Improving Speech Emotion Recognition with Adversarial Data Augmentation Network

Lu YI and Man-Wai MAK, *Senior Member, IEEE*

***Abstract*—When training data is scarce, it is challenging to train a deep neural network without causing the overfitting problem. For overcoming this challenge, this paper proposes a new data augmentation network – namely adversarial data augmentation network (ADAN) – based on generative adversarial networks (GANs). The ADAN consists of a GAN, an autoencoder, and an auxiliary classifier. These networks are trained adversarially to synthesize class-dependent feature vectors in both the latent space and the original feature space, which can be augmented to the real training data for training classifiers. Instead of using the conventional cross-entropy loss for adversarial training, Wasserstein divergence is used in an attempt to produce high-quality synthetic samples. The proposed networks were applied to speech emotion recognition using EmoDB and IEMOCAP as the evaluation datasets. It was found that by forcing the synthetic latent vectors and the real latent vectors to share a common representation, the gradient vanishing problem can be largely alleviated. Also, results show that the augmented data generated by the proposed networks are rich in emotion information. Thus, the resulting emotion classifiers are competitive with state-of-the-art speech emotion recognition systems.**

Index Terms: Speech emotion recognition, data augmentation, generative adversarial networks, Wasserstein divergence.

I. Introduction

In recent years, deep learning has made remarkable progress in many areas, such as speech recognition [1], image recognition [2], and genomics [3]. Generally, deep learning models are complex and require a large amount of data to achieve accurate predictions or classifications. Unfortunately, the data collection process is often expensive and time-consuming, which makes acquiring labeled data a big challenge. This problem is particularly acute in speech emotion recognition because an utterance may contain ambiguous or multiple emotions. Multiple annotators are often employed to label the utterances in speech emotion corpora to increase the annotation reliability. Nevertheless, in some cases, even professional annotators may not be unanimous in their decisions [4]. Therefore, it is important to address the data sparsity problem.

Emotion recognition plays a key role in natural human-computer interaction [5], [6]. Traditional speech emotion recognition systems consist of a feature extractor in the front-end and a classifier at the back-end. For the latter, hidden Markov models (HMMs) and Gaussian mixture models (GMMs) have been used to classify the instantaneous and global features extracted from the front-end [7]. Another approach is to use prosodic features to train a support vector machine (SVM) for classification [8]. However, these hand-crafted features may not be optimal for emotion recognition. Hand-crafted features are not robust in that their performance is highly dependent on the evaluation set.

With the development of deep learning, using deep neural networks to extract features gradually replaces manual feature engineering [9]. In particular, the convolutional neural networks (CNNs) and the long short-term memory recurrent neural networks (LSTM-RNN) have been used to exploit the dynamic structure of frame-based features [10], [11]. Motivated by the success of convolutional neural networks in image classification [12], more and more researchers applied CNNs to extract features from spectrograms [13–15]. For instance, Huang *et al.* [13] proposed a CNN-based feature learning method to extract emotion-salient features that are invariant to nuisance factors. Zhao *et al.* [15] combined CNNs with recurrent neural networks (RNNs) to extract language information from spectrograms. Luo *et al.* [16] proposed an HSF-CRNN system that combines the hand-crafted high-level statistic functional (HSF) features and the features learnt by a convolutional recurrent neural network (CRNN). A similar strategy was proposed in [17] in which a DNN was trained to extract emotion features from hand-crafted features and a CNN was trained to extract emotion features from spectrograms. The combined features were classified by an extreme learning machine.

Recent work tends to apply an end-to-end scheme to tackle the speech emotion recognition tasks [18–21]. Typically, spectrograms and class labels are respectively used as the input and output of the end-to-end systems. In [19], feature maps produced by convolutional filters are divided into time-specific and frequency-specific. To assign higher weights to emotion-related parts in the spectrogram, top-down attention and bottom-up attention were applied to the last convolutional layer.

With the widespread applications of deep learning in emotion recognition, many effective solutions to the data sparsity problem have been investigated [22–25]. Transfer learning [26] is a popular solution to the insufficient-data problem. In particular, domain adaptation (a subset of transfer learning) can leverage labeled data from the source domain to learn a model for the unlabeled data in the target domain. Motivated by the achievements in image classification [27], this technique has been gradually applied to speech emotion recognition as well. For example, Deng *et al.* [22] explored a feature transfer learning method in which source-domain data are transformed to the target-domain through a sparse autoencoder trained from the target-domain training data. Transformed data from the source domain are then used for training an emotion classifier. The authors found that the classifier trained by the transformed data can significantly improve the performance of emotion classification on the target dataset, even if the target dataset is small. In addition to transferring knowledge within the same

This work was in part supported by the Research Grants Council of Hong Kong, Grant No. PolyU152137/17E and PolyU152518/16E.

task (emotion recognition), another study demonstrates that transferring from speaker recognition to emotion recognition could also help improve emotion classification [28].

The introduction of generative adversarial networks (GANs) [29] creates new possibilities for tackling the insufficient data problem. A typical GAN consists of a generator and a discriminator. Both are neural networks that act like two players competing with each other in a zero-sum game. The generator learns to map an arbitrary distribution to the data distribution to confuse the discriminator, and the discriminator is trained to distinguish whether a sample comes from the data distribution (i.e., genuine) or from the generator (i.e., fake). Some researches show that GAN-based data augmentation techniques can help improve the performance of image recognition [30]. Zhang *et al.* [31] proposed an improved GAN to generate high-dimensional representations and showed that GAN-based data augmentation outperforms conventional data augmentation techniques.

In [32], we proposed an adversarial data augmentation network (ADAN) that combines an autoencoder with a GAN to perform data augmentation. The ADAN not only overcomes the gradient vanishing problem that often occurs in vanilla GANs but also produces real-like samples that share common latent representation with the real data. In this paper, we will further explain the effectiveness of ADAN and extend [32] by replacing the adversarial loss with Wasserstein divergence. Unlike [32], we used the whole IEMOCAP dataset and used a newer feature set in this paper. Moreover, in-depth analyses have been added. These analyses include comparisons of different types of GANs, distances between the distributions of real and synthetic data, the effect of varying the number of augmented samples, and the efficiency of the proposed models. Our experimental results demonstrate that the proposed data augmentation approach can further improve the recognition performance on the EmoDB [33] and IEMOCAP [4] datasets.

The rest of this paper is organized as follows. Section II presents a review of common solutions to the data sparsity problem. Section III describes the design of the proposed network and provides a theoretical analysis. Section IV introduces the details of the experiments, including descriptions of data, features, experimental setup, and evaluation protocol. Section V presents and analyses the experimental results. Finally, Section VI presents the conclusions and future work.

II. Related Work

Data sparsity could cause a machine learning model not able to learn the true data distribution, which leads to the overfitting problem. For example, overfitting would occur when training a deep model with only hundreds of samples but each sample has thousands of features. To solve this problem, regularization can be used to impose constraints on the model [34]. Another general solution is dimension reduction with sparsity constraint [35]. This approach will be effective when redundant features exist; otherwise it will eliminate useful information and result in performance degradation.

To solve the data sparsity problem, the training set can be enlarged by data augmentation. Traditional data-augmentation methods typically transform (e.g., adding noise and reverberation to speech signals and cropping, rotating, and flipping of images) the original data, followed by augmenting the transformed data to the original data [36]. More advanced methods augment the data based on GANs or variants of GANs, such as conditional GANs (cGANs) or adversarial autoencoders (AAEs). Hu *et al.* [24] used a very deep CNN to generate additional feature maps for training acoustic models and found that the augmented data help build robust speech recognition systems. Sahu *et al.* [23] synthesized feature vectors through an AAE by using a mixture of Gaussian distributions as the random source. Though the synthetic samples could help to improve classification performance, the synthetic samples tend to follow an arbitrary distribution rather than the actual data distribution. Sahu *et al.* [25] also built a cGAN-based model to create synthetic feature vectors. Several training tricks, such as initializing the generator with the weights from the decoder of the AAE [23] and updating the weights of the generator several times before updating the discriminator in each training epoch, have been applied to train the cGAN.

One major difficulty in training GANs is to ensure a balance between the capability of the generator and the discriminator. To overcome this difficulty, dynamic alternation training [31] can be applied. This training strategy is to dynamically change the number of training epochs between the generator and the discriminator rather than fixing it. Instead of optimizing the number of training epochs, our proposed network aims to improve the learning stability by facilitating the generator to learn the target distribution. Specifically, the generator in our proposed network learns the distribution of a latent representation produced by a simultaneously trained encoder rather than learning a pre-defined distribution.

