INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

GAN-based Data Generation for Speech Emotion Recognition

Sefik Emre Eskimez, Dimitrios Dimitriadis, Robert Gmyr, Kenichi Kumanati

Microsoft, One Microsoft Way, Redmond, WA, USA
{seeskime, didimit, rgmyr, kekumata}@microsoft.com

Abstract

In this work, we propose a GAN-based method to generate synthetic data for speech emotion recognition. Specifically, we investigate the usage of GANs for capturing the data manifold when the data is *eyes-off*, i.e., where we can train networks using the data but cannot copy it from the clients. We propose a CNN-based GAN with spectral normalization on both the generator and discriminator, both of which are pre-trained on large unlabeled speech corpora. We show that our method provides better speech emotion recognition performance than a strong baseline. Furthermore, we show that even after the data on the client is lost, our model can generate similar data that can be used for model bootstrapping in the future. Although we evaluated our method for speech emotion recognition, it can be applied to other tasks.

Index Terms: speech emotion recognition, generative adversarial networks, data augmentation

1. Introduction

Data augmentation is an essential component of low-resource tasks where collecting data is costly. These tasks usually rely on a small set of training data that the neural networks are prone to over-fit. For such low-resource tasks, data augmentation strategies are useful for improving the generalization capabilities of neural networks. One task that fits this scenario is speech emotion recognition (SER) due to difficulty in annotating data.

In recent years, most of the industry has shifted its data collection methods to privacy-focused settings. Federated learning (FL) [1] allows the distributed training of neural networks on *eyes-off* data that otherwise would be inaccessible due to privacy concerns. One shortcoming of traditional FL is that some client data is only available for a short period of time. Then, all this data is lost and cannot be used anymore for future training cycles.

Neural network-based data augmentation methods try to capture the data manifold, resulting in the generation of data statistically similar to the source data [2, 3, 4, 5, 6, 7]. As such, the augmentation networks (or generative networks) can be repurposed and used to model the *eyes-off* data. These generative networks, collected from the clients, can then synthesize data that can be used for training even after the original data is removed. Generative adversarial networks (GANs) [8] can directly model the data manifold and have shown impressive results for different data modalities. It is also shown that the GANs do not merely copy the data and retrieve it during generation; they instead generate new samples that are similar to the original samples.

In this work, we propose a GAN-based generative network for the SER task. We analyze our proposed network in two different training scenarios: 1) all data on a local machine (centralized) 2) each client contains a portion of the data (federated). Then we train SER networks solely on the synthetic data and compare the results. We also compare our network with another GAN-based data augmentation network [7]. The results suggest that our network provides better synthetic data than the baseline, and these generative networks can be used for data modeling for *eyes-off* data to bootstrap future models.

2. Related Work

Traditional systems utilize hand-crafted features usually computed in a sliding-window fashion and aggregated with functionals [9, 10, 6, 5]. There are fixed sets of hand-crafted features that are widely used in the research community, such as OpenSmile feature sets [11]. Some of these features include the mel-frequency cepstral coefficients (MFCCs), fundamental frequency (F_0), spectral features such as energy, frequency, and bandwidth of the formants. The functionals are used to obtain a feature representation for the whole utterance and usually include the mean, std, min, max, and range. Sahu et al. [6] used GANs to synthesize hand-crafted feature vectors for SER data augmentation. They experimented with vanilla GAN and conditional GAN [12]. Bao et al. [5] proposed to leverage the unlabeled speech datasets for data augmentation by using CycleGAN [13] to transfer emotion style of the feature vectors. Both approaches showed promising results.

The hand-crafted features are more suitable for modeling due to their lower dimensionality compared to spectrograms or raw waveforms. However, generating examples of a hand-crafted feature set restricts future use to compatible recognition models. Generating spectrograms or raw waveforms provides much more flexibility in future research by allowing us to train models directly on the raw waveforms [14], multi-channel audio [15] or on extracted features [16].

Chatziagapi et al. [7] proposed generating log-mel spectrograms with GANs to address data imbalance by augmenting the minority class samples. First, they trained an autoencoder using reconstruction loss; then, they initialized their generator and discriminator with the pre-trained decoder and encoder, respectively. For conditioning, they estimated the mean and covariance of the latent code for each class and used these statistics to generate the multivariate noise to represent classes during the GAN training. This method also showed promising results addressing the data imbalance.

