IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 4, 2020, accepted November 14, 2020, date of publication November 26, 2020,
date of current version December 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3040797

High-Fidelity Audio Generation and Representation Learning With Guided Adversarial Autoencoder

KAZI NAZMUL HAQUE[1], RAJIB RANA[1], (Member, IEEE), AND BJÖRN W. SCHULLER, JR.[2,3], (Fellow, IEEE)

[1]Department of Computer Science, School of Science, University of Southern Queensland, Toowoomba, QLD 4301, Australia
[2]Group on Language, Audio and Music (GLAM), Imperial College London, London SW7 2AZ, U.K.
[3]Chair of Embedded Intelligence for Health Care and Wellbeing, University of Augsburg, 86159 Augsburg, Germany

Corresponding author: Kazi Nazmul Haque (shezan.huq@gmail.com)

ABSTRACT Generating high-fidelity conditional audio samples and learning representation from unlabelled audio data are two challenging problems in machine learning research. Recent advances in the Generative Adversarial Neural Networks (GAN) architectures show great promise in addressing these challenges. To learn powerful representation using GAN architecture, it requires superior sample generation quality, which requires an enormous amount of labelled data. In this paper, we address this issue by proposing Guided Adversarial Autoencoder (GAAE), which can generate superior conditional audio samples from unlabelled audio data using a small percentage of labelled data as guidance. Representation learned from unlabelled data without any supervision does not guarantee its' usability for any downstream task. On the other hand, during the representation learning, if the model is highly biased towards the downstream task, it losses its generalisation capability. This makes the learned representation hardly useful for any other tasks that are not related to that downstream task. The proposed GAAE model also address these issues. Using this superior conditional generation, GAAE can learn representation specific to the downstream task. Furthermore, GAAE learns another type of representation capturing the general attributes of the data, which is independent of the downstream task at hand. Experimental results involving the S09 and the NSynth dataset attest the superior performance of GAAE compared to the state-of-the-art alternatives.

INDEX TERMS Audio generation, representation learning, generative adversarial neural network, guided generative adversarial autoencoder.

I. INTRODUCTION

Representation learning aims to map higher-dimensional data into a lower-dimensional representation space where the variational factors of the data are disentangled. Learning a disentangled representation from an unlabelled dataset opens a window of opportunity for researchers to utilise the vastly available unlabelled data for any downstream tasks [1]. Such as, a representation learnt from freely available YouTube audios (movie, news etc.) can be used to improve a task such as emotion recognition from audio where a large labelled dataset is unavailable.

Generative Adversarial Neural Network (GAN) [2] has shown great promise for learning powerful representation. GAN is comprised of a Generator network and a Discriminator network, where these networks are trained to defeat each other based on a minimax game. During training, the Generator tries to fool the Discriminator by generating real-like samples from a random noise/latent distribution, and the Discriminator tries to defeat the Generator by differentiating the generated sample from the real samples [2]. During this game-play, the Generator disentangles the underlying attributes of the data in the given random latent distribution [3]. This helps in learning powerful representations [3]–[9] in a unsupervised manner. GAN based models pose great promise in audio research where limited or no labelled data is available.

The representation learning performance of the GANs usually improves along with its' sample generation quality. Intuitively, GAN models that can generate high-quality

The associate editor coordinating the review of this manuscript and approving it for publication was Ananya Sen Gupta.

samples, intrinsically learns powerful representation [6]. GAN-based models are successful at generating high-fidelity images, however, they fail to perform likewise for the complex audio waveform generation as it requires modelling higher-order temporal scales [10]. To successfully generate audio with GANs, many researchers have worked with the spectrogram of the audio which can be converted back to the audio with minimal loss [10]–[12]. Recently proposed high performing GAN architectures such as BigGAN [13] and StyleGAN [9] are not well explored in the audio field, leaving a room to explore the compatibility of these models for audio data.

A representation learnt with GANs in a completely unsupervised manner does not guarantee the usability of the learnt representation for any particular downstream task. This is because it can ignore the important characteristics of the data during the training which is important for succeeding in the downstream task [14]. So, some bias towards the downstream task is necessary during the unsupervised training to succeed in that downstream task [1].

GAN models perform better for conditional generation using labelled data. The labels add useful side information during the training, which helps the GAN models to decompose overall sample generation tasks into sub-tasks according to the conditioned labels. Though the conditional generation helps to improve performance significantly, it requires an enormous amount of labelled data [15], which is costly and/or error-prone. Using the GAN models to generate high-quality samples with a minimum amount of labelled data therefore remains a crucial challenge [14].

In our previous work, we propose a BigGAN based architecture called ''Guided Generative Adversarial Neural Network (GGAN),'' which can generate state-of-the-art (SOTA) conditional audio with fewer labelled data. This labelled data is used as a guidance to force GGAN to learn guided representation for any downstream task at hand. Note that, the learned representation for any particular downstream task makes it less useful for any other task that is unrelated to the downstream task [14]. In many cases, it is desirable to learn representation in a manner so that it can be used for any particular downstream task as well as can be used for any future tasks independent of the downstream task at hand [16]. It is a challenging problem to learn both generalised and guided representation at the same time with conditional GAN architectures. During the training of any conditional GAN, the latent noise/samples are independent of the given condition. So, GAN learns to map the general characteristics of the training data from the latent samples, which is independent of the condition. On the other hand, if the condition is imposed on the latent samples/noise like GGAN, the latent cannot learn general characteristics as it is biased towards the conditioned attributes. In this paper, we address this problem. Our contributions are as follows:

- We propose a novel autoencoder based GAN model GAAE, which can generate high-fidelity audio samples capturing the diverse modes of the training data distribution leveraging the guidance from a fewer labelled data samples from that dataset or a related dataset.
- We evaluate the conditional sample generation quality of the proposed model based on two audio datasets: the Speech Command dataset (S09) and the Musical Instrument Sound dataset (Nsyth). We demonstrate that the GAAE model performs significantly better than the SOTA models.
- We achieve generalised and guided representation in our GAAE model. Evaluation results on three different datasets: the Speech Command dataset (S09), the Audio Book Speech dataset (Librispeech), and the Musical Instrument Sound dataset (Nsyth) show that the proposed GAAE model performs better than SOTA models.

II. BACKGROUND AND RELATED WORK

A. AUDIO REPRESENTATION LEARNING

While there is a rich literature of supervised representation learning, due to our focus on unsupervised representation learning we will only discuss the related literature here. In the field of unsupervised representation learning, the self-supervised learning has become very popular recently due to its unprecedented success in the field of computer vision [17]–[23] and natural language processing [24]–[27]. Self-supervised learning uses information presents in the unlabelled data to create an alternative supervised signal to train the model for learning feature/representation. For an example, learning representation through predicting the rotation angel of images where rotation angel serves as supervised signal and this learned representation can be used to improve other related image classification tasks [28].

Likewise, in the audio field, researchers have achieved good performances using self-supervised representation learning. In their work, DeepMind [29] have proposed a model to learn a useful representation from unsupervised speech data through predicting a future observation in the latent space. In another work from Google [30], the representation is learnt by predicting the instantaneous frequency based on the magnitude of the Fourier transform. Furthermore, Arsha *et al.* (2020) [31] proposed a cross-modal self-supervised learning method to learn speech representation from the co-relationship between the face and the audio in the video. Other efforts have been made by researchers to learn a general representation by predicting the contextual frames of any particular audio frame like wav2vec [32], speech2vec [33], and audio word2vec [34]. Likewise, there are other successful implementations [35]–[38] of the self-supervised representation learning in the field of audio.

Though self-supervised learning is good for learning representations from unlabelled datasets, it requires manual endeavour to design the supervision signal [39]. To avoid this, researchers have focused on fully unsupervised representation learning mainly using autoencoders [40]–[42]. In [43], the authors learnt representations with an autoencoder

from a large unlabelled dataset, which improved the emotion recognition from speech audio. Similarly, in another work, the authors used a denoising autoencoder to improve affect recognition from speech data [44]. Several works [5], [45], [46] have utilised Variational Autoencoders (VAEs) [47] to learn an efficient speech representation from an unlabelled dataset. Recently, given the popularity of adversarial training, different works have been conducted by researchers to learn a robust representation with GANs [48], [49] and Adversarial Autoencoders [50], [51].

Though learning a representation from prodigiously available unlabelled datasets is very intriguing, the recent work from Google AI has proved that completely unsupervised representation learning is not possible without any form of supervision [1]. Also, representation learnt from an unsupervised method does not guarantee the usability of this learnt representation for any post use case scenario. Thus, as outlined, we proposed the Guided Generative Adversarial Neural Network (GGAN) [14], which can learn a powerful representation from an unlabelled audio dataset according to the supervision given from a fewer amount of labelled data. Therefore, in the learnt representation space, the GGAN disentangles attributes of the data according to the given categories from the labelled dataset, which benefits the related post-use case scenario.

B. AUDIO GENERATION

Most of the audios are periodic, and high-fidelity audio generation requires modelling a higher order magnitude of the temporal scales, which makes it a challenging problem [10]. Most of the research works related to audio generation are based on the audio synthesis viz; Aaron and *et al.* (2016) have proposed a powerful autoregressive model named "Wavenet," which works very well on text to speech (TTS) synthesis for both English and Mandarin. Later, the authors have improved this work by proposing "Parallel Wavenet," which is 20 times faster than the original Wavenet. Other researchers have utilised the seq2seq model for TTS such as Char2Wav [52] and TACOTRON [53]. However, these audio generation methods are conditioned on the text data and mainly focused on speech generation. Thus, these methods cannot be generalised to all other audio domains, even for speech data where transcripts are not available.

In the context of generating audio without any condition on the text data, the GANs are very promising due to their massive success in the field of computer vision [6], [9], [54]–[56]. However, porting these GAN architectures directly to the audio domain does not offer similar performance as the audio waveform is mostly more complex than an image [10], [11]. Therefore, researchers have focused on generating spectrogram (2D image-like representation of audio) rather than generating directly a waveform. Then, the generated spectrogram is converted back to audio. Chris *et al.* (2019) [11] have trained a GAN-based model to generate spectrograms and successfully converted them back to the audio domain with the Griffin-Lim algorithm [57]. In their TiFGAN paper [12], the authors have proposed a phase-gradient heap integration (PGHI) [58] algorithm for better reconstruction of the audio from the spectrogram with minimal loss. As the PGHI algorithm is good at reconstructing audio from the spectrogram, now the challenge is to generate a realistic spectrogram. As the spectrogram is—as outlined—an image-like representation of the audio, any GAN based framework from the image domain should be compatible. Hence, the BigGAN architecture [13] has shown promising performance at generating conditional high resolution/fidelity images, but it was not well explored for audio generation. In this paper we address this gap.

