2

TABLE V
Speech Recognition Results on AISHELL-1 Development Sets Using Adversarial Regularization

Training	p_a	ϵ	α	CER / Relative improvement (%)	
				Clean (dev)	Noisy (dev)
Baseline (Clean)	-	-	-	10.65 / -	31.09 / -
Baseline (Noisy)	-	-	-	11.10 / -	14.09 / -
RAND-REG	1.0	0.3	1.0	11.20 / -0.9	14.30 / -1.5
FGSM-REG	0.5	0.1	0.3	10.21 / 8.0	12.77 / 9.4
	0.5	0.15	0.3	10.42 / 6.1	13.33 / 5.4
	1.0	0.15	0.3	10.36 / 6.7	13.1 / 7.0
LDS-REG	1.0	0.15	1.0	10.28 / 7.4	12.79 / 9.2
	1.0	0.3	1.0	9.43 / 15.0	11.75 / 16.6
	1.0	0.5	1.0	9.99 / 10.0	12.73 / 9.7

TABLE VI
Speech Recognition Results on AISHELL-1 Test Sets Using Adversarial Regularization. The Hyper-Parameters are Selected Based on the Tuning on Development Sets

Training	p_a	ϵ	α	CER / Relative improvement (%)	
				Clean (test)	Noisy (test)
Baseline (Clean)	-	-	-	12.12 / -	35.17 / -
Baseline (Noisy)	-	-	-	12.68 / -	16.41 / -
RAND-REG	1.0	0.3	1.0	12.65 / 0.2	16.52 / -0.7
FGSM-REG	0.5	0.1	0.3	11.68 / 7.9	15.01 / 8.5
LDS-REG	1.0	0.3	1.0	**10.56 / 16.7**	**13.45 / 18.0**

$\xi = 10$ [25] and Iters=1. In addition, in order to verify our proposed adversarial perturbation is more effective than random perturbation, we also conduct experiments based on random perturbation. Specifically, instead of generating perturbation using FGSM in Eq. (14) for AT or LDS in Algorithm 2 for VAT, in random case, we use a random unit vector to replace sign$(\nabla_{x_t} \sum_i J(\boldsymbol{x}, y_i; \boldsymbol{\theta}))$ in Eq. (14) and d_t in Algorithm 2.

1) Data Augmentation Using Adversarial Examples: As mentioned in Section V-D, data augmentation using adversarial is closely related to this work. Thus we conduct experiments using augmentation strategy as well. To assess the utility of data augmentation using adversarial examples, we also compare with data augmentation using small random perturbation instead of adversarial perturbation. These data augmentation methods are denoted as FGSM-AUG, LDS-AUG, and RAND-AUG respectively. They all use double amount of the original training data for training when $p_a = 1$.