III. Methodology

A. Generative Adversarial Nets

A basic GAN comprises a generator that generates real-like data from random samples and a discriminator that attempts to differentiate the generated data from the real ones. Given a set of random samples $\mathbf{z}$'s from a probability distribution $p(\mathbf{z})$, the generator G transforms the samples to mimic the distribution of real data $\mathbf{x}$ and makes the discriminator believe that the generated samples $G(\mathbf{z})$'s are real. Meanwhile, the discriminator D tries to distinguish the true samples $\mathbf{x}$'s from the fake samples $G(\mathbf{z})$'s. These objectives can be expressed as follows [29]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left\{\log D\left(\mathbf{x}\right)\right\} + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\left\{\log\left(1-D\left(G\left(\mathbf{z}\right)\right)\right)\right\}. \quad (1)$$

In practice, rather than training the generator G to minimize $\log\left(1-D\left(G\left(\mathbf{z}\right)\right)\right)$, we can train G to maximize $\log(D(G(\mathbf{z})))$, as suggested in [29]. This objective function can provide stronger gradients that help to overcome the gradient vanishing problem without changing the equilibrium point that the generator G and the discriminator D reach. With this new objective function, D and G are trained to minimize the losses defined in Eq. 2 and Eq. 3.

$$\mathcal{L}_D^{(GAN)} = -\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{\log D(\mathbf{x})\} - \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log(1-D(G(\mathbf{z})))\} \quad (2)$$

$$\mathcal{L}_G^{(GAN)} = -\mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log(D(G(\mathbf{z})))\}. \quad (3)$$

B. Conditional Generative Adversarial Nets

Conditional GAN is an extension of the vanilla GAN in that it considers extra information $\mathbf{y}$, such as class labels or other forms of data. The generator receives the concatenation of $\mathbf{y}$ and random vector $\mathbf{z}$ as input, and the discriminator receives the concatenation of real data $\mathbf{x}$ and extra information $\mathbf{y}$ as input. Thus, the objective function of cGAN is [37]

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{\log D([\mathbf{x},\mathbf{y}])\} + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log(1-D([G([\mathbf{z},\mathbf{y}]),\mathbf{y}]))\}, \quad (4)$$

where $[\mathbf{x},\mathbf{y}]$ means concatenating vectors $\mathbf{x}$ and $\mathbf{y}$.

Similar to the vanilla GAN, the discriminator and the generator are trained to minimize the losses defined below:

$$\mathcal{L}_D^{(cGAN)} = -\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{\log D([\mathbf{x},\mathbf{y}])\} - \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log(1-D([G([\mathbf{z},\mathbf{y}]),\mathbf{y}]))\} \quad (5)$$

$$\mathcal{L}_G^{(cGAN)} = -\mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log(D([G([\mathbf{z},\mathbf{y}]),\mathbf{y}]))\}. \quad (6)$$

C. Adversarial Autoencoders

An adversarial autoencoder (AAE) comprises an encoder, a decoder, and a discriminator. The encoder plays a role similar to the generator in GANs. However, instead of generating fake samples as in GANs, the encoder in AAEs aims to match an aggregated posterior to an arbitrary prior [38]. In addition to the adversarial learning objective, the encoder E and the decoder R are also trained to minimize the reconstruction error:

$$\mathcal{L}_D^{(AAE)} = -\mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log D(\mathbf{z})\} - \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{\log(1-D(E(\mathbf{x})))\} \quad (7)$$

$$\mathcal{L}_E^{(AAE)} = -\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{\log(D(E(\mathbf{x})))\} \quad (8)$$

$$\mathcal{L}_R^{(AAE)} = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{||\mathbf{x}-R(E(\mathbf{x}))||^2\}, \quad (9)$$

where $\mathbf{x}$ represents the encoder's input and $\mathbf{z}$ represents either the encoder's output or a sample from the prior distribution $p_z(\mathbf{z})$.

D. Adversarial Data Augmentation Network

Fig. 1 shows the structure of the adversarial data augmentation network (ADAN) proposed in [32]. It comprises an autoencoder $R(E(\mathbf{x}))$, an auxiliary classifier $C(E(\mathbf{x}))$, a generator $G(\mathbf{z},\mathbf{y})$ and a discriminator $D(\mathbf{h})$. The ADAN is designed to achieve three goals. First, it learns a latent representation that retains emotion information. Second, it attempts to match the posterior distribution $p(\hat{\mathbf{h}}|\mathbf{z},\mathbf{y})$ to the posterior distribution $p(\mathbf{h}|\mathbf{x})$. Third, it minimizes the reconstruction errors between $\mathbf{x}$ and $\hat{\mathbf{x}}$. The three components in Fig. 1 are trained adversarially to achieve these objectives. Specifically, the encoder E and the classifier C are trained to learn the M-dimensional latent

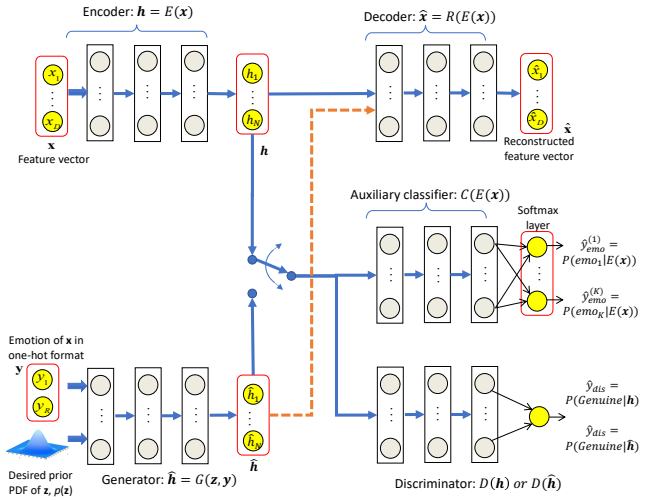


Fig. 1. The structure and data flow of an adversarial data augmentation network (ADAN). The network comprises an autoencoder with an auxiliary classifier (top), a generator (lower-left) and a discriminator (lower-right). The individual subnetworks are DNNs. The dotted line is only used for data augmentation after training.

representations $\mathbf{h}$'s that are highly emotion-discriminative. Simultaneously, the decoder learns to reconstruct emotion vectors in the original space from the latent representations. The generator takes samples drawn from an M-dimensional Gaussian distribution and one-hot encoded emotion labels as input and generates samples in the latent space; its goal is to generate samples that are indistinguishable from the real samples in the latent space, i.e., $p(\mathbf{h}|\mathbf{x}) \approx p(\hat{\mathbf{h}}|\mathbf{z},\mathbf{y})$. The discriminator is optimized to distinguish whether a latent vector comes from the real data or from the generator. The advantage of generating samples in the latent space instead of the original space is that generation of high-dimensional vectors can be avoided.

To train the proposed network, we minimize the losses defined below:

$$\mathcal{L}_D^{(\text{ADAN})} = -\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{\log D(E(\mathbf{x}))\} - \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\{\log(1-D(G(\mathbf{z},\mathbf{y})))\} \quad (10)$$

$$\mathcal{L}_C^{(\text{ADAN})} = -\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left\{\sum_{k=1}^{K} y_{emo}^{(k)} \log C(E(\mathbf{x}))_k\right\} \quad (11)$$

$$\mathcal{L}_R^{(\text{ADAN})} = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\{||\mathbf{x}-R(E(\mathbf{x}))||^2\} \quad (12)$$

$$\mathcal{L}_E^{(\text{ADAN})} = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left\{||\mathbf{x}-R(E(\mathbf{x}))||^2 - \sum_{k=1}^{K} y_{emo}^{(k)} \log C(E(\mathbf{x}))_k\right\} \quad (13)$$

$$\mathcal{L}_G^{(\text{ADAN})} = \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}\left\{\log(1-D(G(\mathbf{z},\mathbf{y}))) - \alpha\sum_{k=1}^{K} y_{emo}^{(k)} \log C(G(\mathbf{z},\mathbf{y}))_k\right\} \quad (14)$$

where $()_k$ denotes the k-th element of a vector, G stands for the generator, R for the decoder, E for the encoder, D for the discriminator and C for the auxiliary classifier. α determines

the contributions of the classification error to the loss in the generator.

E. Wasserstein ADAN

Wasserstein GANs [39] have been proposed to overcome the gradient vanishing problem. Given two probability distributions, $\mathbb{P}_r$ and $\mathbb{P}_g$, the Wasserstein distance is defined as:

$$\mathcal{W}_1\left(\mathbb{P}_r, \mathbb{P}_g\right) = \sup_{\|f\|_L \leq 1} \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathbb{P}_r}\left\{f(\mathbf{x})\right\} - \mathop{\mathbb{E}}_{\tilde{\mathbf{x}}\sim\mathbb{P}_g}\left\{f(\tilde{\mathbf{x}})\right\}, \quad (15)$$

where $\|f\|_L \leq 1$ indicates that f satisfies the 1-Lipschitz constraint. Weight clipping and gradient penalty are two common approaches to impose the 1-Lipschitz constraint. However, according to [40], weight clipping could narrow the search space of function f and lead to a sub-optimal solution. To overcome the limitations of weight clipping, the gradient penalty was introduced [41]. However, under data-sparsity conditions, it is difficult to satisfy the k-Lipschitz constraint for the entire data domain. With these considerations, Wu *et al.* [40] proposed a novel Wasserstein divergence that can approximate the Wasserstein distance without imposing the Lipschitz constraint. It is defined as follows:

$$L_{\mathrm{DIV}} = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathbb{P}_r}\left\{f(\mathbf{x})\right\} - \mathop{\mathbb{E}}_{\tilde{\mathbf{x}}\sim\mathbb{P}_g}\left\{f(\tilde{\mathbf{x}})\right\} + \lambda \mathop{\mathbb{E}}_{\breve{\mathbf{x}}\sim\mathbb{P}_u}\left\{\|\nabla f(\breve{\mathbf{x}})\|^p\right\}, \quad (16)$$

where $\mathbb{P}_u$ is a Radon probability measure, λ controls the impact of the gradient term on the objective function, and p corresponds to the L^p space for function f. In addition, λ and p must satisfy $\lambda > 0$ and $p > 1$ to ensure that L_{DIV} in Eq. 16 is a symmetric divergence, which has been proved in [40].