C. CLOSELY RELATED ARCHITECTURES

The proposed GAAE model is a semi-supervised model, as we leverage a small amount of labelled data during the training. In [59], the authors proposed a semi-supervised version of the InfoGAN model [4] to capture a specific representation and generation according to the supervision which comes from a small number of labelled data. But, the success of this model in terms of the complex data distribution is not evident. Other researchers have explored the scope of semi-supervision in GAN architectures [15], [60], [61] to improve the conditional generation, but most of these works are not explored in the audio domain which leaves a major gap for the researchers to address. The GAAE model is based on an Adversarial Autoencoder (AAE) [8], where we have extended the AAE model to learn both guided and generalise/style representation from an unlabelled dataset in a semi-supervised fashion. Furthermore, in the GAAE model, we have implemented a unique way to leverage the small amount of labelled data for conditional audio generation. Here, we have also proposed a way to utilise the generated conditional samples for improving the representation learning during the training. Moreover, the building block for our GAAE model is a BigGAN architecture; thus, we further contribute by exploring the use of a BigGAN in an autoencoder-based model for audio data.

D. AUDIO REPRESENTATION LEARNING

While there is a rich literature of supervised representation learning, due to our focus on unsupervised representation learning we will only discuss the related literature here. In the field of unsupervised representation learning, the self-supervised learning has become very popular recently due to its unprecedented success in the field of computer vision [17]–[23] and natural language processing [24]–[27]. Self-supervised learning uses information presents in the unlabelled data to create an alternative supervised signal to train the model for learning feature/representation. For an example, learning representation through predicting the rotation angel of images where rotation angel serves as supervised signal and this learned representation can be used to improve other related image classification tasks [28].

Likewise, in the audio field, researchers have achieved good performances using self-supervised representation

learning. In their work, DeepMind [29] have proposed a model to learn a useful representation from unsupervised speech data through predicting a future observation in the latent space. In another work from Google [30], the representation is learnt by predicting the instantaneous frequency based on the magnitude of the Fourier transform. Furthermore, Arsha *et al.* (2020) [31] proposed a cross-modal self-supervised learning method to learn speech representation from the co-relationship between the face and the audio in the video. Other efforts have been made by researchers to learn a general representation by predicting the contextual frames of any particular audio frame like wav2vec [32], speech2vec [33], and audio word2vec [34]. Likewise, there are other successful implementations [35]–[38] of the self-supervised representation learning in the field of audio.

Though self-supervised learning is good for learning representations from unlabelled datasets, it requires manual endeavour to design the supervision signal [39]. To avoid this, researchers have focused on fully unsupervised representation learning mainly using autoencoders [40]–[42]. In [43], the authors learnt representations with an autoencoder from a large unlabelled dataset, which improved the emotion recognition from speech audio. Similarly, in another work, the authors used a denoising autoencoder to improve affect recognition from speech data [44]. Several works [5], [45], [46] have utilised Variational Autoencoders (VAEs) [47] to learn an efficient speech representation from an unlabelled dataset. Recently, given the popularity of adversarial training, different works have been conducted by researchers to learn a robust representation with GANs [48], [49] and Adversarial Autoencoders [50], [51].

Though learning a representation from prodigiously available unlabelled datasets is very intriguing, the recent work from Google AI has proved that completely unsupervised representation learning is not possible without any form of supervision [1]. Also, representation learnt from an unsupervised method does not guarantee the usability of this learnt representation for any post use case scenario. Thus, as outlined, we proposed the Guided Generative Adversarial Neural Network (GGAN) [14], which can learn a powerful representation from an unlabelled audio dataset according to the supervision given from a fewer amount of labelled data. Therefore, in the learnt representation space, the GGAN disentangles attributes of the data according to the given categories from the labelled dataset, which benefits the related post-use case scenario.

E. AUDIO GENERATION

Most of the audios are periodic, and high-fidelity audio generation requires modelling a higher order magnitude of the temporal scales, which makes it a challenging problem [10]. Most of the research works related to audio generation are based on the audio synthesis viz; Aaron and *et al.* (2016) have proposed a powerful autoregressive model named "Wavenet," which works very well on text to speech (TTS) synthesis for both English and Mandarin. Later, the authors have improved this work by proposing "Parallel Wavenet," which is 20 times faster than the original Wavenet. Other researchers have utilised the seq2seq model for TTS such as Char2Wav [52] and TACOTRON [53]. However, these audio generation methods are conditioned on the text data and mainly focused on speech generation. Thus, these methods cannot be generalised to all other audio domains, even for speech data where transcripts are not available.

In the context of generating audio without any condition on the text data, the GANs are very promising due to their massive success in the field of computer vision [6], [9], [54]–[56]. However, porting these GAN architectures directly to the audio domain does not offer similar performance as the audio waveform is mostly more complex than an image [10], [11]. Therefore, researchers have focused on generating spectrogram (2D image-like representation of audio) rather than generating directly a waveform. Then, the generated spectrogram is converted back to audio. Chris *et al.* (2019) [11] have trained a GAN-based model to generate spectrograms and successfully converted them back to the audio domain with the Griffin-Lim algorithm [57]. In their TiFGAN paper [12], the authors have proposed a phase-gradient heap integration (PGHI) [58] algorithm for better reconstruction of the audio from the spectrogram with minimal loss. As the PGHI algorithm is good at reconstructing audio from the spectrogram, now the challenge is to generate a realistic spectrogram. As the spectrogram is—as outlined—an image-like representation of the audio, any GAN based framework from the image domain should be compatible. Hence, the BigGAN architecture [13] has shown promising performance at generating conditional high resolution/fidelity images, but it was not well explored for audio generation. In this paper we address this gap.

F. CLOSELY RELATED ARCHITECTURES

The proposed GAAE model is a semi-supervised model, as we leverage a small amount of labelled data during the training. In [59], the authors proposed a semi-supervised version of the InfoGAN model [4] to capture a specific representation and generation according to the supervision which comes from a small number of labelled data. But, the success of this model in terms of the complex data distribution is not evident. Other researchers have explored the scope of semi-supervision in GAN architectures [15], [60], [61] to improve the conditional generation, but most of these works are not explored in the audio domain which leaves a major gap for the researchers to address. The GAAE model is based on an Adversarial Autoencoder (AAE) [8], where we have extended the AAE model to learn both guided and generalise/style representation from an unlabelled dataset in a semi-supervised fashion. Furthermore, in the GAAE model, we have implemented a unique way to leverage the small amount of labelled data for conditional audio generation. Here, we have also proposed a way to utilise the generated conditional samples for improving

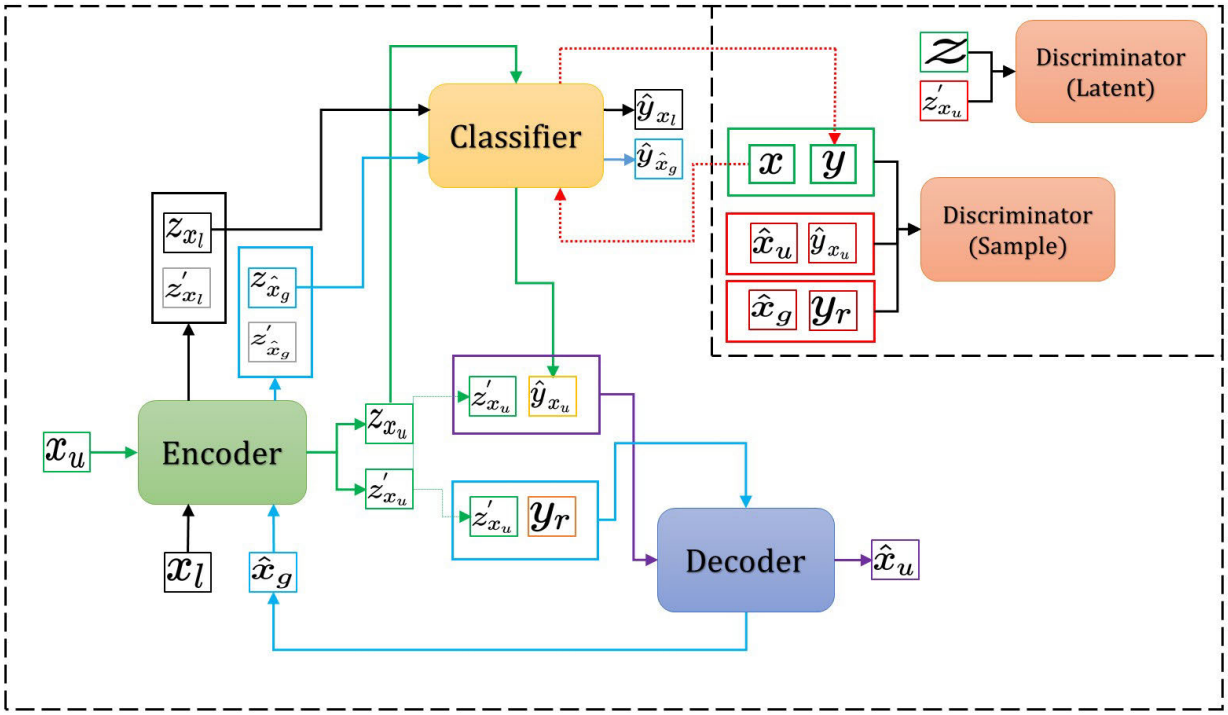


FIGURE 1. This figure illustrates the overall architecture of the GAAE model. Different networks of the GAAE model are shown along with the connections between them. In the figure, the arrows are coloured to highlight the flow of any input/output of the model. For the discriminator, the red boxes show the fake samples and the green boxes indicate the real samples. Here, x_u is the unlabelled data sample, x_l is the labelled data sample,$\hat{x}_u$ is the reconstructed data sample, y_r is the random conditions, and z is the known latent distribution.

the representation learning during the training. Moreover, the building block for our GAAE model is a BigGAN architecture; thus, we further contribute by exploring the use of a BigGAN in an autoencoder-based model for audio data.

III. PROPOSED RESEARCH METHOD

A. ARCHITECTURE OF THE GAAE

The GAAE consists of five neural networks: the Encoder E, the Decoder D, the Classifier C, the Latent Discriminator L and the Sample Discriminator S. Let the parameters for these networks be $\theta_e, \theta_d, \theta_c, \theta_L$, and θ_S respectively. Figure 1 shows the whole architecture of the model and the description is as follows.