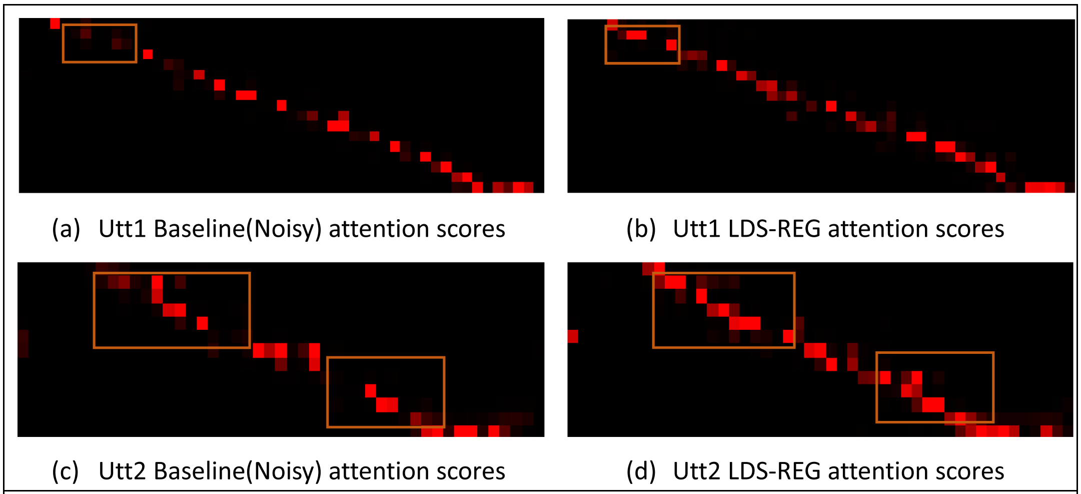
Table III shows the results on the AISHELL-1 development sets with different hyper-parameter combinations. We have tried many hyper-parameter combinations and we are reporting only the representative results in this table. Note that there are two baseline models in Table III, which are Baseline (Clean) and Baseline (Noisy). Both baseline models are trained using cross entropy loss function and without any augmentation nor regularization technique. Baseline (Clean) is trained using clean training data and Baseline (Noisy) is trained using simulated noisy training data. The performance of Baseline (Clean) on noisy dev set is pretty bad. Therefore our remaining models are trained using the simulated noisy training data exclusively. Table IV shows the results on the AISHELL-1 test sets with the best hyper-parameter combination tuned on the dev set. Compared with RAND-AUG and LDS-AUG, FGSM-AUG gives much more improvements on both dev and test sets. We believe that in the data augmentation manner, Algorithm 2 with $Iters = 1$ can not generate sufficient adversarial perturbations.

2) Adversarial Regularization: Table V shows results on AISHELL-1 development sets using adversarial regularization, where the adversarial examples are generated randomly, based on FGSM, or based on LDS. First, we notice that random perturbation based regularization makes the model slightly worse than the noisy baseline model, which indicates that random perturbation fails to make the model more smooth. Compared to adversarial perturbation based regularization that targets to the *weak point* of current model, random perturbation does not make use of any model- or data-specific knowledge and the perturbation is somewhat un-targeted. Thus such kind of regularization can not regularize the model effectively. FGSM-REG is just marginally better than FGSM-AUG in Table III. LDS-REG is the most effective way among all these approaches, giving 17.2% and 18.1% relative CER reduction on clean and noisy development sets respectively. When augmentation strategy is adopted, for every mini-batch training data, model parameters need to be updated twice. In contrast, when the regularization strategy is used, because the loss generated by adversarial examples is used as a regularization term in the loss function (Eq. (15) and Eq. (17)), model parameters only need to be updated once for every mini-batch data, increasing training efficiency.

Table VI shows the results on AISHELL-1 test sets with the best hyper-parameter combination. LDS-REG gives 16.7% and 18.0% relative CER reduction on clean and noisy test sets respectively. Similar in nature as we mentioned in Section V-D, both FGSM-AUG and FGSM-REG obtain similar gains. On the contrast, LDS-REG is superior than LDS-AUG, indicating that explicitly introducing LDS into the loss function is more effective.

TABLE VII
SPEECH RECOGNITION RESULTS ON AISHELL-2 TEST SETS USING ADVERSARIAL DATA AUGMENTATION AND ADVERSARIAL REGULARIZATION

Training	p_a	ϵ	α	Test set CER / Relative improvement (%)					
				Android Clean	Android Noisy	iPhone Clean	iPhone Noisy	HF Clean	HF Noisy
Baseline(Noisy)	-	-	-	10.70	13.09	9.87	11.89	11.42	14.74
FGSM-AUG	1.0	0.15	1.0	10.47 / 2.1	12.50 / 4.5	9.73 / 1.4	11.59 / 2.5	11.17 / 2.2	14.41 / 2.2
LDS-AUG	1.0	0.15	0.3	10.09 / 5.7	12.10 / 7.6	9.41 / 4.7	11.32 / 4.8	10.87 / 4.8	14.25 / 3.3
FGSM-REG	1.0	0.15	0.3	10.62 / 0.7	12.81 / 2.1	9.72 / 1.5	11.68 / 1.8	11.25 / 1.5	14.57 / 1.2
LDS-REG	1.0	0.3	1.0	**9.67 / 9.6**	**11.49 / 12.2**	**9.18 / 7.0**	**10.74 / 9.7**	**10.28 / 10.0**	**13.44 / 8.8**
Chain-TDNN [37]	-	-	-	9.59	-	8.81	-	10.87	-