In this work, we propose a similar system to [7] for generating spectrograms. Our method includes improvements such as large-scale pre-training, spectral normalization, different scale learning rate, gradient penalty, and nearest-neighbor interpolation. These improvements lead to high generation quality. We compare our method with the baseline method using SER classification performance as a metric.

3. Method

In this section, we provide more detail on the neural network architecture, class-conditioning, pre-training, and fine-tuning.

http://dx.doi.org/10.21437/Interspeech.2020-2898

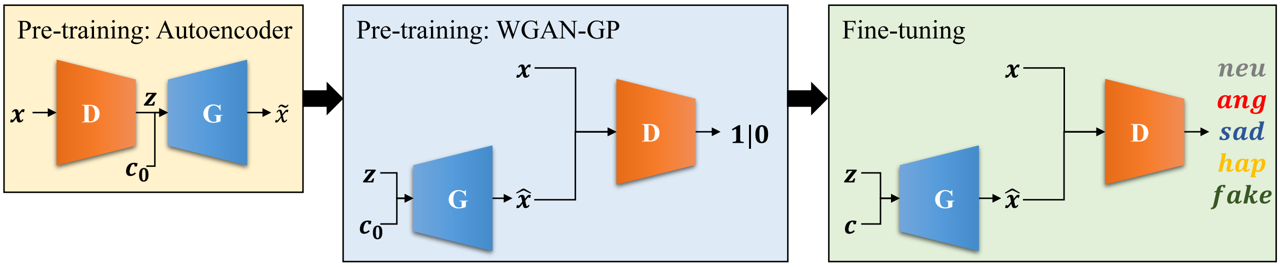

Figure 1: *Training stages of the proposed method are shown. x is the real spectrograms, $\tilde{x}$ is the reconstructed spectrograms, $\hat{x}$ is the generated spectrograms, z is the noise input, c_0 is a N_c-dimension vector of zeros as a placeholder for condition, and c is the one-hot condition vector.*

3.1. Model Architecture

We propose a deep convolutional GAN (DCGAN)-style [17] architecture similar to the baseline model [7]. The baseline work used transposed convolutional layers that cause the output to have checkerboard artifacts, making it easily distinguishable from real samples. To alleviate this effect, we instead use nearest-neighbor interpolation for upsampling. Another major difference is that we do not use any batch-normalization or dropout layers in either the generator or discriminator networks.

Furthermore, for more stable GAN training, we perform spectral normalization for all layers in the generator and discriminator [18, 19], and use a gradient-penalty when training the discriminator [20]. Besides, we employ different learning rate scales for the discriminator and the generator [21, 19], which allows for more efficient GAN training for regularized discriminators. In the original WGAN-GP work [20], the discriminator takes five steps for each generator step; instead, we use a higher learning rate for the discriminator and let the generator and discriminator step at the same frequency.

For class-conditioning, we concatenate the one-hot emotion labels to the noise input, as in [12]. This way, the generator has explicit emotion information during generation. We set the noise dimension to N_z, when we add the emotion conditions, the input of the generator becomes a $N_z + N_c$-dimension vector, where N_c is the number of classes. The generator takes this input and feeds it to two linear layers that are followed by a leaky ReLU activation. The resulting intermediate representation is reshaped into eight by eight images. These images are fed into nine convolutional layers, the first eight of which are followed by leaky ReLU activations. The last layer uses a hyperbolic tangent activation to normalize the spectrogram output to a range of [-1, 1]. After every second convolutional layer, we use nearest-neighbor interpolation to upsample the images.

The discriminator is an eight-layer convolutional neural network with leaky ReLU activations that operates on raw spectrograms. Every second convolutional layer's stride is set to 2 for downsampling. After the convolutional layers, the intermediate representation is flattened and fed into two linear layers. A leaky ReLU activation follows the first linear layer; a softmax activation follows the last layer during fine-tuning, and there is no activation during pre-training. For fine-tuning, the discriminator outputs N_c+1 neurons: N_c for the emotion classes and the last for the fake class as in [2, 7]. This architecture allows the generator to capture the data distribution of the minority classes even when the dataset is imbalanced. We applied spectral normalization to each layer in both networks. The architectures for both networks are shown in Table 1.