1) ENCODER

The Encoder E takes any unlabelled data sample $x_u \sim p_{data}$ and outputs two latent samples $z_{x_u} \sim u_z$ and $z'_{x_u} \sim q_z$, where p_{data} is the true unlabelled data distribution, and u_z,q_z are two different continuous distributions learned by the E. We require the latent z_{x_u} to capture the post-task-specific attributes/characteristics of the data and the latent z'_{x_u} to capture the general/style attributes of the data.

2) CLASSIFIER

We have a classifier network C which is trained with limited labelled data $x_l \sim p_{ldata}$, where p_{ldata} is the labelled data distribution and not necessarily $p_{ldata} \subset p_{data}$. Here, with this p_{ldata}, the whole model gets guidance—thus, we call this data as "guidance data." Now, the C network takes any latent sample and predicts the category class for that latent sample. To train C, we pass x_l through the E network and get two latent vectors $\{z_{x_l}, z'_{x_l}\} = E(x_l; \theta_e)$. Then, we only forward z_{x_l} through C to get the predicted label $\hat{y}_{x_l} = C(z_{x_l}; \theta_c)$ and train C against the true label $y_l \sim Cat(y_l, k = n)$ of the sample x_l, where $Cat(y_l, k = n)$ is the categorical distribution with n numbers of categories/labels. These labels are used as one-hot vector. For now, lets consider that C can classify the label of any sample correctly.

3) DECODER

The Decoder D maps any latent and categorical class/label variable to the data sample. Now, to get the reconstructed sample of x_u, we pass the latent z'_{x_u} and the label of x_u through the D network. As x_u is an unlabelled data sample, we get the label $\hat{y}_{x_u} = C(z_{x_u}, \theta_c)$ through the network C and obtain the reconstructed sample $\hat{x}_u = D(z'_{x_u}, \hat{y}_{x_u}; \theta_d)$ from the D network. Here, we also want to use the D network for generating samples according to the given condition along with the reconstruction. Therefore, the same latent z'_{x_u} is used with a random categorical variable (one-hot vector) y_r, sampled from categorical distribution $Cat(y_r, K = n, p = \frac{1}{n})$, where n is the number of categories/labels, and the sampling probability for each category is $\frac{1}{n}$. Now, we obtain the generated sample $\hat{x}_g \sim p_{gdata}$, where p_{gdata} is the generated data

distribution by the D network, and it is trained to match p_{gdata} with the true data distribution p_{data}. Here, the size of n is the same as of the guided data, and we want the D network to generate data according to the categories from the guided data. Therefore, we ensure this with the Discriminator where the Discriminator receives the labels of the data from the network C. As we use a small number of labelled data, it is hard to train C due to the problem of overfitting. Hence, we use the generated sample $\hat{x}_g$ and train the C network considering y_r as the true label/category, where the predicted label is $\hat{y}_{\hat{x}_g} = C(E(\hat{x}_g, \theta_e), \theta_c)$.

Here, C depends on the correct conditional generation from D, and D depends on the classification from the network C. During the training, the C network starts to predict the category of some samples from the given labelled data correctly. Likewise, the Discriminator learns to identify the correct category for those samples and forces the D network to generate samples with the attributes related to these correctly classified samples. These generated samples bring more characteristics with them, which are not present in the given labelled data but belong to the data distribution. Now, as we feed these generated samples again to the C network with the associated conditional categories as correct labels, it learns to predict the correct category for more samples related to that generated samples. Then again, these new correctly classified samples improve the conditional generation of the D network. Hence, throughout the training, the C network and D network improve each other continuously. Meanwhile, during the training, the representation learning (latent generation) capability of the E network is also ameliorated via the process of reconstructing sample x_u, which also improves the performance of the C and D network eventually.

4) DISCRIMINATORS

The GAAE model has two discriminators: the Sample Discriminator S and the Latent Discriminator L. S makes sure that the generated sample $\hat{x_g}$ and the reconstructed sample $\hat{x_u}$ match the sample from the true data distribution p_{data}. We train S with the sample and its label. Now, for the samples $\hat{x_g}$ and $\hat{x_u}$, we have the labels y_r,$\hat{y}_{x_u}$ respectively. Hence, the pairs $(\hat{x_g}, y_r)$ and $(\hat{x_u}, \hat{y}_{x_u})$ are considered fake labels for the discriminator S. For the true data, both x_l and x_u are used together, where we get the label for the sample x_u from C, and, for the sample x_l we use the available true labels. Hence, in terms of distribution perspective, we obtain the data distribution p_{mdata}, mixing the distributions p_{ldata} and p_{data}. Accordingly, S is trained with the true sample data $x \sim p_{mdata}$ along with its associated label y if it exists, otherwise with the predicted label from C.

Here, the network E learns to map the general characteristics of the data onto the latent distribution q_z, excluding the categories from the guided data. Now, if we can draw the sample from the q_z distribution, then, by using the categorical distribution as condition, we can generate diverse data for different categories (categories from the guided data) from the Decoder D. We can only sample from q_z, if the distribution is known to us. Therefore, we use another Discriminator L so that the E network is forced to match q_z to any known distribution p_z, where p_z can be any known continuous random distribution (e. g., Continuous Normal Distribution, or Continuous Uniform Distribution). The L network is trained through differentiating between the true latent $z \sim p_z$ and the fake latent z'_{x_u}.

B. LOSSES

1) ENCODER, CLASSIFIER AND DECODER

For the E and D networks, we have the sample generation loss G_{loss}, the sample reconstruction loss R_{loss}, and the latent generation loss L_{loss}. To calculate the generation and discrimination loss, we use hinge loss, and for the reconstruction loss the Mean Squared Error (MSE) loss. For the G_{loss}, we take the average of the generation loss for $\hat{x_u}$ and $\hat{x_g}$. Therefore,

$$G_{loss} = -\frac{1}{2}(S(\hat{x_u}, \hat{y}_{x_u}; \theta_s) + S(\hat{x_g}, y_r); \theta_s). \quad (1)$$

$$L_{loss} = -(L(z'_{x_u}; \theta_l). \quad (2)$$

$$R_{loss} = \frac{1}{N}\sum_{i=1}^{N}(\hat{x}_{u_i} - x_{u_i})^2. \quad (3)$$

Now, for the C network, we calculate the classification loss Cl_{loss}, Cg_{loss} for the labelled data sample x_l and the generated sample $\hat{x_g}$ respectively. Here, $\hat{x_g}$ is used as a constant, so it is considered like a sample data x_l. We only forward propagate x_u through E and D and no gradient is calculated for generating $\hat{x_g}$ when it is only used for the loss Cg_{loss}. The model is implemented with pytorch [62] and we detach the gradient of x_g when Cg_{loss} is calculated. Therefore,

$$Cl_{loss} = -\sum y_l \log \hat{y}_{x_l}. \quad (4)$$

$$Cg_{loss} = -\sum y_r \log \hat{y}_{\hat{x}_g}. \quad (5)$$

We get the a combined loss EDC_{loss} for E,D and C. The EDC_{loss} is calculated as

$$EDC_{loss} = \alpha \cdot (\omega_1 \cdot G_{loss} + \omega_2 \cdot (\lambda \cdot R_{loss})) + \beta \cdot (\omega_3 \cdot Cl_{loss} + \omega_4 \cdot Cg_{loss} + \omega_5 \cdot L_{loss}). \quad (6)$$

Here, the weights of the E,C, and D networks are updated to minimise the loss EDC_{loss}, where $\omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \alpha, \beta$, and λ are the hyperparameters. The successful training of our GAEE model depends on these parameters. At the beginning of the training, we noticed that the value of R_{loss} falls rapidly compared to other losses and results in a very small gradient value. To mitigate this problem, we multiply R_{loss} with a hyperparameter $\lambda \in \mathbb{R}_{>0}$ and after hyperparameter tuning, we found 20 as an optimal value for λ. The D network of the model is tuned for both the reconstruction loss R_loss and the generation loss G_loss. Therefore, to balance between these two losses, the hyperparameter ω_1 and ω_2 is used where ω_1, $\omega_2 \in [0, 1]$ and $\omega_1 + \omega_2 = 1$. Here, we can force the model to focus more on either loss by increasing the hyperparameter for that particular loss. Likewise, for Cl_{loss}, Cg_{loss} and L_{loss},

we use the hyperparameters ω_3, ω_4, ω_5 respectively, where $\omega_3, \omega_4, \omega_5 \in [0, 1]$ and $\omega_3 + \omega_4 + \omega_5 = 1$. In the EDC_{loss}, G_{loss} and R_{loss} are responsible for the sample generation quality, where Cl_{loss}, Cg_{loss} and L_{loss} are responsible for the latent generation quality. So, to balance between sample generation and latent generation, we use two hyperparameters α and β, where $\alpha, \beta \in [0, 1]$, and $\alpha + \beta = 1$.

2) DISCRIMINATORS' LOSS

For the Discriminators S and L, we use hinge loss. The discrimination loss for the fake samples are averaged as we calculate the loss for both $\hat{x}_u$ and $\hat{x}_g$. Let the discrimination loss for S and L be S_{loss}, L_{loss} respectively. Therefore,

$$\begin{aligned} S_{loss} = & -min(0, -1 + S(x, C(E(x, \theta_e); \theta_c); \theta_s)) \\ & - \frac{1}{2}(min(0, -1 - S(\hat{x}_u, \hat{y}_{x_u}; \theta_s)) \\ & + min(0, -1 - S(\hat{x}_g, \hat{y}_r; \theta_s))). \end{aligned} \tag{7}$$

$$\begin{aligned} L_{loss} = & -min(0, -1 + L(z, \theta_l) \\ & - min(0, -1 - L(\hat{z}_{x_u}, \theta_l)). \end{aligned} \tag{8}$$

Here, we update the parameters θ_s and θ_l to maximise the loss S_{loss} and L_{loss} respectively. Algorithm 1 shows the training mechanism for the GAAE model.