Incorporating Eq. 16 into ADAN, the losses for the discriminator and the generator become

$$\mathcal{L}_D^{(\mathrm{WADAN})} = \mathbb{E}_{p(\mathbf{x},\mathbf{z},\breve{\mathbf{x}},\mathbf{y})}\Big\{D(E(\mathbf{x})) - D(G(\mathbf{z},\mathbf{y})) + \lambda\left[\|\nabla_{\breve{\mathbf{x}}} D(\breve{\mathbf{x}})\|^p\right]\Big\} \quad (17)$$

$$\mathcal{L}_G^{(\mathrm{WADAN})} = \mathbb{E}_{p(\mathbf{x},\mathbf{y},\mathbf{z})}\Big\{D(G(\mathbf{z},\mathbf{y})) - \alpha\sum_{k=1}^{K} y_{emo}^{(k)} \log C(G(\mathbf{z},\mathbf{y}))_k\Big\}. \quad (18)$$

For other losses, they are the same as Eq. 11 – Eq. 13. The network structure of Wasserstein ADAN (WADAN) is also the same as Fig. 1. The difference between ADAN and WADAN is that the final layer of the discriminator in the former uses the sigmoid activation function, whereas linear activation is applied to the final layer of the discriminator in WADAN.

After the training of ADAN or WADAN, we connected the generator G to the decoder R (the dotted arrow in Fig. 1) for data augmentation. By inputting the one-hot emotion labels and Gaussian random vectors $\mathbf{z}$ to the generator, synthetic samples can be obtained from the output of the decoder. More details of the augmentation will be described in Section IV.

F. Advantages of ADANs and WADANs

ADANs use the Jensen-Shannon divergence (JS-divergence) as the divergence measure. According to [39], the JS-divergence would be a constant if the support sets of two distributions have little or no overlap, which leads to the gradient vanishing problem. Our network structure can overcome this issue as discussed in our previous work [32]. At the early stage of training, the distributions of $\mathbf{h}$ and $\hat{\mathbf{h}}$ are largely overlapped, which causes difficulty for the discriminator to differentiate these two groups of latent vectors. Therefore, the discriminator will produce high cross-entropy loss, and the generator will receive non-zero error gradient. On the other hand, the cross-entropy loss arising from the classifier C to the generator G through the second term of Eq. 14 can help to avoid gradient vanishing. This means that even if the gradient of the first term in Eq. 14 is zero, we still have the gradient of the second term to update G.

In addition to this advantage, ADANs can easily take the advantages of other divergence measures such as Wasserstein divergence (Eq. 17 and Eq. 18) to overcome the gradient vanishing problem. Compared to JS-divergence, the advantage of Wasserstein divergence is that it can measure the distance between two distributions even if they do not overlap. The latent space created by ADANs also makes the learning of emotion information easier and faster because of its low dimension. In addition, many applications [41–43] have shown that the generative models with Wasserstein divergence are superior to those with other divergence measures, such as JS-divergence and maximum mean discrepancy. Therefore, it is believed that WADAN can generate more meaningful emotion vectors, and our experiments in Section IV also demonstrate this.

IV. Experiments

A. Datasets

The experiments in this paper were conducted on the Berlin Database of Emotional Speech (EmoDB) [33] and the Interactive Emotional Dyadic Motion Capture (IEMOCAP) database [4].

EmoDB is a tiny dataset that comprises 535 utterances divided into seven emotion classes. All utterances were spoken by ten actors.

IEMOCAP contains the utterances of ten actors participating in dyadic interactions. The data can be divided into improvised sessions and scripted sessions. In this study, we considered two situations: four emotions – *angry*, *happy*, *neutral*, and *sad* – in the improvised sessions and the entire dataset. For the first situation, 2280 utterances – with 289 *angry*, 284 *happy*, 1099 *neutral*, and 608 *sad* – are involved. For the latter one, a total of 10,039 utterances with 11 emotion classes are considered. The entire dataset is severely imbalanced in that the smallest class has two samples, while the largest class has around 2000 samples.

By using datasets of different sizes, we can investigate the capability of GAN-based augmentation in two scenarios: (1) the number of samples is significantly less than the feature dimensions and (2) the number of samples is slightly larger than the feature dimensions.

B. Emotion Features

OpenSmile [44] was used to extract emotion features specified in Interspeech 2011 Speaker State Challenge [45], which gives a 4368-dimensional feature vector for each utterance. This feature set is an extension of the Interspeech 2009 Emotion Challenge [46] and Interspeech 2010 Paralinguistic Challenge [47]. The former one mainly focuses on addressing the short-time emotional states and the latter one deals with speaker traits such as age and gender. The 2011 Challenge considers the short-time states and long-time traits. As a result, the features can represent emotions well and the number of features is suitable for studying the data sparsity problem.

The feature set in Interspeech 2011 Challenge comprises low-level descriptors, such as root-mean-square (RMS) frame energies, mel-frequency cepstrum coefficients (MFCCs), zero-crossing rates, voice probabilities, fundamental frequencies, and so on [48]. To extract these features, 25-ms frames with 10-ms frame shift were extracted from the waveforms. The low-level descriptors were then processed by statistical functionals – such as maximum, minimum, range, standard deviation, kurtosis, the slope of contour, etc. – to extract high-level descriptors of the speech signals. By applying these statistical functionals to the low-level descriptors, frame-level features can be *summarized* and converted to utterance-level suprasegmental features. More details can be found in the website of openSMILE.[1] For both datasets, we removed the features with zero variances and normalized the remaining features independently by z-norm.

C. Evaluations

Because we focused on speaker-independent emotion recognition, we applied leave-one-speaker-out cross-validation (LOSO-CV) to evaluate the performance of the emotion classifiers. There are ten speakers in each dataset. For each fold in EmoDB, we used the utterances of nine speakers for training and the utterances of the remaining speaker for testing. Because IEMOCAP consists of five sessions, each with a male and a female speaker, we extended the leave-one-speaker-out cross-validation to leave-one-session-out cross-validation. Specifically, for each fold in the LOSO-CV, we used four sessions for training and the remaining one for testing, which is equivalent to using utterances from eight speakers for training and utterances from the remaining two speakers for testing. Thus, we performed 10-fold cross-validation on EmoDB and 5-fold cross-validation on IEMOCAP. The LOSO-CV can ensure that no testing data were involved in either data augmentation or training of emotion classifiers. For performance comparison, we used both weighted accuracy (WA) and unweighted average recall (UAR), defined as follows:

$$\text{WA} = \frac{\sum_{k=1}^{K} \text{true-positives}_k}{\sum_{k=1}^{K} \text{total-positives}_k} \tag{19}$$

$$\text{UAR} = \frac{1}{K} \sum_{k=1}^{K} \frac{\text{true-positives}_k}{\text{total-positives}_k} \tag{20}$$

[1]https://www.audeering.com/opensmile/

where true-positives_k is the number of correctly classified samples for emotion category k, and total-positives_k is the actual number of samples with this emotion. K is the number of emotion categories.

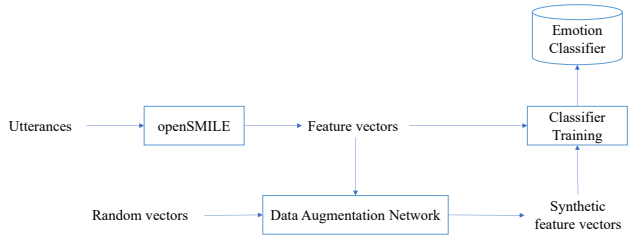


Fig. 2. Pipeline for the data augmentation and classifier training.

D. Experimental Setup

Fig. 2 shows the pipeline for the whole system, and the ADAN and WADAN were trained according to the algorithm shown in Algorithm 1 in the appendix.

We set the dimension of the random vectors and latent vectors to 100, i.e., $M = \dim(\mathbf{h}) = \dim(\hat{\mathbf{h}}) = 100$. The parameter α in Eq. 14 was set to 1 when the training data are from EmoDB or the improvised sessions in IEMOCAP. To deal with the class imbalance problem, different class weights were assigned to the classification loss (Eq. 11). If the largest class has n_m samples, and class k has n_k samples, then the weight assigned to this class will be $\frac{n_m}{n_k}$. The parameter α in Eq. 14 and Eq. 18 was set to 0.1 when the entire set of IEMOCAP was used in the cross-validation. Based on experience, the values of coefficients λ and p were set to 10 and 5, respectively. When the network converged, the generator was connected to the decoder to generate synthetic samples. We presented the one-hot emotion labels and Gaussian random vectors $\mathbf{z}$ to the generator. The synthetic latent vectors output from the generator were then passed to the decoder (the dashed arrow in Fig. 1) to produce augmented data in the original space. In our experiments, we created ten augmented sets, each of which has the same size and label distribution as the original set. The synthetic data was then augmented to the initial training set to train the emotion classifiers.

The components in the ADAN are fully-connected neural networks with two hidden layers. The number of hidden neurons is 800 for the encoder and the decoder, while it is 100 for the remaining parts. The size of hidden layers is chosen according to the number of input neurons and the number of output neurons such that no overfitting occurred. Support vector machines (SVMs) and simple deep neural networks (DNNs) were trained for emotion classification. ReLU was applied to every layer except for the last layer of the DNNs and the ADAN. Linear kernels were used in the SVM classifiers. The Xavier algorithm [49] was used to initialize the weights of the DNNs, and the Adam optimizer [50] with a learning rate of 0.0001 was used to train them. The SVMs were implemented using the *scikit-learn* package while the DNNs were implemented by using Tensorflow.