(a) Utt1 Baseline(Noisy) attention scores

(b) Utt1 LDS-REG attention scores

(c) Utt2 Baseline(Noisy) attention scores

(d) Utt2 LDS-REG attention scores

Utt1 Reference	当在新闻发布会上被问及伤情的时候
(a) Utt1 Baseline (Noisy)	当来新闻发布会上被问及伤情的时候
(b) Utt1 LDS-REG	当在新闻发布会上被问及伤情的时候
Utt2 Reference	据记者从多个渠道了解
(c) Utt2 Baseline (Noisy)	自竞者出多个渠道而言
(d) Utt2 LDS-REG	据记者从多个渠道了解

Fig. 3. Visualization of attention scores on two utterances from the noisy AISHELL-1 test set. (a) and (c) are by the AISHELL-1 Baseline (Noisy) model, while (b) and (d) are by the AISHELL-1 LDS-REG model. The bottom part shows the reference transcriptions and recognized hypotheses by the two models. Red underlined characters indicate recognition errors.

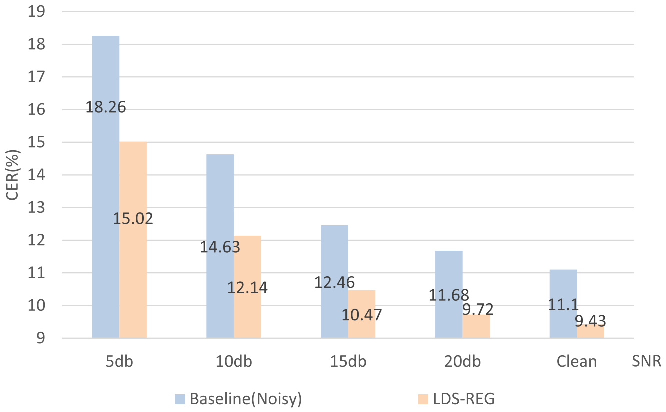
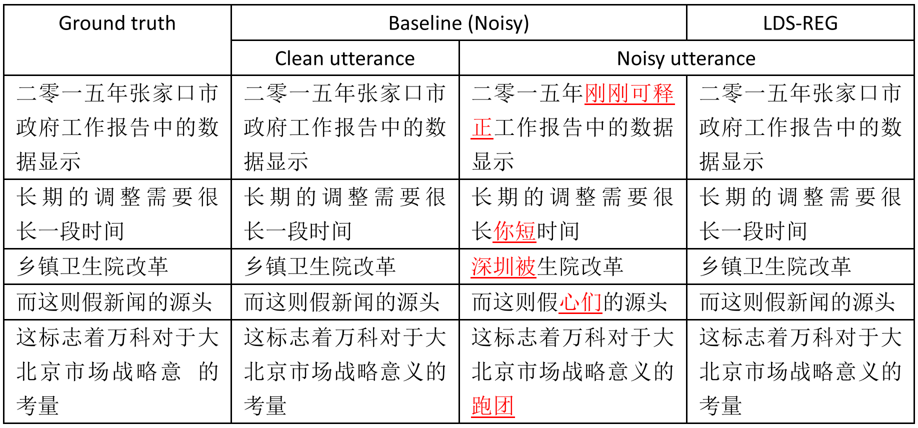


Fig. 4. CER and relative improvement of Baseline (Noisy) and LDS-REG models on AISHELL-1 noisy dev set under various SNRs.