Algorithm 1 Minibatch Stochastic Gradient Descent Training of the Proposed GAAE Model. The Discriminator Is Updated k Times in One Iteration. Here, for Our Experiment, We Use $k = 2$ for Better Convergence

1: **for** number of training iterations **do**
2: **for** k steps **do**
3: Sample the latent/noise samples $\{\boldsymbol{z}^{(1)} \ldots, \boldsymbol{z}^{(m)}\}$ from p_z, the conditions (labels) $\{\boldsymbol{y_r}^{(1)}, \ldots, \boldsymbol{y_r}^{(m)}\}$ from $Cat(y_r)$, the unlabelled data samples $\{\boldsymbol{x_u}^{(1)}, \ldots, \boldsymbol{x_u}^{(m)}\}$ from p_{data} and the labelled data samples $\{\boldsymbol{x_l}^{(1)}, \ldots, \boldsymbol{x_l}^{(m)}\}$ from p_{ldata}. Here, m is the minibatch size.
4: Update the discriminator S by ascending its stochastic gradient:

$$\nabla_{\theta_s} \frac{1}{m} \sum_{i=1}^{m} \left[\boldsymbol{S_{loss}}^{(i)} \right].$$

5: Update the discriminator L by ascending its stochastic gradient:

$$\nabla_{\theta_l} \frac{1}{m} \sum_{i=1}^{m} \left[\boldsymbol{L_{loss}}^{(i)} \right].$$

6: **end for**
7: Repeat step [3].
8: Update the Encoder E, Decoder D, and Classifier C by descending its stochastic gradient:

$$\nabla_{\theta_e, \theta_d, \theta_c} \frac{1}{m} \sum_{i=1}^{m} \left[\boldsymbol{EDC_{loss}}^{(i)} \right].$$

9: **end for**

IV. DATA AND EVALUATION METRICS

A. DATASETS

The effectiveness of the GAAE model is evaluated on both speech and non-speech audios. For the speech audio, we chose the S09 dataset [63] and the Librispeech dataset [64]. For the non-speech audio, we use the popular Nsynth dataset [65]. The S09 dataset consists of utterances for different digit categories from zero to nine. This dataset comprises 23,000 one-second audio samples uttered by 2618 speakers, where it only contains the labels for the audio digits [63].

The Librispeech dataset is an English speech dataset with 1000 hours of audio recordings, and there are three subsets available in the Librispeech dataset containing approximately 100, 300, and 500 hours of recordings, respectively. For our work, we use the subset with 100 hours of clean recordings. In this subset, the audios are uttered by 251 speakers where 125 are female, and 126 are male [64]. For our experiment, we only apply the audios along with the gender labels of the speakers.

The Nsynth audio dataset contains 305,979 musical notes of size four seconds from ten different instruments, where the sources are either acoustic, electronic, or synthetic [65]. We use three acoustic sources: Guitar, Strings, and Mallet from the Nsynth to test the compatibility of the GAAE model for a non-speech dataset.

B. DATA PREPROCESSING

We use the audio of length one second and the sampling rate of 16kHz. For the Librispeech dataset, the one-second audio is taken randomly from any particular audio clip where for the Nsynth dataset, the first one-second is taken from any audio sample as it holds the majority of the instrument sound representation.

The audio data is converted to the log-magnitude spectrograms with the short-time Fourier Transform, and the generated log-magnitude spectrograms of the GAAE model are converted to audio using the PGHI algorithm [58]. In the rest of the paper, we refer to the log-magnitude spectrogram as the spectrogram.

To obtain the spectrogram representation of the audio we followed the procedure from this paper [66]. The short-time Fourier Transform is calculated with an overlapping Hamming window of size 512 ms, and the hopping length 128 ms. Therefore, we get the size of the spectrogram as 256×128, 1D matrix. We standardise the spectrogram with the equation $\frac{X-\mu}{\sigma}$, where X is the spectrogram, μ is the mean of the spectrogram, and σ is the standard deviation of the spectrogram. We clip the dynamic range of the spectrogram at $-r$, where, for the S09 and Librispeech dataset, we determine the suitable value of r to be 10, and for the Nsynth dataset we determine it 15. Here, the log-magnitude

spectrograms is a normal distribution and any inappropriate value of the r can make the distribution skewed, which is not appropriate for training the GAAE network. We investigate the histogram of the values combining all the log-magnitude spectrograms from the whole training dataset to determine the value of r. After the clipping, we normalise the spectrogram values between -1 and 1. The spectrogram representation of the audio is used as the input to the GAAE model, which then generates spectrograms with values between -1 and 1. We then convert these spectrograms to audios via the PGHI algorithm. In this paper we refer to these audios calculated from generated spectrograms as ''generated audios.''

C. MEASUREMENT METRICS

We measure the performance of the GAAE model based on the generated samples and the learnt representations. The generated samples are evaluated with the Inception Score (IS) [67] and Fréchet Inception Distance (FID) [68], [69], which have become a de-facto standard for measuring the performance of any GAN based model [70].

To evaluate the representation/latent learning, we consider classification accuracy, latent space visualisation, and latent interpolation.

1) INCEPTION SCORE (IS)

The IS score is calculated based on the pretrained Inception Network [71] trained on the ImageNet dataset [72]. The logits are calculated for the images from the bottleneck layer of the Inception Network. Then, the score is calculated using

$$\exp(\mathbb{E}_x KL(p(y|x)||p(y))). \tag{9}$$

Here, x is the image sample, KL is the Kullback-Leibler Divergence (KL-divergence) [73], $p(y|x)$ is the conditional class distribution for sample x predicted by the Inception Network, and $p(y)$ is the marginal class distribution. The IS score computes the KL-divergence between the conditional label distribution and the marginal label distribution, **where the higher value indicates good generation quality.**

2) FRÉCHET INCEPTION DISTANCE (FID)

The IS score is computed solely on the generated samples; thus, no comparison is made between the generated and real samples which is not a good measure for the samples' diversity (mode) of the generated samples. The FID score solves this problem by comparing real samples with the generated samples [70] during the score calculation. The Fréchet Inception Distance (FID) computes the Fréchet Distance [74] between two multivariate Gaussian distributions for the generated and real samples, parameterised by the mean and the covariance of the features extracted from the intermediate layer of the pretrained Inception Network. The FID score is calculated using

$$||\mu_r - \mu_g||^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}), \tag{10}$$

where, μ_r, μ_g are the means for the features of the real and generated samples, respectively, and similarly, Σ_r, Σ_g are the covariances, respectively. **A lower value of the FID score indicates good generation quality.**

The Inception Network is trained on the imagenet dataset, thus, offering reliable IS and FID scores for a related image dataset, but the spectrograms of the audios are entirely different from the imagenet samples. So, the Inception Network does not offer trustworthy scores for the audio spectrograms. Hence, instead of using the Inception model, we train a classifier network based on the audio datasets and use this trained classifier to calculate the IS and FID scores. For S09 dataset, we use the pretrained classifier released by the authors of the paper ''Adversarial Audio Synthesis'' [11]. For the Nsynth dataset, we train a simple Convolutional Neural Network (CNN) as the Classifier, as there was no pre-trained classifier available.

V. EXPERIMENTAL SETUP, RESULTS AND DISCUSSION

For implementing our GGAN model, we follow the network implementations, optimisation, and hyperparameters from the BigGAN paper [13]. For the optimisation, we use the Adam optimiser [75]. Learning rate of $5 \cdot 10^{-5}$ is used for the networks E, D, and C, where $2 \cdot 10^{-4}$ is the learning rate for both S and L. Details of the network architectures are given in the appendix (Architectural Details).

A. IMPACT OF LABELLED DATA FOR CONDITIONAL SAMPLE GENERATION

1) SETUP

First, we evaluate the conditional sample generation quality (measured with IS and FID score) of the GAAE model for different percentage of labelled data (1% - 5 %, 100%) as guidance.

The IS and FID scores is calculated based on the 50,000 generated samples [67] for the random latent z, and the random condition y_r. The spectrograms of the samples are generated using the Decoder D network and converted to audios. These generated audios are then used to calculate the IS and FID scores. For all the datasets, we use a continuous normal distribution of size 128 to sample the latent $z \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$. For the S09 dataset, we use the ten digit categories (0-9) as the conditions $y_r \sim Cat(y_r, K = 10, p = 0.1)$. We use the three instrument categories (1-3) as conditions $y_r \sim Cat(y_r, K = 3, p = 0.33)$ for the Nsynth dataset.

For any percentage of data used as guidance, we train the GAAE model three times. Each training takes approximately 60,000 iterations with mixed-precision [76] for the batch size 128. Each time, a dataset is sampled randomly for guidance. Rest of the data is used as unsupervised manner. We limited ourselves to three times due to having high computation time: approximately 21 hours on the two Nvidia p100 GPUs. The total computation time for the S09 and the Nsynth dataset is approximately 21 $\times$ 3 $\times$ 6 (1-5%,100% data) $\times$ 2 (two datasets) = 756 hours or 31.5 days.

The results of the GAAE model are compared with a Supervised BigGAN [77] and an Unsupervised BigGAN [77]. For the S09 dataset, we take the results from the GGAN publication [14]. For the Nsynth dataset, we train these models with a similar setting as was used in the GGAN paper. To calculate the IS and FID score for the Nsynth dataset, we use our pretrained supervised CNN classifier (details in the appendix) trained on three classes: Guitar, Strings, and Mallet.

2) RESULTS AND DISCUSSIONS

The percentage of labelled training data used as guidance has a significant impact on the IS and FID score, which can be found from the table 3. The more we feed the labelled data during the training, the more we boost the performance of the GAAE model for sample generation and diversity. However, notably only with 1% labelled data, the GAAE model achieves acceptable performance. For 5 % labelled data, GAAE achieves scores close to that of using 100% labelled data. So, we compare the scores for 5% data, with other models in the literature.

TABLE 1. Comparison between the sample generation quality of the GAAE model and the other models for the S09 dataset. The generation quality is measured by IS score and FID scores.

Model Name	IS Score	FID Score
Real (Train Data) [11]	9.18 ± 0.04	-
Real (Test Data) [11]	8.01 ± 0.24	-
TiFGAN [67]	5.97	26.7
WaveGAN [11]	4.67 ± 0.01	-
SpecGAN [11]	6.03 ± 0.04	-
Supervised BigGAN	$\mathbf{7.33 \pm 0.01}$	$\mathbf{24.40 \pm 0.50}$
Unsupervised BigGAN	6.17 ± 0.20	24.72 ± 0.05
GGAN [14]	7.24 ± 0.05	25.75 ± 0.10
GAAE	7.28 ± 0.01	22.60 ± 0.07

The results for S09 dataset are summarised in Table 1. Using only 5 % labelled training data as guidance, the GAAE model achieves IS score 7.28 ± 0.01 and FID score of 22.60 ± 0.07. The IS score of GAAE is close to that produced by the supervised BigGAN model (7.33 ± 0.01) and better than other models mentioned in table 1. Even the GAAE model has outperformed the supervised BigGAN model (FID score: 24.40 ± 0.50) in terms of diverse image generation, where the GAAE has used only 5 % labelled data and the supervised BigGAN is trained with all available labelled training data.