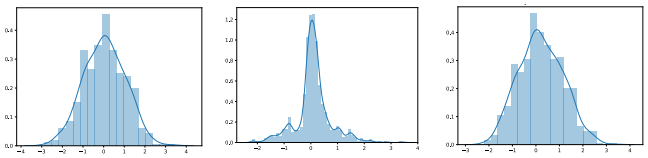
V. Results

A. Methods Comparison

Fig. 3 shows the loss curves of different models. If training the GAN based on the original objective (Eq. 1), the gradient vanishing problem would occur, as Fig. 3(a) shows. In order to provide more gradients to the generator, GANs are usually trained based on Eq. 2 and Eq. 3. However, it is still difficult to train a GAN with this improved objective. The increase in the generator loss (Eq. 3) in Fig. 3(b) suggests that the GAN fails to produce synthetic vectors that deceive the discriminator. The steady decrease in the discriminator loss (Eq. 2) in Fig. 3(b) suggests that the discriminator is strong enough to detect the synthetic samples produced by the generator. All of these evidences suggest that the GAN fails to converge. Several remedies – including choosing a smaller learning rate for the discriminator and increasing the number of iterations for training the generator – have been tried. But, all of them failed to make the GAN converge. The availability of class labels in cGAN (Fig. 3(c)) improves the situation slightly as the generator loss tends to flatten out. Nevertheless, convergence is still unstable. The generator loss of cGAN fluctuates widely because it is unable to achieve a real balance between the generator and the discriminator. It seems that the discriminator has been optimized and converged too early while the generator is still struggling for optimization. This reflects the difficulty in training a standard GAN in which the learning between the generator and the discriminator needs to be carefully balanced. Since the generated samples are very different from the real samples, especially when high dimensional synthetic data points are produced from a low dimensional random distribution, the discriminator can distinguish them easily.

Following [23], [38], the input to the AAE was obtained by sampling a 2-dimensional Gaussian mixture model in which each Gaussian component represents one emotion class. The corresponding loss curves are shown in Fig. 3(d). Different from the GAN and cGAN, the increase in the discriminator loss of AAE, ADAN, and WADAN suggests that the synthetic samples cause adversity in the discriminator. The loss curves of ADAN and WADAN in Fig. 3(e) and Fig. 3(f) also suggest that gradient vanishing did not occur even though the discriminator loss dropped to a small value.

The t-SNE [51] tool was used to project the data onto a 2-dimension embedded space for visualization, as shown in Fig. 4. We plotted the same number of real and synthetic samples in the graphs. In the figure, emotional states are represented by different colors, with darker and lighter colors corresponding to synthetic and genuine vectors, respectively. Except for the standard GAN, all networks can generate reasonable synthetic samples. Because the random samples input to the generator of GAN and cGAN were sampled from a Gaussian distribution, it seems to be difficult for the generator to learn the multi-class data distribution without label information. The figure also reveals that the cGAN can capture the entire dataset's distribution, but it is unable to match the distribution of individual classes. It is evident that the generated data from the AAE follow the predefined mixture of Gaussian distributions rather than the actual data distribution. For the proposed model (ADAN), not only are clusters formed in the latent space but also the synthetic samples follow the real data distribution. The t-SNE plots for WADAN are similar to those for ADAN, so we present one of them due to the page limit.



(a) Original data (b) AAE (c) ADAN

Fig. 5. Histograms of a randomly selected vector component in (a) original data, (b) AAE-generated data, and (c) ADAN-generated data.

We further investigated the properties of ADAN and AAE by analyzing their feature distributions. Fig. 5 shows the histogram of a randomly selected feature. The bin width was selected according to the Freedman-Diaconis rule. It is obvious that the distribution of the selected feature produced by ADAN is closer to the real data distribution. In addition, we measured the distances between the feature distributions of real data and those of synthetic data using the maximum mean discrepancy [52]:

$$\mathrm{MMD} = \frac{1}{N^2}\sum_{n_1=1}^{N}\sum_{n_1'=1}^{N} k\left(\mathbf{x}_{n_1}^r, \mathbf{x}_{n_1'}^r\right) + \frac{1}{N^2}\sum_{n_2=1}^{N}\sum_{n_2'=1}^{N} k\left(\mathbf{x}_{n_2}^f, \mathbf{x}_{n_2'}^f\right) - \frac{2}{N^2}\sum_{n_1=1}^{N}\sum_{n_2=1}^{N} k\left(\mathbf{x}_{n_1}^r, \mathbf{x}_{n_2}^f\right),$$

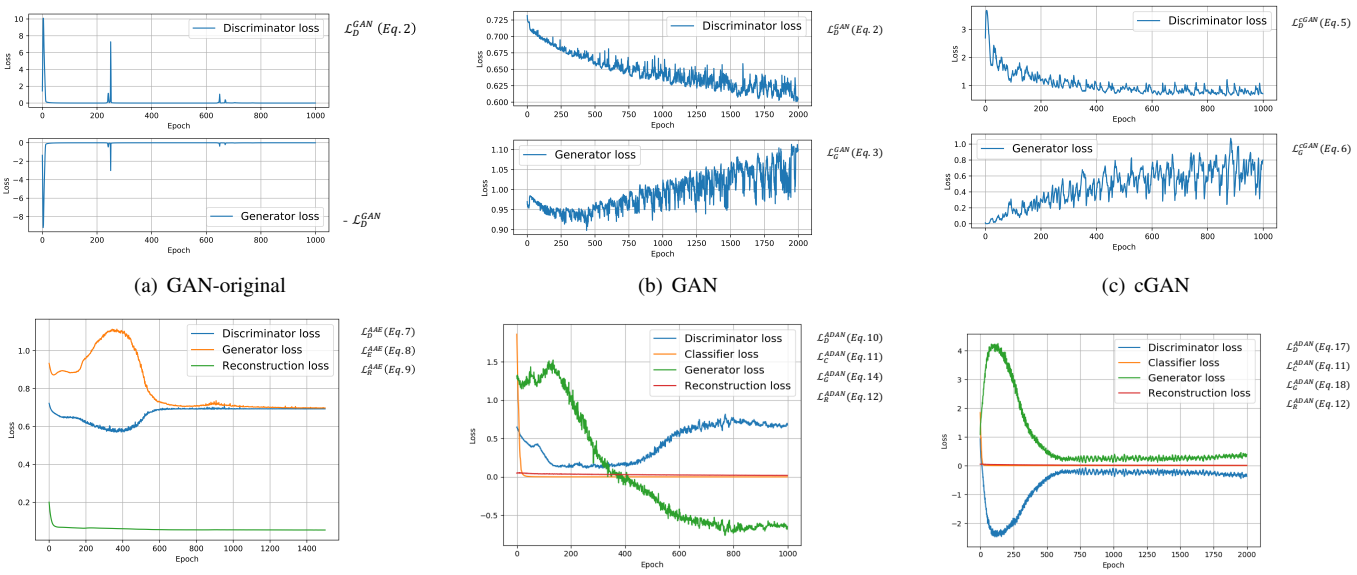
where $\mathbf{x}^r$'s are real samples and $\mathbf{x}^f$'s are synthetic samples. $k(\mathbf{x}^r, \mathbf{x}^f)$ represents the kernel function. The mixture of the radial basis function (RBF) kernels was applied, i.e., $k(\mathbf{x}, \mathbf{y}) = \sum_{q=1}^{K} \exp\left(-\frac{1}{2\sigma_q^2}\|\mathbf{x}-\mathbf{y}\|^2\right)$. The parameters σ_q's were set to $0.1, 1.0, 5.0, 10.0$, and 100, respectively.

Using the above settings, the average MMDs for AAE, ADAN, and WADAN are 0.0244, 0.0057, and 0.0054, respectively. Smaller MMD means smaller differences, which indicates that the synthetic samples generated by ADANs or WADANs are closer to the actual data distribution.

B. Performance on EMODB

As EmoDB is a small dataset, it is expected that it can benefit a lot from data augmentation. The original and the augmented data were used to train DNN classifiers and SVM classifiers. The DNNs are fully connected neural networks with two hidden layers, each with 100 neurons. L2 regularization was applied to train the DNNs.