3) Results on AISHELL-2: AISHELL-1 only has 150 hours of training data. We expand our experiments to a bigger data set, which is AISHELL-2. As mentioned previously, AISHELL-2 has about 1000 hours training data and covers many more topics than AISHELL-1. Furthermore, AISHELL-2 also provides three test sets with different microphone channels, while the released training data was recorded by iPhone exclusively. AISHELL-2 is a more general and practical corpus. The results on AISHELL-2 test sets are shown in Table VII. Note that training on 1000-hour AISHELL-2 is time-consuming so we used the best hyper-parameter combinations from the AISHELL-1 experiments directly on this large corpus. Furthermore, for AISHELL-2 experiments, parallel training with bigger mini-batch, i.e., 120 utterances per mini-batch (4 GPUs, 30 utterances per GPU) is used to accelerate training. We find that the LAS model can converge much better with a bigger mini-batch.



Ground truth	Baseline (Noisy)		LDS-REG
	Clean utterance	Noisy utterance	
二零一五年张家口市政府工作报告中的数据显示	二零一五年张家口市政府工作报告中的数据显示	二零一五年刚刚可释正工作报告中的数据显示	二零一五年张家口市政府工作报告中的数据显示
长期的调整需要很长一段时间	长期的调整需要很长一段时间	长期的调整需要很长你短时间	长期的调整需要很长一段时间
乡镇卫生院改革	乡镇卫生院改革	深圳被生院改革	乡镇卫生院改革
而这则假新闻的源头	而这则假新闻的源头	而这则假心们的源头	而这则假新闻的源头
这标志着万科对于大北京市场战略意 的考量	这标志着万科对于大北京市场战略意义的考量	这标志着万科对于大北京市场战略意义的跑团	这标志着万科对于大北京市场战略意义的考量

Fig. 5. Some recognized results of clean and corresponding noisy utterances on Baseline (Noisy) and LDS-RED models respectively from AISHELL-1 dev set.

Table VII indicates that the contribution of FGSM-REG and LDS-REG shrinks as the amount of training data increases. With a bigger training set, the variety of data increases and hence model robustness and smoothness get improved as well. FGSM-AUG and FGSM-REG give similar and minor improvements, while LDS-REG is still the most effective approach to improving model's robustness on bigger corpus, with relative 7.0-12.2% CER reduction. The last row in Table VII shows the Chain-TDNN results on three clean test sets recently reported in [36]. We can see that as compared with the popular Chain-TDNN model, our LDS-REG trained end-to-end model can achieve comparable and even better results (on HF Clean test set) without using any extra language model.

D. Analysis

The above results show the effectiveness of adversarial regularization for LAS model training. We visualize the attention scores a_{it} (Eq. 1) in Fig. 3. In Fig. 3, the left two figures ((a), (c)) are learned attention alignments from the AISHELL-1 "Baseline (Noisy)" model. The alignments from the Baseline model are discontinuous and fuzzy in the highlighted regions. The right two figures ((b), (d)) are the corresponding attention alignments from the AISHELL-1 LDS-REG model. It is obvious that the alignments learned from LDS-REG are more continuous and unambiguous. As we know, alignment learned from the attention layer is crucial for LAS modeling, as it represents the correlation between text characters and acoustic features. A better model should learn better alignment from training data.

Fig. 3 also shows the recognized hypotheses by the Baseline vs. the LDS-REG model. Red characters indicate recognition errors. Due to the error propagation of LAS model we have mentioned in Section I, there are many continuous substitution errors in the recognized results, such as Utt2 in Fig. 3. The LDS-REG model performs much better than the Baseline, which is consistent with the quality of learned attention scores.

Fig. 4 shows CER and relative improvement of Baseline (Noisy) and LDS-REG models on AISHELL-1 noisy dev set under various SNRs. From Fig. 4 we can find that the relative improvement is inversely proportional to the SNR roughly. LDS-REG can obtain 17.7% relative improvements on the most noisy (5 db) test set, which indicates that our proposed adversarial regularization approach can enhance model's robustness to noise interferences. Even on the clean test set, our proposed LDS-REG approach can still achieve 15.0% relative improvement.