Table 1: *The details of the network architectures are shown. Both networks include leaky ReLU activations after each layer except for the last. Spectral normalization is applied to all layers. The filter size for the convolutional layers is set to 3.*

Network	Layer	Input Shape	Stride	Output Shape
Generator	FC	(N_z+N_c,)	-	(4096,)
	FC	(4096,)	-	(32768,)
	Conv2D	(512, 8, 8)	1	(512, 8, 8)
	Conv2D	(512, 8, 8)	2	(512, 16, 16)
	Conv2D	(512, 16, 16)	1	(256, 16, 16)
	Conv2D	(256, 16, 16)	2	(256, 32, 32)
	Conv2D	(256, 32, 32)	1	(128, 32, 32)
	Conv2D	(128, 32, 32)	2	(128, 64, 64)
	Conv2D	(128, 64, 64)	1	(64, 64, 64)
	Conv2D	(64, 64, 64)	2	(64, 128, 128)
	Conv2D	(64, 128, 128)	1	(1, 128, 128)
Discriminator	Conv2D	(1, 128, 128)	1	(64, 128, 128)
	Conv2D	(64, 128, 128)	2	(64, 64, 64)
	Conv2D	(64, 64, 64)	1	(128, 64, 64)
	Conv2D	(128, 64, 64)	2	(128, 32, 32)
	Conv2D	(128, 32, 32)	1	(256, 32, 32)
	Conv2D	(256, 32, 32)	2	(256, 16, 16)
	Conv2D	(256, 16, 16)	1	(512, 16, 16)
	Conv2D	(512, 16, 16)	2	(512, 8, 8)
	FC	(32768,)	-	(4096,)
	FC	(4096,)	-	(N_c+1,)

3.2. Training Strategies

Training the neural networks on abundant unlabeled data before fine-tuning on the actual task (transfer learning) is shown to be highly beneficial for NLP, image, and speech domains [22, 23, 24, 25]. Since labeled speech emotion data is minimal, we employed pre-training to leverage the unlabeled data. Figure 1 shows the training overview of our system.

Pre-Training: Reconstruction Loss - First, we form an autoencoder using our discriminator as an encoder and generator as a decoder, similar to [7]. We replace the last layer of the encoder (discriminator) to output a N_z-dimension latent code. For the remaining N_c-dimension input, we provide a vector of zeros as a condition placeholder during the pre-training. Consequently, the corresponding parameters for the emotion conditions do not get activated during the pre-training. We solely use L_1 loss for autoencoder training. This initialization method leads to a stable solution and avoids the mode collapse issue usually encountered during GAN training. This step is distinctly different from the baseline model [7], where the authors pre-trained the network on the same emotion data, which is still prone to over-fitting. In contrast, we employ a large unlabeled speech corpus with different recording conditions and many speakers.

Pre-Training: WGAN-GP - After the autoencoder pre-

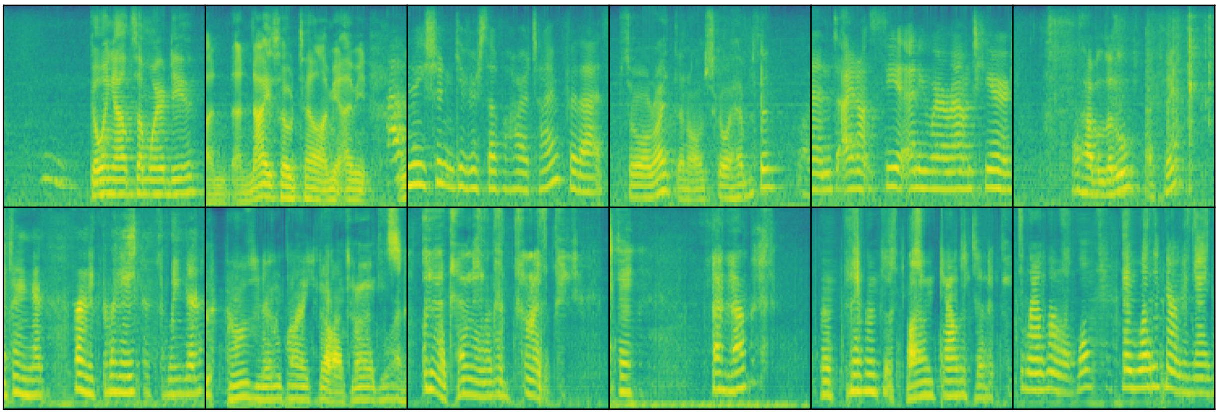

Figure 2: *Spectrogram examples are shown for real and fake samples. The first row shows the real spectrograms, and the second row shows the generated spectrograms. The generated and real samples are hard to distinguish visually.*