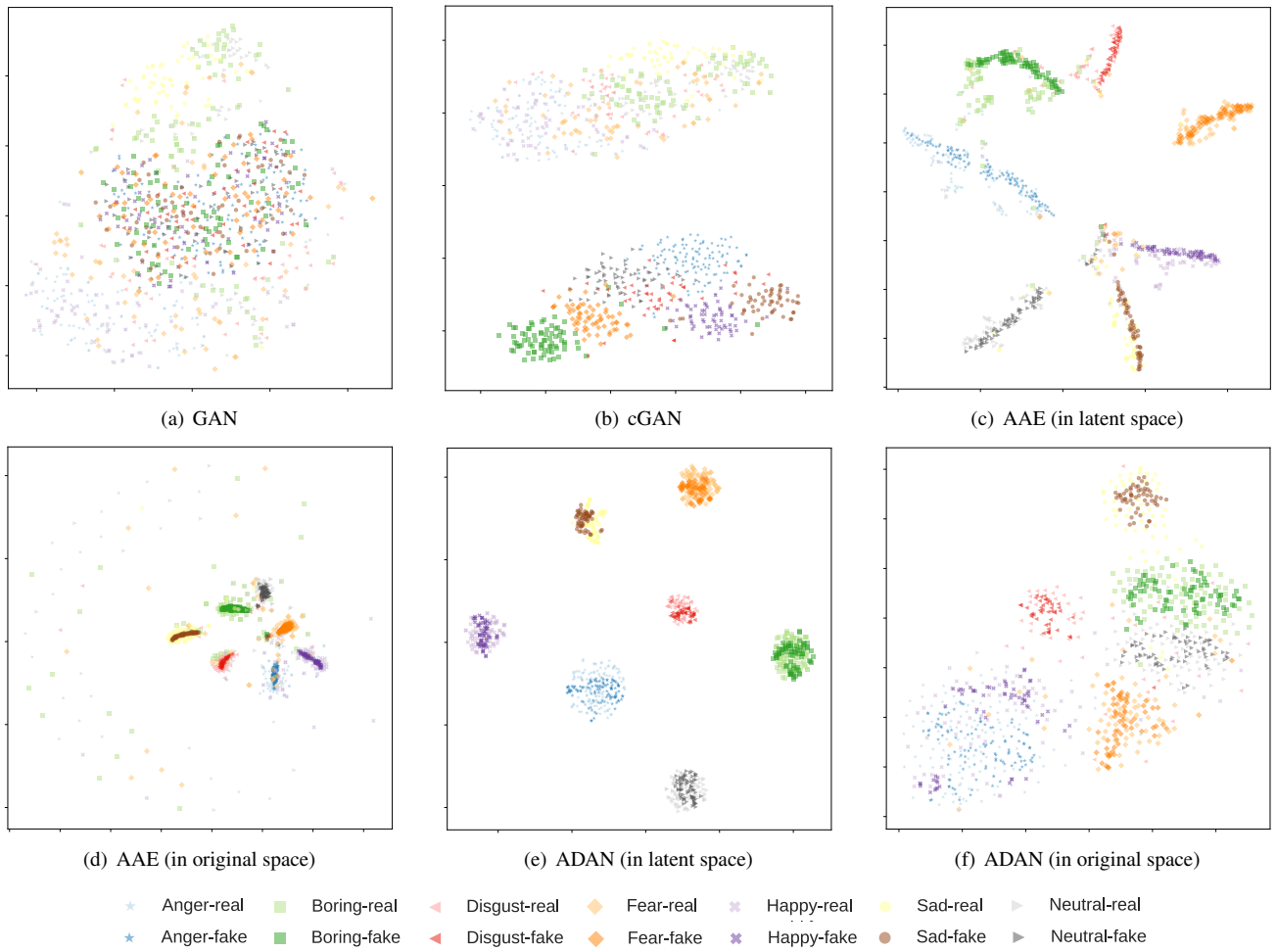
To analyze the effect of data augmentation, we increased the number of synthetic samples gradually. The synthetic samples were shuffled and randomly selected with the same category ratio as that of the original data. Because the performance of the classifiers could be affected by the selected samples, we repeated the experiments 3 times for each trial of random selection and took the average performance as the final accuracy. Fig. 6 shows the unweighted average recall of SVM and DNNs on EmoDB.



(a) GAN-original (b) GAN (c) cGAN

(d) AAE (e) ADAN (f) WADAN

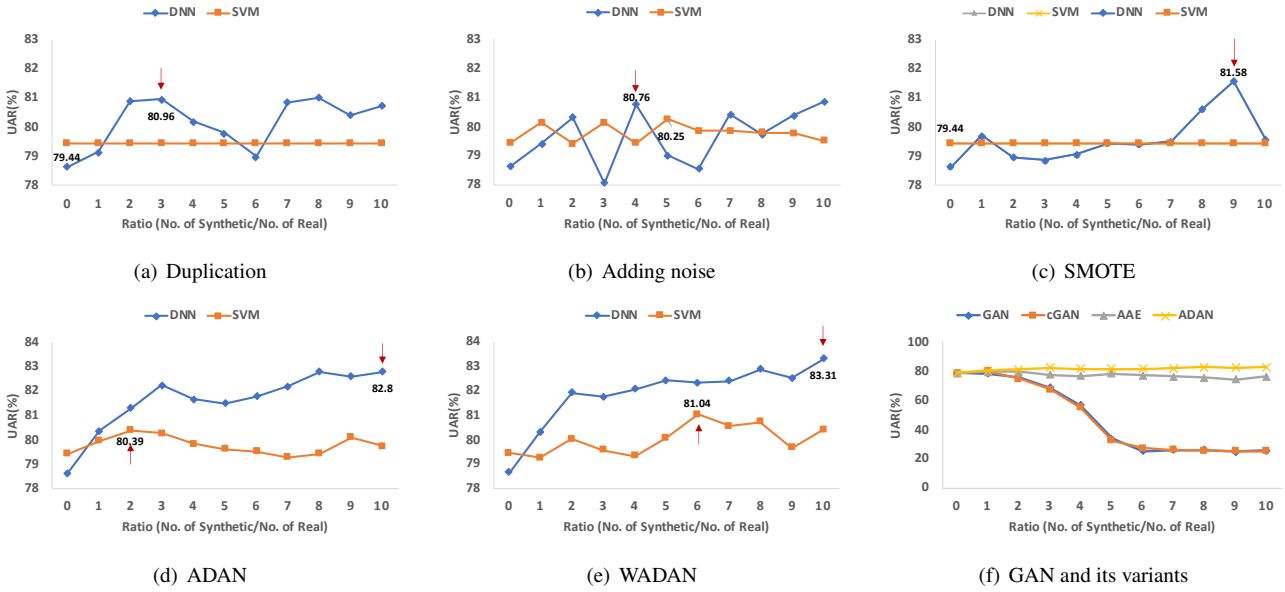
Fig. 3. Losses during the course of training of (a) original GAN corresponding to Eq. 1; (b) GAN corresponding to Eq. 2 and Eq. 3; (c) cGAN corresponding to Eq. 5 and Eq. 6; (d) AAE corresponding to Eq. 7 and Eq. 8; (e) ADAN corresponding to Eq. 10, Eq. 11, Eq. 12, and Eq. 14; and (f) WADAN corresponding to Eq. 17, Eq. 11, Eq. 12, and Eq. 18. All graphs are based on EmoDB.



(a) GAN (b) cGAN (c) AAE (in latent space)

(d) AAE (in original space) (e) ADAN (in latent space) (f) ADAN (in original space)

Fig. 4. T-SNE plots of (a) data in the original feature space after GAN-based augmentation, (b) data in the original feature space after cGAN-based augmentation, (c) real data in the latent space based on AAE, (d) data in the original feature space after AAE-based augmentation, (e) data in the latent space after ADAN-based augmentation, and (f) data in the original feature space after ADAN-based augmentation.



(a) Duplication (b) Adding noise (c) SMOTE

(d) ADAN (e) WADAN (f) GAN and its variants

Fig. 6. Unweighted average recall (UAR) achieved by different data augmentation techniques when the amount of augmented data on EmoDB is progressively increased. (a) copying real observations, (b) randomly adding Gaussian noise to the features, (c) SMOTE [53], (d) ADAN, (e) WADAN, and (f) GAN and its variants.

To highlight the effectiveness of ADAN, we compared its performance against some common data augmentation techniques, such as duplicating the samples, randomly adding noise to the feature values, and the synthetic minority over-sampling technique (SMOTE) [53]. Because the feature dimension is much larger than the number of training samples, non-linear classifiers are prone to over-fitting. Therefore, linear SVMs were used for classification.

Better DNNs can be trained, as shown in Fig. 6. Duplicating samples helps the DNN to learn the data distribution better, but more repeated samples would not lead to better performance. Augmenting data by adding noise to the features vectors can be considered as transforming the original data. A better DNN classifier can be trained but its performance depends on the amount of noise, resulting in performance fluctuation. The performance of SMOTE is relatively stable. SMOTE is designed for increasing the number of minority-class samples. It is not very effective for generating samples for all classes.

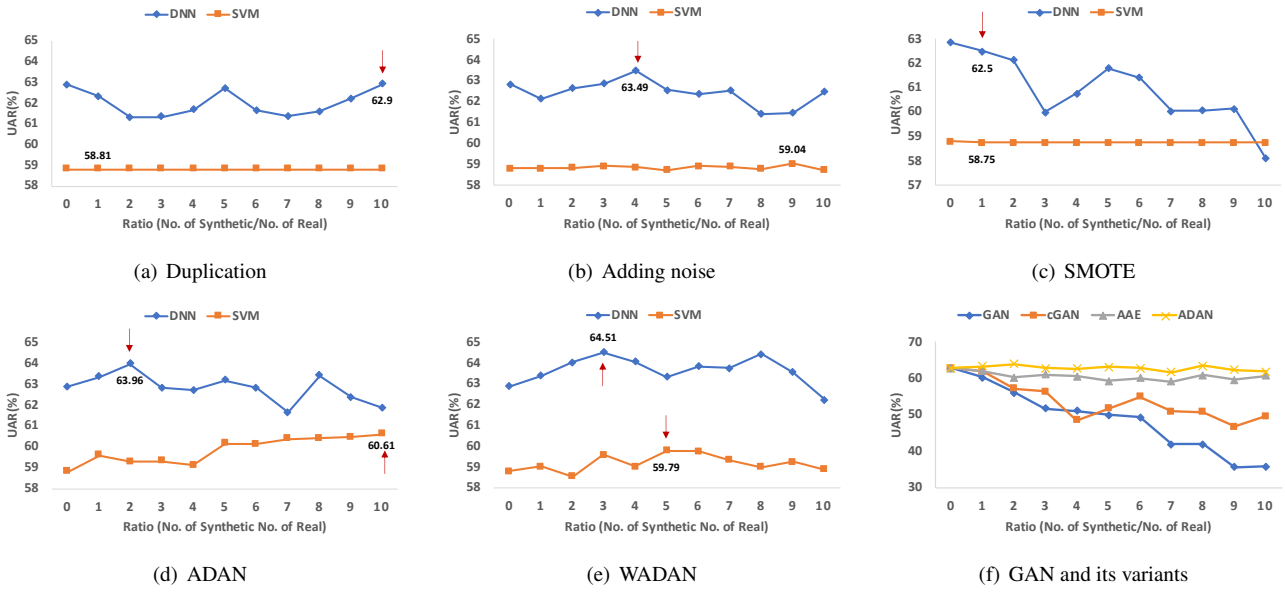
For the ADAN-based approach, with more synthetic data adding to the training set, a better emotion recognizer can be trained. It is also interesting to see that the ADAN-based method can help to improve the performance of SVMs as well. This suggests that the synthetic data from ADAN can help the SVM to find better separation hyperplanes. The performance can be improved further with the WADAN-based data augmentation approach, as Fig. 6(e) shows. The UAR increases gradually with more synthetic samples augmented to the training set. In the 10-fold cross-validation, one of them can always reach an extraordinary result of 100% UAR when the number of augmented sets is greater than 5. These results suggest that the ADAN and WADAN can generate novel and meaningful emotion vectors, which can help to improve the performance of emotion recognition.