For the Nsynth dataset, the GAAE model has achieved the IS score of 2.58 ± 0.03 and the FID score of 141.71 ± 0.32 again with 5 % labelled training data as guidance. Performance of GGAN in terms of IS score is very close to that of the supervised BigGAN (2.64 ± 0.08) and better than that of the unsupervised BigGAN (2.21 ± 0.11). The performance in terms of FID score is even better than that of the supervised BigGAN (148.30 ± 0.23). Table 2 presents the comparisons.

The decoder is trained for both reconstruction and generation of the training data. During the reconstruction, it tries to reconstruct all the training samples, which helps it to learn more modes of the data distribution than the supervised BigGAN model. Figure 3 and 2 display the spectrogram of the generated and the real samples of the Nsynth, S09 datasets, respectively. From these figures, we observe that the generated samples are visually indistinguishable from the real samples. This attests the superior generation quality of the GAAE model. This is also true when we convert these spectrograms to audios. The audios can be found at: https://bit.ly/3coz5qO.

TABLE 2. Comparison between the sample generation quality of the GAAE model and the other models for the Nsynth dataset. The generation quality is measured by IS and FID scores.

Model Name	IS Score	FID Score
Real (Train Data)	2.83 ± 0.02	-
Real (Test Data)	2.81 ± 0.12	-
Supervised BigGAN	$\mathbf{2.64 \pm 0.08}$	$\mathbf{148.30 \pm 0.23}$
Unsupervised BigGAN	2.21 ± 0.11	172.01 ± 0.15
GGAN	2.52 ± 0.06	149.23 ± 0.09
GAAE	2.58 ± 0.03	141.71 ± 0.32

B. EVALUATION OF CONDITIONAL SAMPLE GENERATION BASED ON GUIDANCE

1) SETUP

In this section, we evaluate the effectiveness of guidance for accurate conditional sample generation. It is cumbersome to check all the generation manually. Therefore, we manually check only a few audio samples. For large-scale validation, we use an approach similar to [70]. We train a simple CNN classifier with the samples generated for different random conditions/categories and use the random categories associated with the generated samples as the true labels. Then, we evaluate the CNN classifier on the test dataset based on the classification accuracy. The rationale is that if the GAAE model does not learn to generate correct samples for any given category and the generated samples do not match the training data distribution; the CNN model will not be able to achieve good accuracy on the test dataset. We compare this CNN classifier with another CNN classifier which is trained using all the available training data. For further comparisons, we train two more CNN models with the generated samples from the supervised BigGAN and the GGAN model.

2) RESULT AND DISCUSSION: MANUAL TEST

The generated samples for the S09 and Nsynth dataset are shown in figure 2 and figure 3, respectively. It is not visually evident that the model was able to generate correct samples according to the given conditions/categories. However, when we convert these spectrograms to audios, it is clear that the model is able to generate audios correctly according to the categories demonstrating the effectiveness of the guidance data to learn the specific categorical distribution of the training dataset (cf. under the above link).

3) RESULTS AND DISCUSSIONS: CNN BASED CLASSIFICATION ACCURACY

For the S09 dataset, the test data classification accuracy for the CNN model trained with all the available labelled

TABLE 3. The relationship between the percentage of the data used as guidance during the training and the sample generation quality of the GAAE model, measured with the IS and the FID score. The scores are calculated for the S09 and the Nsynth dataset.

Labelled Data	IS Score (S09)	FID Score (S09)	IS Score (Nsynth)	FID Score(Nsynth)
1%	6.94 ± 0.04	24.21 ± 0.16	2.48 ± 0.08	145.89 ± 1.32
2%	7.06 ± 0.03	23.89 ± 0.11	2.53 ± 0.07	144.21 ± 0.65
3%	7.12 ± 0.04	23.15 ± 0.10	2.56 ± 0.05	143.01 ± 0.43
4%	7.19± 0.02	22.91 ± 0.08	2.57 ± 0.04	142.46 ± 0.38
5%	7.28 ± 0.01	22.60 ± 0.07	2.58 ± 0.03	141.71 ± 0.32
100%	**7.45 ± 0.03**	**19.31 ± 0.01**	**2.67 ± 0.02**	**137.65 ± 0.02**

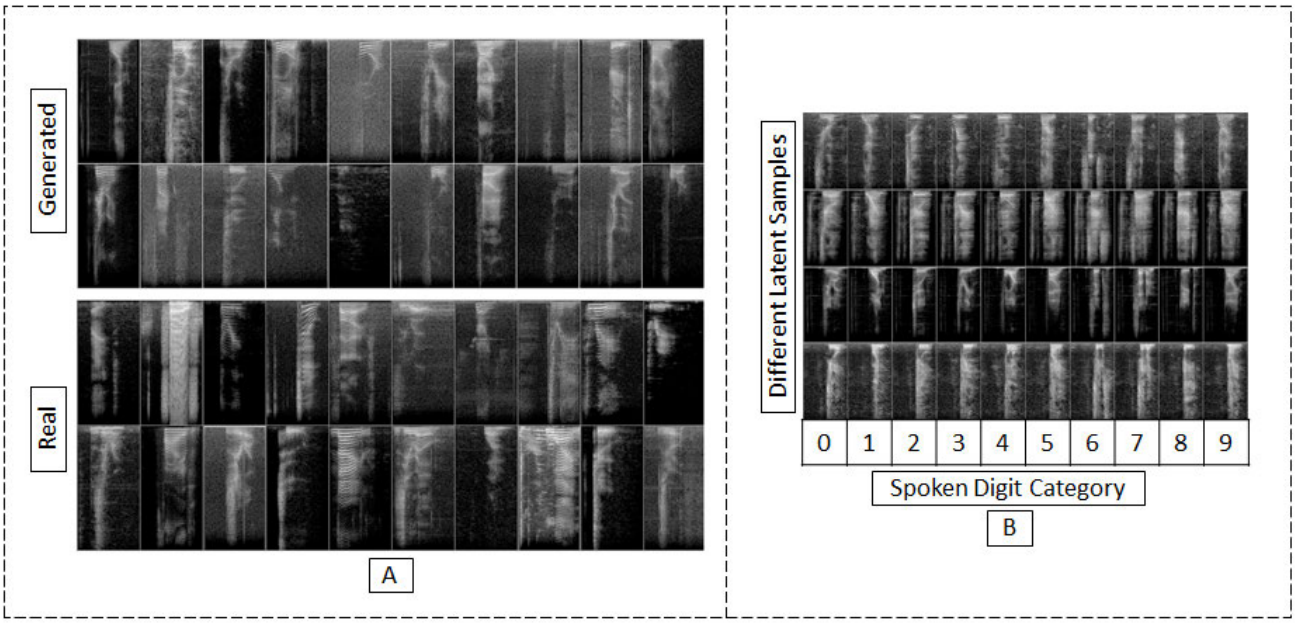


FIGURE 2. A. Illustration of the difference between the generated spectrograms and the real spectrograms of the data for the S09 dataset. The top two rows show the randomly generated samples from the GAAE model, and the bottom two rows are the real samples from the training data. Notice the visual similarity between the generated and the real samples. B. This figure shows the generated spectrograms of the S09 dataset from the GAAE model according to different digit categories. Each row represents the samples generated for a fixed latent variable where the digit condition is changed from 0 to 9. Furthermore, any column shows the generated spectrogram for a particular digit category.

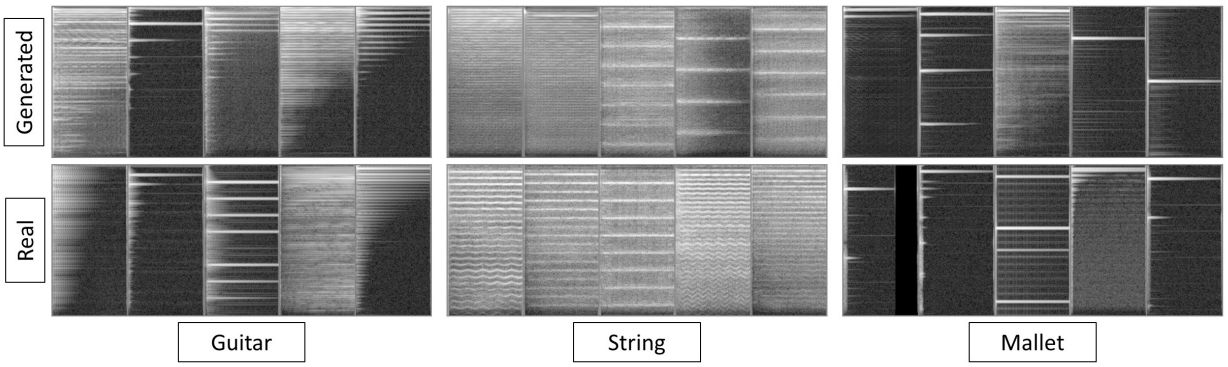


FIGURE 3. Difference between the generated spectrograms of the GAAE model and the real spectrograms of the data for the Nsynth dataset. The top row shows the generated samples, and the bottom row shows the real samples. The first block shows the spectrogram of the guitar, and the other two illustrate the spectrograms for the strings and mallet.

data is 95.52% ± 0.50. The accuracy is 91.14% ± 0.17, when the CNN model is trained based on the generated samples from the GAAE model (trained with 5% labelled data). The table 4 shows the comparison with other models. With the generated samples from the GAAE model, the CNN model achieves greater classification accuracy than the supervised BigGAN (86.58% ± 0.56) and the GGAN model (86.72% ± 0.47).

TABLE 4. The comparison between different CNN classifiers based on the test data classification accuracy from the S09 dataset. The CNN models are trained with the generated samples from different models.

Sample for Training	Test Accuracy
Train Data	95.52% ± 0.50
Supervised BigGAN	86.58% ± 0.56
GGAN	86.72% ± 0.47
GAAE	91.14% ± 0.17
GAAE + Train Data	**97.33% ± 0.19**

TABLE 5. The comparison between different CNN classifiers based on the test data classification accuracy from the Nsynth dataset. The CNN models are trained with the generated samples from different models.

Sample for Training	Test Accuracy
Train Data	92.01% ± 0.94
Supervised BigGAN	83.50% ± 0.62
GGAN	81.40% ± 0.48
GAAE	86.80% ± 0.23
GAAE + Train Data	**94.56% ± 0.09**

When we trained the CNN model mixing the train data, and the generated samples from the GAAE model, the accuracy of the CNN model increased from 95.52% ± 0.50 to 97.33% ± 0.19. Along with the accuracy, the stability of the CNN model is also improved significantly. This can be observed through the standard deviation in the results. We conducted the same evaluation on the Nsynth dataset and received similar results which we present in table 5.