Fig. 5 shows some recognized results of clean utterances and the noisy counterpart on Baseline (Noisy) and LDS-RED models respectively from AISHELL-1 dev set. From these examples, we can easily find that the Baseline model performs well under clean condition while poorly under noisy condition. Our proposed LDS-REG model is much more robust to noise and performs well even under noisy condition.

VII. Conclusion

This paper modified two adversarial regularization techniques for attention based end-to-end speech recognition systems. We elaborated how to generate adversarial training examples for LAS models, with FGSM-based AT and LDS-based VAT techniques. Comparing with adversarial data augmentation, adversarial regularization in the objective function is far more powerful in improving model robustness. Experimental results on two Mandarin corpora showed up to 18.0% and 12.2% relative character error rate reduction on AISHELL-1 and AISHELL-2 noisy test sets. Other testing conditions indicated similar improvements. Training with adversarial regularization could also improve the alignments between acoustic features and output characters.

Acknowledgment

The authors would like to thank T. Miyato for constructive discussions and C. Shan's help during LAS model training.

References

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[2] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 369–376.

[3] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 6645–6649.

[4] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, “End-to-end attention-based large vocabulary speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2016, pp. 4945–4949.

[5] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2016, pp. 4960–4964.

[6] C.-C. Chiu *et al.*, “State-of-the-art speech recognition with sequence-to-sequence models,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4774–4778.

[7] S. Kim, T. Hori, and S. Watanabe, “Joint CTC-attention based end-to-end speech recognition using multi-task learning,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2017, pp. 4835–4839.

[8] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, “Hybrid CTC/attention architecture for end-to-end speech recognition,” *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 8, pp. 1240–1253, Dec. 2017.

[9] B. Li *et al.*, “Multi-dialect speech recognition with a single sequence-to-sequence model,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4749–4753.

[10] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, “Semi-supervised end-to-end speech recognition,” in *Proc. Interspeech*, 2018, pp. 2–6.

[11] T. Hori, W. Wang, Y. Koji, C. Hori, B. Harsham, and J. R. Hershey, “Adversarial training and decoding strategies for end-to-end neural conversation models,” *Comput. Speech Lang.*, vol. 54, pp. 122–139, 2019.

[12] I. Goodfellow *et al.*, “Generative adversarial nets,” in *Proc. Advances Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[13] Y. Ganin *et al.*, “Domain-adversarial training of neural networks,” *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.

[14] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by back-propagation,” in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.

[15] K. Wang, J. Zhang, S. Sun, Y. Wang, F. Xiang, and L. Xie, “Investigating generative adversarial networks based speech dereverberation for robust speech recognition,” in *Proc. Interspeech*, 2018, pp. 1581–1585.

[16] S. Pascual, A. Bonafonte, and J. Serrà, “Segan: Speech enhancement generative adversarial network,” in *Proc. Interspeech*, 2017, pp. 3642–3646.

[17] S. Sun, B. Zhang, L. Xie, and Y. Zhang, “An unsupervised deep domain adaptation approach for robust speech recognition,” *Neurocomputing*, vol. 257, pp. 79–87, 2017.

[18] Z. Meng *et al.*, “Speaker-invariant training via adversarial learning,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 5969–5973.

[19] Z. Meng, J. Li, Y. Gong, and B.-H. Juang, “Adversarial teacher-student learning for unsupervised domain adaptation,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 5949–5953.

[20] W. Hsu *et al.*, “Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 5901–5905.

[21] S. Yang *et al.*, “Statistical parametric speech synthesis using generative adversarial networks under a multi-task learning framework,” in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2017, pp. 685–691.

[22] C. M. Bishop *et al.*, “Pattern recognition and machine learning,” Springer, 2006.

[23] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in *Artificial Intelligence Safety and Security*. Chapman & Hall, 2018, pp. 99–112.