We also trained the DNNs using augmented data from GAN, cGAN and AAE, and compared their performance as shown in Fig. 6(f). The performance is almost the same when training the DNNs with a small number of generated samples. However, when more augmented data generated from GAN or cGAN were used for training, the classifiers tended to make decisions by random guessing. For the AAE, no obvious performance gain was observed when the amount of augmentation increased. The mode collapse problem seems to occur such that the generated data fail to capture the whole picture of the actual data distribution, which can also be observed in Fig. 4(d).

TABLE I
COMPARISON OF DIFFERENT METHODS IN TERMS OF HIGHEST WEIGHTED ACCURACY (WA) AND UNWEIGHTED AVERAGE RECALL (UAR) ON THE EMODB DATASET.

Methods	**WA** (%)	**UAR**
Baseline		
SVM with IS11_Speaker_State (Mak [54])	80.56	-
DNN with IS11_Speaker_State (Mak [54])	80.19	-
Copying observations (DNN)	82.06	80.96
Adding noise to features (DNN)	82.06	80.85
SMOTE (DNN)	82.43	81.58
GAN (DNN)	80.37	78.60
cGAN (DNN)	81.50	79.93
AAE (DNN)	81.12	79.73
Related state-of-the-art approaches		
GMM/SVM (Luengo *et al.* [8])	78.30	-
2-D ACRNN (Chen *et al.* [55])	-	79.38
3-D ACRNN (Chen *et al.* [55])	-	82.82
Proposed		
ADAN + SVM	$81.50_{\pm 0.05}$	$80.39_{\pm 0.09}$
ADAN + DNN	$83.55_{\pm 0.19}$	$82.80_{\pm 0.03}$
WADAN + SVM	$81.87_{\pm 0.26}$	$81.04_{\pm 0.23}$
WADAN + DNN	$\mathbf{84.49}_{\pm 0.19}$	$\mathbf{83.31}_{\pm 0.20}$

Table I shows the highest weighted accuracy (WA) and unweighted average recall (UAR) obtained by different approaches. They were obtained by using different amounts of augmented data, which correspond to the peaks in Fig. 6 (as indicated by the red arrows). For example, the result of



(a) Duplication (b) Adding noise (c) SMOTE

(d) ADAN (e) WADAN (f) GAN and its variants

Fig. 7. Unweighted average recall (UAR) achieved by different data augmentation techniques when the amount of augmented data on IEMOCAP is progressively increased. (a) copying real observations, (b) randomly adding Gaussian noise to the features, (c) SMOTE [53], (d) ADAN, (e) WADAN, and (f) GAN and its variants.

ADAN+DNN was obtained by augmenting the training data with ten augmented sets, while that of SMOTE was obtained by augmenting the original data with nine augmented sets. For GAN, cGAN, and AAE, the original data were augmented with one augmented set because performance drops when the number of augmented sets increases. Table I shows that after ADAN-based data augmentation, we can train a better DNN to recognize emotions. The performance can be further improved when training the classifiers with the augmented samples generated from the WADAN, which is much better than the result obtained from Luengo *et al.* [8] that trained SVM classifiers using hand-crafted features. The UAR is even higher than that of Chen *et al.* [55], which uses a 3-D convolutional recurrent neural networks to generate discriminative features.

Bitouk *et al.* [56] and Vlasenko *et al.* [57] have found that feature selection can greatly improve the performance of emotion recognition. However, because the number of training samples used in these studies is different from ours, their results have not been added to Table I. Nevertheless, our work demonstrates that data augmentation is a prospective solution to the data sparsity problem, especially when training DNN classifiers. The promising results obtained from our proposed networks suggest that our proposed ADAN and WADAN can generate real-like samples, which can help to train better classifiers for speech emotion recognition.

C. Performance on IEMOCAP (improvised only)

Because the emotions in EmoDB have relatively clear boundaries, it may be easy to synthesize the emotion data. To obtain more convincing results, we conducted experiments on the IEMOCAP dataset in which different emotions do not have clear boundaries. Thus, to mimic its data distribution is more challenging.

We applied the same procedure as described in Section V-C to the IEMOCAP dataset. We used the original and augmented data to train DNNs with four hidden layers, each with 512 nodes. The same set of data was also used for training linear SVM classifiers. Due to the imbalanced training data, we assigned different class weights to the classification loss. Fig. 7 shows the performance of the DNNs and the SVM classifiers when the amount of augmented data is progressively increased. The results show that the traditional data augmentation techniques could not improve the performance much when more weights assigned to the losses of the smaller classes. In most instances, the performance is degraded when adding more data generated from the traditional methods, especially the SMOTE-based data augmentation. The ADAN-based approach performs better than the traditional methods, while WADAN-based data augmentation performs the best among all methods. This suggests that the WADAN-based method can generate novel data that helps the classifier to better recognize the emotions.

For the SVMs, increasing the amount of augmented data generated from the traditional approaches could not help the classifiers to better recognize the emotions. Duplicating training samples will not affect the decision boundary of linear SVMs, which explains why the accuracy does not change with respect to the amount of augmented data in Fig. 7(a). Having more noisy samples, as depicted in Fig. 7(b), could hurt performance. SMOTE uses interpolation to create novel samples for the minority class. But when it is used for creating samples for all classes, it could not produce effective samples for training the classifiers.

The comparison between GAN-based, cGAN-based, AAE-based, and ADAN-based data augmentation is shown in Fig. 7(f). Different from EmoDB, for IEMOCAP, the cGAN-based approach outperforms the GAN-based approach more significantly. A possible reason is that IEMOCAP has more data

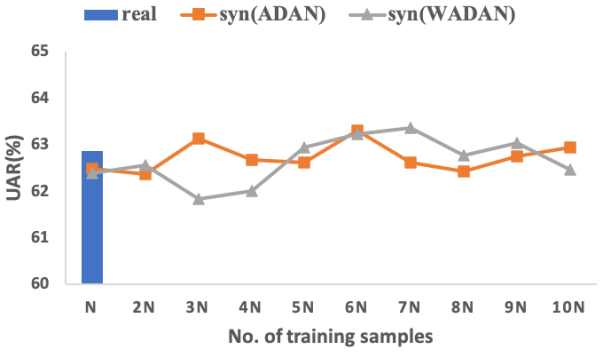


Fig. 8. Unweighted average recall (UAR) achieved by training the DNN classifiers with synthetic samples only.

and fewer categories than EmoDB, which would help to train better data augmentation networks. However, the performance of ADAN is stable for both datasets, which suggests that ADAN is capable of handling datasets of different scales.

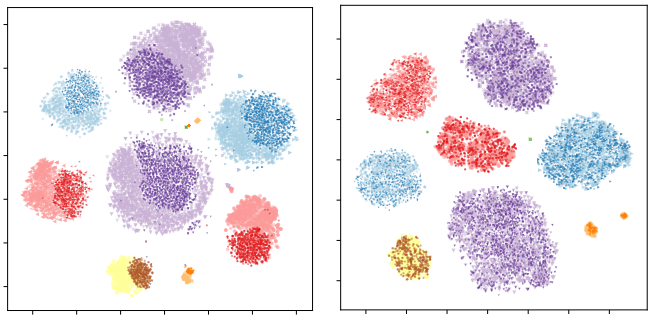
To evaluate the quality of the generated samples, we also used synthetic samples only to train the DNN classifier and evaluated it with the real test data. As shown in Fig. 8, the DNN classifier trained using only synthetic samples as input is competitive to a DNN trained using the real samples. When the number of training samples increases, the performance is also better. This suggests that the generated samples contain emotion information useful for emotion classification.

Table II compares the best performance of different systems. For the traditional approaches and our proposed methods, the results correspond to the peaks in Fig. 7 (as indicated by the red arrows). For the variants of GAN-based approaches, the results were obtained after adding one augmented set to the original data. There are many other approaches that achieve outstanding performance, but we only selected those using the same training data as ours. Variable-Length DNN and 3-D ACRNN are end-to-end systems that use spectrograms as input and produce emotion labels as output. The results show that ADAN-based methods outperform the baseline methods but could not beat the end-to-end methods in [21] and [55].