These results demonstrate the superior performance of our GAAE model for generating samples for different categories. It can potentially be used as a data augmentation model where the generated samples from the model can be used to augment any related dataset or same dataset.

C. CONDITIONAL SAMPLE GENERATION USING GUIDANCE FROM A DIFFERENT DATASET

In the above two experiments, we used the guidance data from the same dataset. In this section, we explore the feasibility of guidance from a completely different dataset.

1) SETUP

In the S09 dataset, there are both male and female speakers, but no label is available for the gender of the speakers. We aim to verify if GAAE can generate samples from S09 dataset according to the condition on the gender category, where the guidance comes from a different dataset for gender category. To achieve this, we collect ten male and ten female speakers' audio data (randomly chosen with labels) from the Librispeech dataset to use as guidance during the training with the S09 dataset. During the training of the GAAE model, the guidance data from Librispeech dataset is also merged with S09 dataset as unlabelled data. So, GAAE learns to generate both samples from Librispeech dataset as well as from S09 dataset.

The network we used before to calculate the IS and FID score, is trained on the digit classification tasks for S09 dataset, not for the gender classification task thus will no longer offer a meaningful evaluation. To eradicate this problem, we train another simple CNN model for the gender classification to calculate the IS and the FID score. For this purpose, we randomly select 15 male and 15 female speakers from Librispeech dataset. We use data from ten male and ten female speakers for training and data from others for testing. We achieve an accuracy of 98.3 ± 0.50. We use this model to calculate the IS and FID Score for the generated samples from different models. Now, the calculated scores will reflect the quality of the generated samples according to gender distribution.

We define two GAAE models: one is trained with gender guidance, and another is trained with digit guidance. We compare the IS and FID score of these models. Note that gender information is being collected from a different dataset: Librispeech. If the gender guided model achieves better score, then we can establish the feasibility of guidance using an external dataset. To further validate this, we add results from other models (Unsupervised BigGAN, Supervised BigGAN and GGAN) trained based on digit guidance.

We choose a continuous normal distribution of size 128 for latent $z \sim \mathcal{N}(\mu = 0,\ \sigma^2 = 1)$ and two gender categories for the conditions $y_r \sim Cat(y_r, K = 2, p = 0.5)$.

2) RESULTS AND DISCUSSIONS

The calculated scores are presented in table 6. Gender guided GAAE produces the best FID and IS scores, which establish that it is feasible to get guidance from a different dataset in the GAAE model.

TABLE 6. Comparison between the performance of the GAAE model trained with gender guidance and the other models on the S09 dataset, in terms of the quality of the generated samples based on the gender attributes of the speaker, measured with the IS and the FID score.

Model Name	IS Score	FID Score
Train Data	1.92 ± 0.04	-
Test Data	1.91 ± 0.05	-
Unsupervised BigGAN	1.13 ± 0.89	56.01 ± 0.85
Supervised BigGAN	1.48 ± 0.56	35.22 ± 0.50
GGAN (Digit Guided)	1.58 ± 0.05	37.75 ± 0.10
GAAE (Digit Guided)	1.61 ± 0.17	29.84 ± 0.43
GAAE (Gender Guided)	**1.78 ± 0.03**	**20.21 ± 0.01**

D. GUIDED REPRESENTATION LEARNING

The GAAE model learns two types of representations/latent spaces: (1) it uses $z_{x_u} \sim u_z$ to learn guidance specific characteristics of the data (Guided representation) and uses (2) $z'_{x_u} \sim q_z$ to learn general characteristics of the data (General representation/Style representation).

1) SETUP

In the GAAE model, the Classifier C is built on top of the latent $z_{x_u} \sim u_z$ (see Fig. 1). The encoder network E, therefore, learns this latent variable to disentangle the class categories according to the guided data. For the S09 dataset, we use digit

classes as guidance, so, in this latent space (representation space), the digit category should be disentangled. To observe this disentanglement, we visualise the higher dimensional (128) latent space generated for the S09 test data in the 2D plane with the t-SNE (t-distributed stochastic neighbour embedding) [78] visualisation method. We use the same visualisation for the Nsynth dataset.

2) RESULTS AND DISCUSSIONS

Figure 4 shows the representation space for S09 test dataset and figure 5 shows the visualisation for the Nsynth dataset. From both figures, it is noticeable that the guided categories are well separated in the representation space, and data points of the similar categories are clustered together. So, the encoder E learns to map the data sample to the representation space u_z ensuring data categories used as guidance are well separable in the representation space.

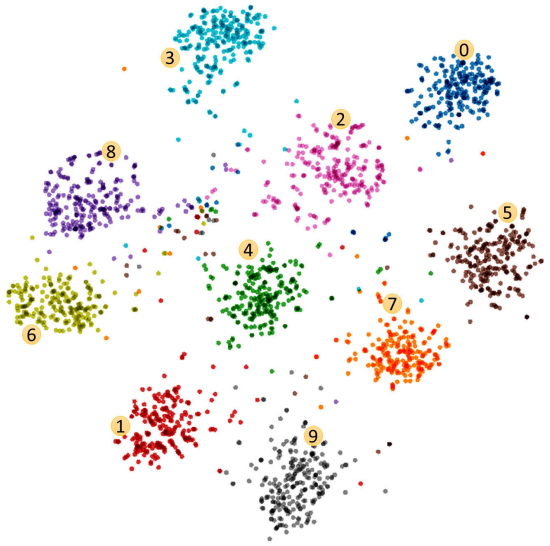


FIGURE 4. t-SNE visualisation of the learnt representation of the test data of the S09 dataset. Here, different colours of points represent different digit categories. In the representation space, the different digit categories are clustered together and easily separable.

E. GENERAL REPRESENTATION/STYLE REPRESENTATION LEARNING

1) SETUP

The encoder network E of the GAAE model is trained to match the q_z distribution with the known p_z distribution. This allows sampling z'_{x_u} from the q_z distribution.

Now, it is expected that when Decoder D learns to generate samples from the latent space q_z, it disentangles the general characteristics/attributes (independent of the guided attributes) of the data in the q_z latent space. To evaluate this disentanglement in the representation space $z'_{x_u} \sim q_z$ for both S09 and Nsynth dataset, we generate audio samples for different categories/conditions keeping the z'_{x_u} the same.

In our model, Decoder can achieve disentanglement implies that the pretrained E extracts general attributes in

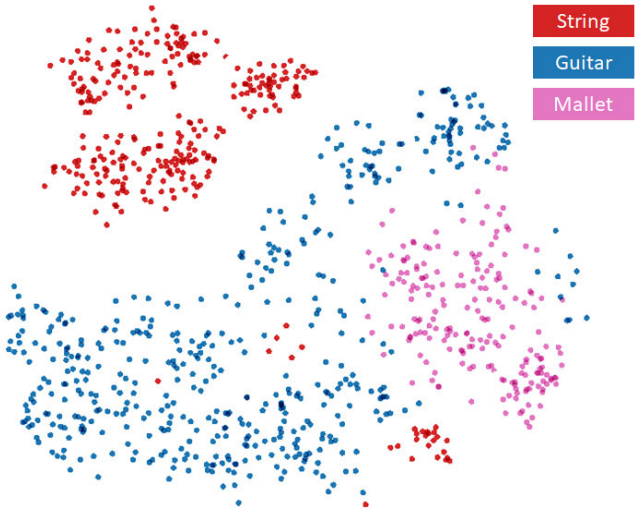


FIGURE 5. t-SNE visualisation of the learnt representation of the test data of the Nsynth dataset. Here, different colours of points represent different instrument categories. In the representation space, the different instrument categories are clustered together and easily separable.

latent z'_{x_u} from any related dataset, which was not used during the training. To validate, we pass the test data from S09 and Nsynth dataset through E to get the general representation z'_{x_u}. Then for a fixed z'_{x_u} and different conditions (digit categories), we generate samples from the pretrained D network.

As the GAAE model learns general/style attributes in the z'_{x_u} latent space, it should disentangle the gender of the speaker in the latent space for S09 dataset. To evaluate this, we use the trained E network from the GAAE model to extract latent representation z'_{x_u} for an entirely different Librispeech dataset where gender labels are available. For 5000 randomly sampled data from the Librispeech dataset, we extract the feature/latent z'_{x_u} from E and visualise the result in 2D plain using t-SNE visualisation for exploration.

2) RESULTS AND DISCUSSIONS

After investigating the generated audios of the S09 dataset, the digit categories are changed according to the given condition y_r and the general characteristics (such as the voice of the speaker, audio pitch, background noise etc.) of the audio is changed with the change of z'_{x_u}. So, the D network learns to capture general attributes of the data in the latent space z'_{x_u}. For the Nsynth dataset, we notice a similar behaviour.

We investigate the audio samples generated based on the extracted feature z'_{x_u} of the input data sample. Exploration of the audios shows that they preserve some characteristics (like speaker gender, voice, pitch, tone, background noise etc. for S09 test data) from the input data sample. We also notice similar scenarios for the Nsynth dataset. The audios can be found at: https://bit.ly/36Oz9z9. Note that the initial one second is the input audio data and rest are the generated audios.

Figure 6 shows the visualisation of the extracted representation for the Librispeech dataset. We observe that the

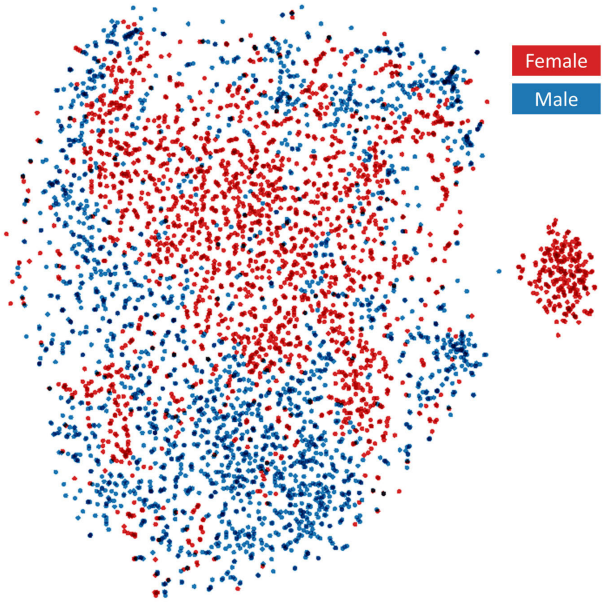


FIGURE 6. t-SNE visualisation of the learnt representation of the Libri speech dataset. Here, different colours of points represent the gender of the speakers. The representations of the different gender categories are clustered together.