[24] T. Miyato, S. Maeda, S. Ishii, and M. Koyama, “Virtual adversarial training: A regularization method for supervised and semi-supervised learning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1979–1993, Aug. 2019.

[25] T. Miyato, S. Maeda, M. Koyama, K. Nakae, and S. Ishii, “Distributional smoothing with virtual adversarial training,” *ICLR*, 2016.

[26] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, “Generating natural language adversarial examples,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 2890–2896.

[27] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples (2014),” *ICLR*, 2014.

[28] S. Sun, C.-F. Yeh, M. Ostendorf, M.-Y. Hwang, and L. Xie, “Training augmentation with adversarial examples for robust speech recognition,” in *Proc. Interspeech*, 2018, pp. 2404–2408.

[29] N. Carlini and D. Wagner, “Audio adversarial examples: Targeted attacks on speech-to-text,” in *Proc. Deep Learn. Sec. Workshop*, 2018, pp. 1–7.

[30] M. M. Cisse, Y. Adi, N. Neverova, and J. Keshet, “Houdini: Fooling deep structured visual and speech recognition models with adversarial examples,” in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 6977–6987.

[31] M. Alzantot, B. Balaji, and M. Srivastava, “Did you hear that? adversarial examples against automatic speech recognition,” 2018, *arXiv:1801.00554*.

[32] C. M. Bishop, “Pattern recognition and machine learning.”

[33] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, “Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 7893–7897.

[34] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, “Unsupervised data augmentation,” 2019, *arXiv preprint arXiv:1904.12848*.

[35] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, “Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline,” in *Proc. 20th Conf. Oriental Chapter Int. Coordinating Committee Speech Databases Speech I/O Syst. Assessment*, 2017, pp. 1–5.

[36] J. Du, X. Na, X. Liu, and H. Bu, “Aishell-2: Transforming mandarin ASR research into industrial scale,” 2018, *arXiv:1808.10583*.

[37] Y. Miao, M. Gowayyed, and F. Metze, “Eesen: End-to-end speech recognition using deep RNN models and WFST-based decoding,” in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2015, pp. 167–174.

[38] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, “Advancing acoustic-to-word CTC model,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 5794–5798.

[39] A. Graves, “Sequence transduction with recurrent neural networks,” 2012, *arXiv:1211.3711*.

[40] K. Cho *et al.*, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1724–1734.

[41] A. Vaswani *et al.*, “Attention is all you need,” in *Proc. Advances Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[42] S. Zhou, L. Dong, S. Xu, and B. Xu, “Syllable-based sequence-to-sequence speech recognition with the transformer in mandarin chinese,” in *Proc. Interspeech*, 2018, pp. 791–795.

[43] C. Weng *et al.*, “Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition,” in *Proc. Interspeech*, 2018, pp. 761–765.

[44] T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1412–1421.

[45] R. Prabhavalkar *et al.*, “Minimum word error rate training for attention-based sequence-to-sequence models,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4839–4843.

[46] J. Guo, T. N. Sainath, and R. J. Weiss, “A spelling correction model for end-to-end speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 5651–5655.

[47] D. Yu and J. Li, “Recent progresses in deep learning based acoustic models,” *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 3, pp. 396–409, Jul. 2017.

[48] Y. Shinohara, “Adversarial multi-task learning of deep neural networks for robust speech recognition,” in *Proc. Interspeech*, 2016, pp. 2369–2372.

[49] D. Serdyuk, K. Audhkhasi, P. Brakel, B. Ramabhadran, S. Thomas, and Y. Bengio, “Invariant representations for noisy speech recognition,” 2016, *arXiv:1612.01928*.

[50] Q. Wang, W. Rao, S. Sun, L. Xie, E. S. Chng, and H. Li, “Unsupervised domain adaptation via domain adversarial training for speaker recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4889–4893.