training, we initialize the weights of our discriminator and generator from the encoder and decoder respectively. In this case, the discriminator outputs a single neuron, and it is tasked with detecting if the sample is real or fake. Similar to autoencoder pre-training, we provide the generator with an N_z-dimension noise vector plus N_c-dimension zeros vector. Again, since the data is unlabeled, we keep the parameters corresponding to the condition inputs inactive. During WGAN-GP pre-training, we enable the gradient penalty and different learning rate scales. After the WGAN-GP pre-training, the resulting networks can generate highly realistic log-mel spectrograms, indistinguishable from the real data. An example set of spectrograms is shown in Figure 2. After these two steps of pre-training, the networks can generalize well and is ready for fine-tuning.

Fine-Tuning - We load the pre-trained weights from the previous step and replace the last layer of the discriminator to output N_c+1 neurons, as shown in Table 1. In this step, we provide the one-hot emotion labels to the generator. Since the generator can already generate high-quality spectrograms, now it focuses on capturing the underlaying emotion structure to fool the discriminator. For optimizing the discriminator, we use the sparse categorical cross-entropy loss.

Another advantage over the baseline method is that, using conditional GANs, we can create paired speech emotion data that might be useful for paired style transfer networks by using the same noise input and different emotion conditions. Figure 3 shows the samples generated from the same noise input and different emotion conditions.

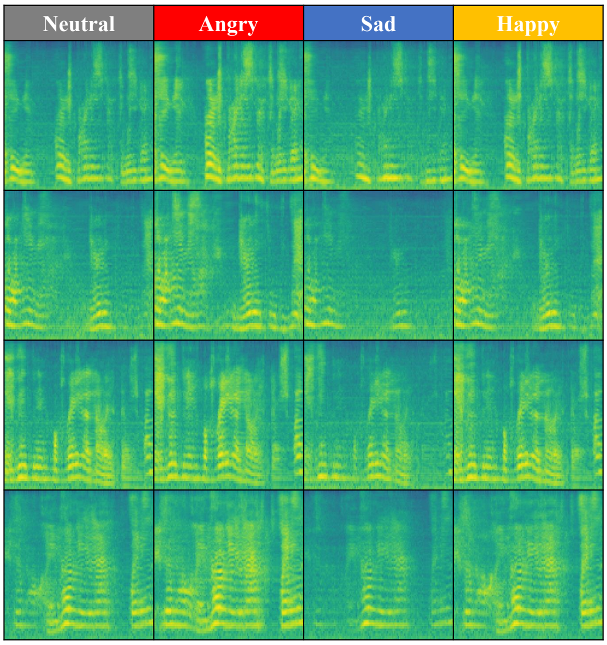

Figure 3: *Paired spectrogram examples are shown for the conditional generation. Each row shows the samples generated from the same noise vector, and each column shows a different emotion condition. From left to right, columns show the neutral, angry, sad, and happy emotions, respectively.*

4. Experiments

In this section, we provide the dataset information, implementation details, the experiment scenarios, and the qualitative evaluation results.

4.1. Datasets

In our experiments, we employed two datasets: interactive emotional dyadic motion capture database (IEMOCAP) [26] for labeled speech emotion data, and LibriSpeech [27] for unlabeled speech data.

IEMOCAP is a well-established multi-modal dataset in SER literature. It contains 12 hours of emotion data from 10 speakers (5 females and 5 males). The recordings are provided in 5 sessions; for each session, there are two speakers. The recordings are provided at a 16 kHz sampling rate. The dataset contains 12 emotions; however, we followed the other works in literature and select only four emotions: *neutral*, *angry*, *sad*, and *happy*. Note that we also merged the *excited* category with *happy* category, following previous works in the field. The total speech emotion data that we used for these four categories is around 7 hours.