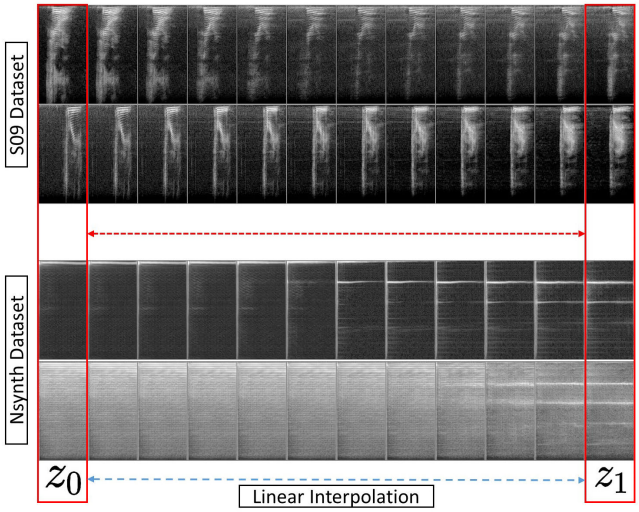


FIGURE 7. Generated spectrograms based on the linear interpolation between two latent samples; z_0 and z_1. The first two rows show the generated spectrograms for the S09 dataset (one and zero) and the bottom two rows exhibit the spectrograms for the Nsynth dataset (mallet and string). For any particular row, the first and the last spectrograms are the generations based on the fixed two latent points and the in-between spectrograms are the generation based on the interpolation between these two fixed points.

latent representation for the same gender of the speakers are clustered together and are easily separable from the latent space. This exploration exhibits that the GAAE model is able to learn the gender attributes of the speaker from the S09 dataset successfully even though gender information of the speaker was never used during the training.

F. COHERENCE OF THE GENERAL REPRESENTATION/ LATENT SPACE

1) SETUP

It is expected that the D network can learn the latent space q_z in a way so that it is coherent and if we move in any direction in the latent space the generated samples should be changed accordingly. To investigate this, we conduct linear interpolation between two latent points as described in the DCGAN paper [3]. A particular point z_i within two latent points z_0 and z_1 is calculated with the equation $z_i = z_0 + \eta(z_0 - z_1)$, where η is the step size from z_0 to z_1. With this equation, we get the latent points in between z_0 and z_1. Using this D network, we obtain the generated samples for these latent points, where the random categorical condition y_r is fixed.

2) RESULTS

Figure 7 shows the generated samples for both the S09 and Nsynth datasets based on the interpolated points. We observe that the transition between two spectrograms generated based on two fixed latent samples z_0 and z_1 is very smooth. Moreover, when we convert the spectrograms to audio, we observe the same smooth transition, which indicates the disentanglement of the general attributes in the latent space q_z The audios can be found at: https://bit.ly/2yPcTIE.

VI. HYPERPARAMETER TUNING

We tune the hyperparameters based on the S09 dataset as tuning is resource and time-intensive. We then use the hyperparameters for other datasets. From equation 6, ω_1, and ω_2 are two important hyperparameters for training the GAAE model, where $\omega_2 = 1$ - ω_1. When we increase ω_1, the model focuses more on the generation loss G_{loss} and less on the reconstruction loss R_{loss}. If we reduce ω_1, the model increases the focus for reconstruction and reduces the focus for the generation. The impact of ω_1 and ω_2 on the IS scores, FID scores, and classification accuracy are presented in figure 8. The best value for ω_1 is 0.6 and for ω_2, it is 0.4.

The α and β from equation 6 are two other important hyperparameters. The value of the α parameter determines how much the model will focus on generation (G_{loss}) and reconstruction loss (R_{loss}), where the β parameter determines the focus for the classification (Cl_{loss}, Cg_{loss}) and latent generation loss (L_{loss}). From figure 8, we observe that 0.5 is the best value for both of the hyperparameters.

There are three more hyperparameters: ω_3, ω_4, and ω_5 (See equation 6). Here, ω_3 and ω_4 control the classification loss (Cl_{loss}, Cg_{loss}) for labelled data. And, ω_5 controls the latent generation loss (L_{loss}). Here, we maintain equal balance between the classification and the latent generation loss. Likewise, we use 0.25 for ω_3,ω_4 and 0.50 for ω_5.

VII. CLASSIFIER OF THE GAAE MODEL

The success of the GAAE model is mostly dependent on its internal Classifier C. In this section, we evaluate the performance of C. We benchmark its performance using a supervised Classifier, the Classifier from GGAN and the Classifier

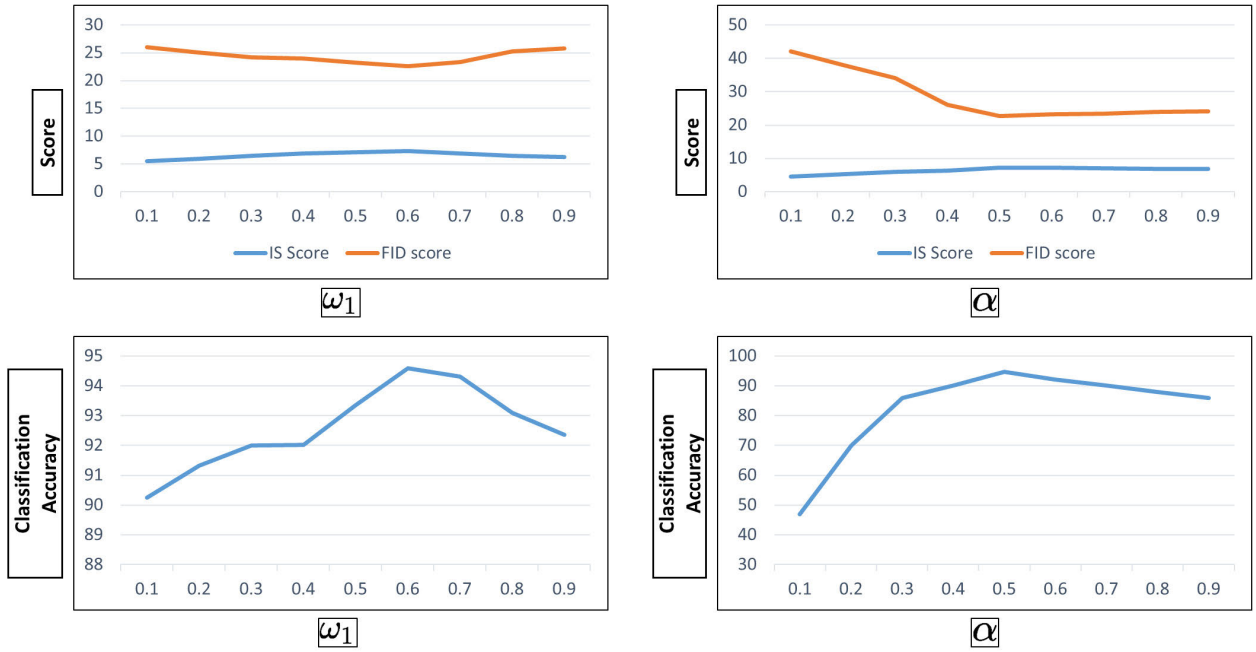


FIGURE 8. Relationship between the hyperparameters and the measurement metrics of the GAAE model. The top left plot explains the relationship between ω_1 and IS and FID scores. Similarly, the top right explicates the relationship between α and IS and FID scores. Here, The bottom left box illustrates the relationship between ω_1 and the classification accuracy. Furthermore, the bottom right plot demonstrates the impact of α on the classification accuracy.

TABLE 7. Relationship between the percentage of the data used as the guidance during the training and the S09 test dataset classification accuracy of the GAAE model.

Training Data Size	CNN Network	BiGAN	GGAN	GAAE
1%	82.21 ± 1.2	73.01 ± 1.02	84.21 ± 2.24	90.21 ± 0.16
2%	83.04 ± 0.34	75.56 ± 0.41	85.39 ± 1.24	91.45 ± 0.12
3%	83.78 ± 0.23	78.33 ± 0.07	88.25 ± 0.10	92.67 ± 0.06
4%	84.11 ± 0.34	80.03 ± 0.01	91.02 ± 0.50	93.70 ± 0.05
5%	84.50 ± 1.02	80.84 ± 1.72	92.00 ± 0.87	94.59 ± 0.03
100%	**95.52 ± 0.50**	**86.77 ± 2.61**	**96.51± 0.07**	**97.68 ± 0.01**

TABLE 8. Relationship between the percentage of the data used as the guidance during the training and the Nsynth test dataset classification accuracy of the GAAE model.

Training Data Size	CNN Network	BiGAN	GGAN	GAAE
1%	85.76 ± 1.10	82.21 ± 0.84	88.52 ± 0.32	90.26 ± 0.09
2%	89.79 ± 0.51	86.65 ± 0.57	91.69 ± 0.24	92.96 ± 0.07
3%	89.83 ± 0.49	87.21 ± 0.46	91.95 ± 0.20	93.12 ± 0.05
4%	90.52 ± 0.25	87.59 ± 0.41	92.16 ± 0.19	93.73 ± 0.02
5%	91.07 ± 0.31	87.95 ± 0.39	92.45 ± 0.14	94.23 ± 0.02
100%	**92.01 ± 0.94**	**88.09 ± 0.24**	**93.56 ± 0.09**	**94.89 ± 0.01**

from BiGAN [55]. For the supervised Classifier, we train a simple CNN classifier using 1 % - 5 %, 100 %, of training data, where the data is heavily augmented using techniques like adding random noise, rotation of the spectrogram, multiplication with random zero patches, etc. ([79]). We train a BiGAN model on top of the unsupervised BigGAN and extract BiGANs' feature network after the training. We then train another feed-forward classifier network on BiGANs' feature network using similar percentages of labelled data. We keep the weights for the feature network fixed during the training. We evaluate all these Classifiers using the test dataset. As the Classifier C of the GAAE model is trained with fewer labelled data along with the generated samples from the decoder D, it will only perform better if generation is accurate according to the different categories and the quality of the generated samples is close to the real samples.

The relationship between the percentage of the data used as guidance and the test data classification accuracy is shown in table 7, 8 for S09 and Nsyth dataset, respectively. Results from both tables demonstrate that the GAEE model

outperforms other models in terms of classification accuracy leveraging the minimal amount of labelled data (average 5%-8% percent improvement for both datasets while using 1% labelled data).