End-to-end systems can make use of discriminative training to extract emotion features from raw data. These features are more discriminative than the hand-crafted features extracted by OpenSMILE. Therefore, classifiers based on OpenSMILE features can hardly compete with these end-to-end systems. One may argue that it is also difficult to train an end-to-end system with such a small dataset. However, it has been shown that many techniques can reduce the number of parameters in CNN or RNN, which facilitates the training of end-to-end systems. In [19], the author proposed a novel pooling method that can downsample the feature maps to avoid over-parametrization. The author also argued that weights sharing can help to reduce the number of parameters in CNN. Nevertheless, our proposed WADAN-based system is not only superior to other traditional data augmentation technique, but it can also compete with the end-to-end system. Although we have only used OpenSMILE features to demonstrate the capability of ADANs, the method is general enough for creating augmented data to further improve the performance of the end-to-end systems.

TABLE II
COMPARISON OF DIFFERENT METHODS IN TERMS OF WEIGHTED ACCURACY (WA) AND UNWEIGHTED AVERAGE RECALL (UAR) ON THE IEMOCAP DATASET.

Methods	**WA** (%)	**UAR**
Baseline		
SVM with IS11_Speaker_State	63.20	58.81
DNN with IS11_Speaker_State	64.47	62.86
Copying observations (DNN)	65.79	62.90
Adding noise to features (DNN)	64.43	63.49
SMOTE (DNN)	65.83	62.50
GAN (DNN)	55.09	60.23
cGAN (DNN)	55.79	61.78
AAE (DNN)	63.86	62.01
Related state-of-the-art approaches		
Variable-Length DNN (Ma *et al.* [21])	**71.45**	64.22
2-D ACRNN (Chen *et al.* [55])	-	62.40
3-D ACRNN (Chen *et al.* [55])	-	**64.74**
Proposed		
ADAN + SVM	$64.78_{\pm 0.07}$	$60.07_{\pm 0.03}$
ADAN + DNN	$63.55_{\pm 0.37}$	$63.96_{\pm 0.27}$
WADAN + SVM	$63.82_{\pm 0.03}$	$59.79_{\pm 0.02}$
WADAN + DNN	$66.92_{\pm 0.20}$	$64.51_{\pm 0.15}$



(a) ADAN (b) WADAN

Fig. 9. Latent representations obtained from (a) ADAN, and (b) WADAN. Each marker represents one class, and there are 11 classes in total. Lighter colors represent the real samples, while darker colors represent the synthetic samples.

D. Performance on IEMOCAP (Entire dataset)

It is challenging to mimic the distributions of a severe imbalanced dataset. As mentioned in Section IV, different class weights were assigned to the classification loss (Eq. 11) when training the ADAN or WADAN. This can ensure that each class can form a cluster, even if it only has one sample. Fig. 9 shows the latent representations extracted from the ADAN and WADAN, respectively. The synthetic samples generated from the WADAN cover the real ones completely, while the synthetic samples generated from the ADAN only cover a part of the real data. In Fig. 9(a), there are two classes in green, but their corresponding synthetic representations cover only one of them. This indicates that ADAN fails to learn the distributions of some classes. From the perspective of the t-SNE plots, WADAN performs better than ADAN.

Since some classes only have 2 or 3 samples, it is hard to recognize all the emotions using 5-fold cross-validation. Therefore, we only considered four emotions and added the corresponding augmented data generated from the network trained with the entire dataset. The generated data were augmented to

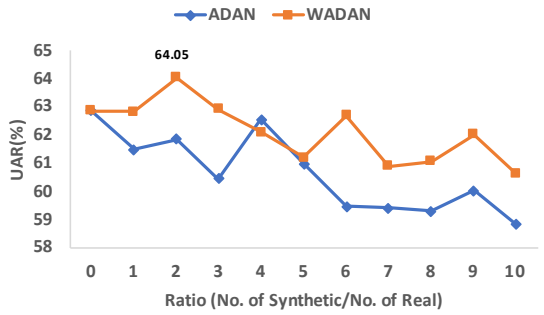


Fig. 10. Unweighted average recall (UAR) achieved by ADAN and WADAN when the amount of augmented data on the entire IEMOCAP is progressively increased.

the improvised set only. This means that our real data used to train the classifiers are the same as those in Section V-C, but the synthetic samples are generated differently. From the results shown in Fig. 10, it can be observed that the WADAN still performs well and its performance is much better than that of ADAN. This implies a better generalization capability of WADAN, even under extreme data-imbalance scenarios. However, the performance is worse than that in Fig. 7. A possible reason is that the networks were trained with the data including samples from the scripted sessions, which may affect the distributions of the generated samples.

E. Efficiency Analysis

The time complexity of the feedforward operation of a fully-connected (FC) network is $O(N_w)$, where N_w is the number of connection weights. This is because each neuron computes the linear weighted sum of its input. Our ADAN and WADAN fall into this category. The time complexity of training an FC network, however, is more complicated as it depends on a number of factors, including the kind of activation function, whether the network is over-specified (larger than needed), and whether regularization is applied [58], [59]. Instead of analyzing the theoretical time complexity, we estimated the actual computation time on a GTX1080Ti GPU by inputting a single sample to the networks. The average computation time is 0.02s and 0.07s for ADAN and WADAN, respectively. We also recorded the execution time of each training epoch with a batch size of 128 in ADAN and WADAN. The results are shown in Table III.

The number of epochs needed to achieve convergence depends on the number of training samples. Our observation is that to train an ADAN for EmoDB, IEMOCAP (improvised only) and IEMOCAP (entire set), at least 800, 200, and 100 epochs are respectively needed. For the WADAN, we found that the performance is better if the number of epochs is set to 1500. In summary, it is more efficient to use ADAN to generate synthetic samples, and our proposed networks may also be efficient for big data.

VI. Conclusions

Insufficient data could prevent deep learning models from reaching their full potential, which is a serious problem in DNN-based emotion recognition. Typically, the lack of training data will lead to overfitting in complex models. In this paper, we proposed a novel data augmentation network to produce synthetic samples that share the common latent representations with the original data. Instead of learning the emotion-aware vectors in the high-dimensional space, the proposed method can create an emotion-aware latent space and reconstruct samples in the original space. The results demonstrate that the proposed method can overcome the gradient vanishing problem in typical GANs, and produce emotion-rich augmented samples that are beneficial for training better emotion classifiers.

TABLE III
THE EXECUTION TIME OF EACH EPOCH WHEN TRAINING THE ADAN AND WADAN WITH DIFFERENT SIZES OF TRAINING DATA.

No. of Samples	Execution time (s)	
	ADAN	WADAN
$\sim$ 400 (EmoDB)	0.067	0.15
$\sim$ 2,000 (IEMOCAP improvised only)	0.3	0.67
$\sim$ 10,000 (IEMOCAP)	1.4	3.14

Other data augmentation techniques, such as duplicating the observations, transforming data, and SMOTE, were employed for comparison. The results reveal that the effects of different methods on nonlinear models are different. SMOTE can generate significant synthetic samples of the minority classes. Adding noise to the original samples can help when the original samples are reasonably distinguishable but will hurt performance when they are entangled with each other. The proposed ADAN and WADAN can generate valuable and novel samples that help to improve the recognition of emotions. Compared to other data augmentation techniques, our proposed method can achieve better performance. It can also make simple linear SVM classifiers trained with OpenSmile features on par with non-linear SVM classifiers trained with more advanced emotion features.

Our proposed model can overcome the difficulties in training standard GANs. The two dynamic inputs of the discriminator help to avoid gradient vanishing and training imbalance. The strategy of generating emotion-aware samples in the latent space followed by reconstructing samples in the original space facilitates the generator to confuse the discriminator. It also ensures that the generated samples can follow the actual data distribution. By replacing the cross-entropy adversarial loss by Wasserstein divergence, we have successfully made the proposed ADAN amendable to imbalanced datasets.

We have only considered the simple case in which the inputs to the data augmentation network are OpenSmile emotion vectors. Nevertheless, the augmentation network is general enough for other scenarios in which data augmentation is beneficial.

The proposed model still has limitations in that the synthetic samples follow closely the training data distribution. As a result, the augmented data would not be helpful if the distribution of test data is largely different from the training data distribution. In future research, we may consider generalizing the autoencoder and incorporating transfer learning into our proposed data augmentation model to make it more robust across different acoustic environments.

APPENDIX

Algorithm 1 Algorithm of ADAN and WADAN

Require: Batch size m, encoder E, decoder R, classifier C, generator G, discriminator D, coefficient α, coefficient λ, power p, training iterations n, and other hyperparameters

1: **for** $i \leftarrow 0$ to n **do**

2: Sample real data $\mathbf{x}_1, \ldots, \mathbf{x}_m$ from $\mathbb{P}_r$

3: Sample Gaussian noise $\mathbf{z}_1, \ldots, \mathbf{z}_m$ from $\mathcal{N}(\mathbf{0}, \mathbf{1})$

4: Sample emotion labels $\mathbf{y}_1, \ldots, \mathbf{y}_m$ from $\mathbb{P}_r$

5: (WADAN) Sample vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_m)$ from uniform distribution $U[0, 1]$

6: (WADAN) $\breve{\mathbf{x}}_j = (1 - \mu_j)\, \mathbf{x}_j + \mu_j G\left(\mathbf{z}_j, \mathbf{y}_j\right)$

7: Update the weights $\mathbf{w}_C$ of C by descending Eq. 11:

$$\mathbf{w}_C \leftarrow \text{Adam}\left(\nabla_{\mathbf{w}_C}\left(\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_C^j\right), \mathbf{w}_C\right)$$

8: Update the weights $\mathbf{w}_D$ of D by descending Eq. 10 for ADAN or Eq. 17 for WADAN:

$$\mathbf{w}_D \leftarrow \text{Adam}\left(\nabla_{\mathbf{w}_D}\left(\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_D^j\right), \mathbf{w}_D\right)$$

9: Freezing the weights of D.