For pre-training, we adopted 360 hours of speech data from the LibriSpeech Corpus [27], a widely used dataset for automatic speech recognition (ASR) research. LibriSpeech contains 2500 speakers in total, each of whom narrates book passages. The recordings are provided at a 16 kHz sampling rate. LibriSpeech allows our model to learn from a wide variety of speakers and speech styles, improving the generation quality and generalization capability of our networks beyond that which is possible using only the limited labeled speech emotion data.

4.2. Pre-processing

We extracted the spectrograms of the speech signals using a 32 ms window size, 16 ms hop size (50%), and 512 FFT bins. The number of mel coefficients is set to 128, and we created 128 by 128 spectrogram patches (around 2 seconds) for training. To stabilize the training, we normalized the spectrograms within the [-1, 1] value range by calculating the minimum and maximum values of the combined Librispeech and IEMOCAP datasets.

4.3. SER Network

For an objective evaluation, we employed a neural network for SER and compared its classification performance when trained on the real data with that when trained on synthetic data. The network architecture is identical to our discriminator/encoder architecture, except for the last layer. We replaced the last layer with a linear layer that outputs N_c outputs rather than N_c+1. For all experiments listed in this section, we trained this network from scratch using the specified data. Since our focus is on data generation rather than improving the classification performance, we selected this simple SER architecture for all our experiments. During testing, for each file, we employed a sliding window with a hop size of 32 frames. The emotion probabilities for each window were aggregated using the mean.

4.4. Implementation Details

All networks were implemented using the PyTorch framework [28]. We used the Adam optimizer for training all networks, setting β_1=0.5 and β_2=0.99. The learning rate during autoencoder pre-training was set to 5e-5. Throughout GAN training (both pre-training and fine-tuning), the learning rates of the generator and discriminator were set to 2e-6 and 2e-5, respectively. The learning rate for the SER network was set to 1e-5 and decayed by an order of 10 for every 10k iterations. The coefficient for the gradient penalty was set to 10, the mini-batch size was 32 samples, and the noise dimension, N_z, was 128. The number of emotions, N_c, was 4. We optimized the networks for 200K iterations during the autoencoder and GAN pre-training.

For all experiments listed in this section, we employed *leave-one-session-out cross-validation*, which is a common practice for small datasets such as IEMOCAP. We trained the data augmentation models using four sessions, generated the synthetic data, trained the SER network with the synthetic data and evaluated the performance on the remaining session. We repeated this for all sessions and reported the average results of five sessions. We used 5-fold cross-validation for the original IEMOCAP experiments as well.

4.5. Scenarios and Results

We consider two scenarios: 1) a server possessing all data (centralized), and 2) each client containing a subset of the data. The first scenario evaluates if the synthesized data can capture the data statistics and if it is useful for the classification task. It is equivalent to data augmentation experiments, so we can directly compare our proposed method with the baseline. The second scenario evaluates if we can model the data on different clients and pool the synthesized data of different clients in a way that benefits the classification task. In this way, the server will take the *eyes off* raw data stored in each client to ensure privacy.

Table 2: *The classification results for the original and synthetic data. We show the mean and std F1-scores of 5-fold cross-validation.*

Data	F1-Score Mean	F1-Score Std
IEMOCAP	57.27	0.99
Baseline [7]	43.05	2.97
Proposed - Centralized	48.54	3.89
Proposed - Federated	**50.39**	3.31

Centralized Training- In this scenario, the data from the four sessions were used to fine-tune the GAN. Then, we generated 40K samples for each emotion category, trained the SER network, and evaluated on the remaining session. We did not observe further improvements by adding more synthetic data samples. The results are shown in Table 2, denoted as *Proposed - Centralized*. Compared to the original IEMOCAP results, some performance loss is expected for the synthetic data, as shown in [5]. While training on the original data yields a *57.27%* mean F1-score, our synthetic data yields *48.54%*. Our proposed method provides improved classification performance compared to the baseline method.