[51] Y. Adi, N. Zeghidour, R. Collobert, N. Usunier, V. Liptchinsky, and G. Synnaeve, “To reverse the gradient or not: An empirical comparison of adversarial and multi-task learning in speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 3742–3746.

[52] S. Sun, C. Yeh, M. Hwang, M. Ostendorf, and L. Xie, “Domain adversarial training for accented speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4854–4858.

[53] A. Tripathi, A. Mohan, S. Anand, and M. Singh, “Adversarial learning of raw speech features for domain invariant speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 5959–5963.

[54] H. Yakura and J. Sakuma, “Robust audio adversarial example for a physical attack,” 2018, *arXiv:1810.11793*.

[55] X. Wang *et al.*, "Adversarial examples for improving end-to-end attention-based small-footprint keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 6366–6370.

[56] G. H. Golub and H. A. Van der Vorst, "Eigenvalue computation in the 20th century," in *Numerical Analysis: Historical Developments in the 20th Century*. Amsterdam, The Netherlands: Elsevier, 2001, pp. 209–239.

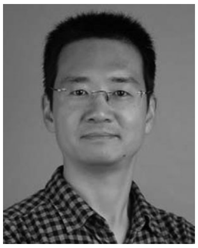
[57] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," 2015, *arXiv:1510.08484*.

[58] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances Neural Inf. Process. Syst.*, 2015, pp. 1171–1179.

[59] D. S. Park *et al.*, "Specaugment: A simple data augmentation method for automatic speech recognition," 2019, *arXiv:1904.08779*.

Sining Sun (S'16) received the B.Eng. degree from Northwestern Polytechnical University (NPU), Xi'an, China, in 2014. He is currently working toward the Ph.D. degree in computer science and technology at the Audio, Speech, and Language Processing Group, Shaanxi Provincial Key Laboratory of Speech and Image Information Processing, School of Computer Science, NPU. In 2016, he was a research student with the School of Computer Science and Engineering, Nanyang Technological University. In 2017, he visited the University of Washington and Mobvoi AI Lab in Seattle as a research intern. His current research interests include robust and far-field automatic speech recognition, adversarial learning.

Pengcheng Guo received the B.Eng. degree from Northwestern Polytechnical University (NPU), Xi'an, China, in 2016. He is currently working toward the Ph.D. degree in computer science and technology at the Audio, Speech, and Language Processing Group, Shaanxi Provincial Key Laboratory of Speech and Image Information Processing, School of Computer Science, NPU. In 2017, he was a research student with the School of Computer Science and Engineering, Nanyang Technological University. In 2019, he visited the ByteDance AI Lab as an intern. His current research interests include robust automatic speech recognition and end-to-end speech recognition.



Lei Xie (M'07−SM'15) received the Ph.D. degree in computer science from Northwestern Polytechnical University (NPU), Xi'an, China, in 2004. From 2001 to 2002, he was with the Department of Electronics and Information Processing, Vrije Universiteit Brussel, Brussels, Belgium, as a Visiting Scientist. From 2004 to 2006, he was a Senior Research Associate with the Center for Media Technology, School of Creative Media, City University of Hong Kong. From 2006 to 2007, he was a Postdoctoral Fellow with the Human Computer Communications Laboratory, Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong. He is currently a Professor with the School of Computer Science, NPU. He has authored and coauthored more than 200 papers in major journals and proceedings, such as IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, PATTERN RECOGNITION, Interspeech, and ICASSP. His current research interests include general areas of speech and language processing, e.g., speech recognition, enhancement and synthesis.

Mei-Yuh Hwang (F'19), received the Ph.D. degree in computer science from Carnegie Mellon University, Pittsburgh, PA, USA, in 1993. She was with the Microsoft Seattle and China for 18 years, publishing numerous conference and journal papers, and delivering industry products in speech recognition, machine translation, and language understanding. She is an affiliate professor with the University of Washington, and is the director of Mobvoi AI Lab in Seattle WA, since early 2016. Her research interests include bridging the gap between academia and industry.