10: Update the weights $\mathbf{w}_R$ of R by descending Eq. 12:

$$\mathbf{w}_R \leftarrow \text{Adam}\left(\nabla_{\mathbf{w}_R}\left(\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_R^j\right), \mathbf{w}_R\right)$$

11: Update the weights $\mathbf{w}_E$ of E by descending Eq. 13:

$$\mathbf{w}_E \leftarrow \text{Adam}\left(\nabla_{\mathbf{w}_E}\left(\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_E^j\right), \mathbf{w}_E\right)$$

12: Update the weights $\mathbf{w}_G$ of G by descending Eq. 14 for ADAN or Eq. 18 for WADAN:

$$\mathbf{w}_G \leftarrow \text{Adam}\left(\nabla_{\mathbf{w}_G}\left(\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_G^j\right), \mathbf{w}_G\right)$$

13: **end for**

REFERENCES

[1] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[3] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning," *Nature Biotechnology*, vol. 33, no. 8, pp. 831–838, 2015.

[4] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, "IEMOCAP: Interactive emotional dyadic motion capture database," *Language Resources and Evaluation*, vol. 42, no. 4, pp. 335–359, 2008.

[5] E. Cambria, "Affective computing and sentiment analysis," *IEEE Intelligent Systems*, vol. 31, no. 2, pp. 102–107, 2016.

[6] I. Chaturvedi, R. Satapathy, S. Cavallari, and E. Cambria, "Fuzzy commonsense reasoning for multimodal sentiment analysis," *Pattern Recognition Letters*, vol. 125, no. 264–270, 2019.

[7] B. Schuller, G. Rigoll, and M. Lang, "Hidden Markov model-based speech emotion recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003, pp. II–1.

[8] I. Luengo, E. Navas, and I. Hernáez, "Feature analysis and evaluation for automatic emotion identification in speech," *IEEE Transactions on Multimedia*, vol. 12, no. 6, pp. 490–501, 2010.

[9] K. Han, D. Yu, and I. Tashev, "Speech emotion recognition using deep neural network and extreme learning machine," in *Proc. Interspeech*, 2014, pp. 223–227.

[10] S. Ghosh, E. Laksana, L. P. Morency, and S. Scherer, "Representation learning for speech emotion recognition," in *Proc. Interspeech*, 2016, pp. 3603–3607.

[11] J. Lee and I. Tashev, "High-level feature representation using recurrent neural network for speech emotion recognition," in *Proc. Interspeech*, 2015, pp. 1537–1540.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[13] Z. Huang, M. Dong, Q. Mao, and Y. Zhan, "Speech emotion recognition using CNN," in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 801–804.

[14] N. Cummins, S. Amiriparian, G. Hagerer, A. Batliner, S. Steidl, and B. W. Schuller, "An image-based deep spectrum feature representation for the recognition of emotional speech," in *Proceedings of the 25th ACM International Conference on Multimedia*, 2017, pp. 478–484.

[15] Z. Zhao, Y. Zhao, Z. Bao, H. Wang, Z. Zhang, and C. Li, "Deep spectrum feature representations for speech emotion recognition," in *Proceedings of the Joint Workshop of the 4th Workshop on Affective Social Multimedia Computing and first Multi-Modal Affective Computing of Large-Scale Multimedia Data*, 2018, pp. 27–33.

[16] D. Luo, Y. Zou, and D. Huang, "Investigation on joint representation learning for robust feature extraction in speech emotion recognition," in *Proc. Interspeech*, 2018, pp. 152–156.

[17] L. Guo, L. Wang, J. Dang, L. Zhang, and H. Guan, "A feature fusion method based on extreme learning machine for speech emotion recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 2666–2670.

[18] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, "Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5200–5204.

[19] P. Li, Y. Song, I. McLoughlin, W. Guo, and L. Dai, "An attention pooling based representation learning method for speech emotion recognition," in *Proc. Interspeech*, 2018, pp. 3087–3091.

[20] H. M. Fayek, M. Lech, and L. Cavedon, "Evaluating deep learning architectures for speech emotion recognition," *Neural Networks*, vol. 92, pp. 60–68, 2017.

[21] X. Ma, Z. Wu, J. Jia, M. Xu, H. Meng, and L. Cai, "Emotion recognition from variable-length speech segments using deep learning on spectrograms," in *Proc. Interspeech*, 2018, pp. 3683–3687.

[22] J. Deng, Z. Zhang, E. Marchi, and B. Schuller, "Sparse autoencoder-based feature transfer learning for speech emotion recognition," in *Proceedings of the Humaine Association Conference on Affective Computing and Intelligent Interaction*, 2013, pp. 511–516.

[23] S. Sahu, R. Gupta, G. Sivaraman, W. AbdAlmageed, and C. Espy-Wilson, "Adversarial auto-encoders for speech based emotion recognition," in *Proc. Interspeech*, 2017, pp. 1243–1247.

[24] H. Hu, T. Tan, and Y. Qian, "Generative adversarial networks based data augmentation for noise robust speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5044–5048.

[25] S. Sahu, R. Gupta, and C. Espy-Wilson, "On enhancing speech emotion recognition using generative adversarial networks," *arXiv preprint arXiv:1806.06626*, 2018.

[26] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[27] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018.

[28] J. Gideon, S. Khorram, Z. Aldeneh, D. Dimitriadis, and E. M. Provost, "Progressive neural networks for transfer learning in emotion recognition," in *Proc. Interspeech*, 2017, pp. 1098–1102.

[29] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[30] A. Antoniou, A. Storkey, and H. Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[31] Z. Zhang, J. Han, K. Qian, C. Janott, Y. Guo, and B. Schuller, "Snore-GANs: Improving automatic snore sound classification with synthesized data," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 1, pp. 300–310, 2020.

[32] L. Yi and M. W. Mak, "Adversarial data augmentation network for speech emotion recognition," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2019.

[33] F. Burkhardt, A. Paeschke, M. Rolfes, W. F. Sendlmeier, and B. Weiss, "A database of German emotional speech," in *Proceedings of the 9th European Conference on Speech Communication and Technology*, 2005.

[34] S. N. Negahban, P. Ravikumar, M. J. Wainwright, B. Yu *et al.*, "A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers," *Statistical Science*, vol. 27, no. 4, pp. 538–557, 2012.

[35] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Policy*, pp. 1–30, 2004.

[36] T. Devries and G. W. Taylor, "Dataset augmentation in feature space," in *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, 2017.

[37] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *CoRR*, vol. abs/1411.1784, 2014.

[38] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2016.

[39] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the International Conference on Machine Learning*, 2017, pp. 214–223.

[40] J. Wu, Z. Huang, J. Thoma, D. Acharya, and L. Van Gool, "Wasserstein divergence for gans," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 653–668.

[41] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein GANs," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5767–5777.

[42] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *Stat*, vol. 1050, 2017.

[43] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation." in *AAAI Conference on Artificial Intelligence*, 2018, pp. 4058–4065.

[44] F. Eyben, F. Weninger, F. Gross, and B. Schuller, "Recent developments in OpenSMILE, the Munich open-source multimedia feature extractor," in *Proceedings of the 21st ACM International Conference on Multimedia*, 2013, pp. 835–838.

[45] B. Schuller, S. Steidl, A. Batliner, F. Schiel, and J. Krajewski, "The INTERSPEECH 2011 speaker state challenge," in *Proc. Interspeech*, 2011, pp. 3201–3204.

[46] B. Schuller, S. Steidl, and A. Batliner, "The INTERSPEECH 2009 emotion challenge," in *Proc. Interspeech*, 2009, pp. 312–315.

[47] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan, "The INTERSPEECH 2010 paralinguistic challenge," in *Proc. Interspeech*, 2010, pp. 2794–2797.

[48] J. R. Deller Jr, J. G. Proakis, and J. H. L. Hansen, *Discrete-time Processing of Speech Signals*. Macmillan Pub. Company, 1993.

[49] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[50] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[51] L. V. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[52] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A kernel method for the two-sample-problem." in *Advances in Neural Information Processing Systems*, 2006, pp. 513–520.

[53] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[54] M. W. Mak, "Feature selection and nuisance attribute projection for speech emotion recognition," Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Tech. Rep., Dec 2016.

[55] M. Chen, X. He, J. Yang, and H. Zhang, "3-D convolutional recurrent neural networks with attention model for speech emotion recognition," *IEEE Signal Processing Letters*, vol. 25, no. 10, pp. 1440–1444, 2018.

[56] D. Bitouk, R. Verma, and A. Nenkova, "Class-level spectral features for emotion recognition," *Speech communication*, vol. 52, no. 7-8, pp. 613–625, 2010.

[57] B. Vlasenko, B. Schuller, A. Wendemuth, and G. Rigoll, "Combining frame and turn-level information for robust recognition of emotions within speech," in *Proc. Interspeech 2007, Antwerp, Belgium*, 2007.

[58] M. Hosseini, M. Horton, H. Paneliya, U. Kallakuri, H. Homayoun, and T. Mohsenin, "On the complexity reduction of dense layers from $O(N^2)$ to $O(N \log N)$ with cyclic sparsely connected layers," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2019, pp. 1–6.

[59] R. Livni, S. Shalev-Shwartz, and O. Shamir, "On the computational efficiency of training neural networks," in *Advances in neural information processing systems*, 2014, pp. 855–863.