Federated Training- In this scenario, each client contained the data from a single session. We sent a pre-trained copy of our proposed GAN to each client and fine-tuned them on each client's respective emotion data. Then we collected these models and generated 10K samples for each emotion from each client's model. Again, we employed 5-fold cross-validation: we pooled data from four sessions and trained the SER network and evaluated on the remaining session. Therefore, the total number of samples generated per emotion was equal to centralized training (40K). We repeated this for each session and averaged the results, which is shown in Table 2. The results suggest that modeling each session with a different GAN yields a 3.8% relative classification improvement compared to the centralized training.

Discussion- The results suggest that the *eyes-off* data can be modeled with some expected information loss and can be used in the future for bootstrapping new models. In the federated setting, each client modeled the emotion data of two speakers as opposed to modeling the emotion data of eight speakers. We argue that, due to having more parameters per speaker, the federated setting yielded more personalized spectrograms that capture the variations of emotions for those speakers. Therefore, it yielded slightly better results than the centralized training. Readers should note that although we call this a *federated setting*, it should not be confused with traditional federated learning since we do not aggregate model parameters from different clients.

5. Conclusions

In this work, we proposed a GAN that can generate speech emotion spectrograms, which can be used for training SER networks. We showed that our method provides better generation quality compared to the baseline method, and showed that the pure synthetic data could yield decent results for *eyes-off* data. We proposed to use the GANs for modeling imbalanced and highly skewed data among clients for future use, even after the original data is removed. Our future work includes modeling the raw waveforms for more flexibility in future data use.

6. Acknowledgements

The authors would like to thank William Hinthorn, Masaki Itagaki, Yashesh Gaur, Michael Zhang and Xuedong Huang for their project support and technical discussions.

7. References

[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[2] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi, "Bagan: Data augmentation with balancing gan," *arXiv preprint arXiv:1803.09655*, 2018.

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

[4] X. Zhu, Y. Liu, Z. Qin, and J. Li, "Data augmentation in emotion classification using generative adversarial networks," *arXiv preprint arXiv:1711.00648*, 2017.

[5] F. Bao, M. Neumann, and N. T. Vu, "Cyclegan-based emotion style transfer as data augmentation for speech emotion recognition," *Manuscript submitted for publication*, pp. 35–37, 2019.

[6] S. Sahu, R. Gupta, and C. Espy-Wilson, "On enhancing speech emotion recognition using generative adversarial networks," *Proc. Interspeech 2018*, pp. 3693–3697, 2018.

[7] A. Chatziagapi, G. Paraskevopoulos, D. Sgouropoulos, G. Pantazopoulos, M. Nikandrou, T. Giannakopoulos, A. Katsamanis, A. Potamianos, and S. Narayanan, "Data augmentation using gans for speech emotion recognition," *Proc. Interspeech 2019*, pp. 171–175, 2019.

[8] 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 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[9] M. El Ayadi, M. S. Kamel, and F. Karray, "Survey on speech emotion recognition: Features, classification schemes, and databases," *Pattern Recognition*, vol. 44, no. 3, pp. 572–587, 2011.

[10] K. Han, D. Yu, and I. Tashev, "Speech emotion recognition using deep neural network and extreme learning machine," in *Fifteenth annual conference of the international speech communication association*, 2014.

[11] F. Eyben, M. Wöllmer, and B. Schuller, "Opensmile: the munich versatile and fast open-source audio feature extractor," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 1459–1462.

[12] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[13] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[14] T. N. Sainath, R. J. Weiss, A. W. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *INTERSPEECH*, 2015.

[15] M. Wu, K. Kumatani, S. Sundaram, N. Strom, and B. Hoffmeister, "Frequency domain multi-channel acoustic modeling for distant speech recognition," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6640–6644, 2019.

[16] J. Guo, K. Kumatani, M. Sun, M. Wu, A. Raju, N. Ström, and A. Mandal, "Time-delayed bottleneck highway networks using a dft feature for keyword spotting," in *ICASSP*, 2018, pp. 5489–5493.

[17] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[18] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[19] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[20] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

[21] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in neural information processing systems*, 2017, pp. 6626–6637.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[23] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[24] A. Romero, C. Gatta, and G. Camps-Valls, "Unsupervised deep feature extraction for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1349–1362, 2015.

[25] S. E. Eskimez, Z. Duan, and W. Heinzelman, "Unsupervised learning approach to feature analysis for automatic speech emotion recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5099–5103.

[26] 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, p. 335, 2008.

[27] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[28] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.