VIII. CONCLUSION AND LESSON LEARNT

In this paper, we propose the Guided Adversarial Autoencoder (GAAE), which is capable of generating high-quality audio samples using very few labelled data as guidance. After evaluating the GAAE model using two audio datasets: S09 and Nsynth, we show that the GAAE model can outperform the existing models with respect to sample generation quality and mode diversity. Harnessing the power of high-fidelity audio generation, the GAAE model can disentangle the specific attributes of the data in the learnt latent/representation space according to the guidance. This learnt representation can be beneficial to any related downstream task at hand. We also show that besides the guided representation learning, the GAAE model learns to disentangle other attributes of the data independent of the given guidance. Hence, the GAAE model learns a representation for the specific downstream task at hand and a generalised representation for future unknown related tasks.

We evaluate the GAAE model based on the audio of size one second; thus, it remains a challenge to make this model work for longer audio sample generation. In representation learning, the GAAE model can be used efficiently for any long audio sample by dividing it into one-second chunks. GAAE model successfully learns generation and representation using a minimum of 1 % labelled data. We believe this will encourage other researchers to explore the GAAE model further for few-shot learning.

Furthermore, we built the GAAE model based on BigGAN architecture. This leaves an excellent opportunity for studying other high performing GAN architectures such as progressive GAN [80] or the Style GAN [9].

APPENDIX
ARCHITECTURAL DETAILS

This section presents the details of the neural networks used in this paper. We follow the abbreviations and description style from the original work of Mario *et al.* [15].

A. SUPERVISED BigGAN

We use the exact implementation of the Supervised BigGAN from our former GGAN paper [14]. Therefore, for the implementation of both the Generator and the Discriminator, we apply a Resnet architecture from the BigGAN work [13]. The layers are shown tables 10 and 11. The Generator and Discriminator architectures are shown in Tables 12 and 13, respectively. We use a learning rate of 0.00005 and 0.0002 for the Generator and the Discriminator, respectively. We set the number of channels (ch) to 16 to minimise the computational expenses, as the higher number of channels such as 64 and 32 only offer negligible improvements.

TABLE 9. Abbreviations for defining the architectures.

Full Name	Abbreviation
Resample	RS
Batch normalisation	BN
Conditional batch normalisation	cBN
Downscale	D
Upscale	U
Spectral normalisation	SN
Input height	h
Input width	w
True label	y
Input channels	ci
Output channels	co
Number of channels	ch

TABLE 10. Architecture of the ResBlock generator with upsampling for the supervised BigGAN.

Layer Name	Kernal Size	RS	Output Size
Shortcut	[1,1,1]	U	2h × 2w × c_{o}
cBN, ReLU	-	-	h × w × c_{i}
Convolution	[3,3,1]	U	2h × 2w × c_{o}
cBN, ReLU	-	-	2h × 2w × c_{o}
Convolution	[3,3,1]	U	2h × 2w × c_{o}
Addition	-	-	2h × 2w × c_{o}

TABLE 11. Architecture of the ResBlock discriminator with downsampling for the supervised BigGAN.

Layer Name	Kernal Size	RS	Output Size
Shortcut	[1,1,1]	D	h/2 × w/2 × c_{o}
ReLU	-	-	h × w × c_{i}
Convolution	[3,3,1]	-	h × w × c_{o}
ReLU	-	-	h × w × c_{o}
Convolution	[3,3,1]	D	h/2 × w/2 × c_{o}
Addition	-	-	h/2 × w/2 × c_{o}

TABLE 12. Architecture of the generator for the supervised BigGAN.

Layer Name	RS	SN	Output Size
Input z	-	-	128
Dense	-	-	4 × 2 × 16. ch
ResBlock	U	SN	8 × 4 × 16. ch
ResBlock	U	SN	16 × 8 × 16. ch
ResBlock	U	SN	32 × 16 × 16. ch
ResBlock	U	SN	64 × 32 × 16. ch
ResBlock	U	SN	128 × 64 × 16. ch
Non-local block	-	-	128 × 64 × 16. ch
ResBlock	U	SN	256 × 128 × 1. ch
BN, ReLU	-	-	256 × 128 × 1
Conv [3, 3, 1]	-	-	256 × 128 × 1
Tanh	-	-	256 × 128 × 1

B. UNSUPERVISED BigGAN

Similarly, for the unsupervised BigGAN, follow the same implementation from the original GGAN work [14]. Tables 14 and 15 show the upsampling and downsampling layers, respectively. The architectures of the Generator and Discriminator are shown in the tables 16 and 17, respectively.

TABLE 13. Architecture of the discriminator for the supervised BigGAN.

Layer Name	RS	Output Size
Input Spectrogram	-	256 × 128 × 1
ResBlock	D	128 × 64 × 1. ch
Non-local block	-	128 × 64 × 1. ch
ResBlock	-	64 × 32 × 1. ch
ResBlock	D	32 × 16 × 2. ch
ResBlock	D	16 × 8 × 4. ch
ResBlock	D	8 × 4 × 8. ch
ResBlock	D	4 × 2 × 16. ch
ResBlock (No Shortcut)	-	4 × 2 × 16. ch
ReLU	-	4 × 2 × 16. ch
Global sum pooling	-	1 × 1 × 16. ch
Sum(embed(y)·h)+(dense → 1)	-	1

TABLE 14. Architecture of the ResBlock generator with upsampling for the unsupervised BigGAN.

Layer Name	Kernal Size	RS	Output Size
Shortcut	[1,1,1]	U	2h × 2w × c_{o}
BN, ReLU	-	-	h × w × c_{i}
Convolution	[3,3,1]	U	2h × 2w × c_{o}
BN, ReLU	-	-	2h × 2w × c_{o}
Convolution	[3,3,1]	U	2h × 2w × c_{o}
Addition	-	-	2h × 2w × c_{o}

The learning rate and channels are the same as for the supervised BigGAN.

C. BiGAN

For the BiGAN model, we train a Feature Extractor and Discriminator network on top of the unsupervised BigGAN. The Feature Extractor network creates the features for real samples, and the Discriminator tries to differentiate between the generated features and the random noise. The detail is exactly followed from the original BiGAN work [55]. The downsampling layer is the same as the unsupervised BigGAN and can be found in table 15. The architecture of the Feature Extractor network is shown in table 18. Furthermore, the architecture of the Discriminator is given in table 19.

TABLE 15. Architecture of the ResBlock discriminator with downsampling for the unsupervised BigGAN.

Layer Name	Kernal Size	RS	Output Size
Shortcut	[1,1,1]	D	h/2 × w/2 × c_{o}
ReLU	-	-	h × w × c_{i}
Convolution	[3,3,1]	-	h × w × c_{o}
ReLU	-	-	h × w × c_{o}
Convolution	[3,3,1]	D	h/2 × w/2 × c_{o}
Addition	-	-	h/2 × w/2 × c_{o}

D. GAAE

In the GAAE model, the downsampling and upsampling layers are the same as those shown in table 10 and 11, respectively.

TABLE 16. Architecture of the generator for the unsupervised BigGAN.

Layer Name	RS	SN	Output Size
Input z	-	-	128
Dense	-	-	4 × 2 × 16. ch
ResBlock	U	SN	8 × 4 × 16. ch
ResBlock	U	SN	16 × 8 × 16. ch
ResBlock	U	SN	32 × 16 × 16. ch
ResBlock	U	SN	64 × 32 × 16. ch
ResBlock	U	SN	128 × 64 × 16. ch
Non-local block	-	-	128 × 64 × 16. ch
ResBlock	U	SN	256 × 128 × 1. ch
BN, ReLU	-	-	256 × 128 × 1
Conv [3, 3, 1]	-	-	256 × 128 × 1
Tanh	-	-	256 × 128 × 1

TABLE 17. Architecture of the discriminator for the unsupervised BigGAN.

Layer Name	RS	Output Size
Input Spectrogram	-	256 × 128 × 1
ResBlock	D	128 × 64 × 1. ch
Non-local block	-	128 × 64 × 1. ch
ResBlock	-	64 × 32 × 1. ch
ResBlock	D	32 × 16 × 2. ch
ResBlock	D	16 × 8 × 4. ch
ResBlock	D	8 × 4 × 8. ch
ResBlock	D	4 × 2 × 16. ch
ResBlock (No Shortcut)	-	4 × 2 × 16. ch
ReLU	-	4 × 2 × 16. ch
Global sum pooling	-	1 × 1 × 16. ch
Dense	-	1

TABLE 18. Architecture of the Feature Extractor Network for the BiGAN.

Layer Name	RS	Output Size
Input Spectrogram	-	256 × 128 × 1
ResBlock	D	128 × 64 × 1. ch
Non-local block	-	128 × 64 × 1. ch
ResBlock	-	64 × 32 × 1. ch
ResBlock	D	32 × 16 × 2. ch
ResBlock	D	16 × 8 × 4. ch
ResBlock	D	8 × 4 × 8. ch
ResBlock	D	4 × 2 × 16. ch
ResBlock (No Shortcut)	-	4 × 2 × 16. ch
ReLU	-	4 × 2 × 16. ch
Global sum pooling	-	1 × 1 × 16. ch
Dense	-	128

The Encoder architecture is given in table 20, where we use two dense layers to obtain z_{x_u} and z'_{x_u} from a global sum pooling layer. For the Decoder, the conditional vector y_r or $\hat{y}_{x_u}$ is given through the conditional Batch Normaliser (cBN) from the upsampling layer. The classifier network is built upon some dense layer, and the architecture is given in table 22. For the Sample Discriminator, we exactly follow the implementation in table 13. Here, in the table 13, y is the conditional vector, and h is the output from the global sum pooling layer. For the Latent Discriminator, we have

TABLE 19. Architecture of the Discriminator for the BiGAN.

Layer Name	RS	Output Size
Input Spectrogram	-	256 × 128 × 1
ResBlock	D	128 × 64 × 1. ch
Non-local block	-	128 × 64 × 1. ch
ResBlock	-	64 × 32 × 1. ch
ResBlock	D	32 × 16 × 2. ch
ResBlock	D	16 × 8 × 4. ch
ResBlock	D	8 × 4 × 8. ch
ResBlock	D	4 × 2 × 16. ch
ResBlock (No Shortcut)	-	4 × 2 × 16. ch
ReLU	-	4 × 2 × 16. ch
Global sum pooling	-	1 × 1 × 16. ch
Concat with input feature	-	256+128=384
Dense	-	128
ReLU	-	128
Dense	-	1

TABLE 20. Architecture of the Encoder for the GAAE.

Layer Name	RS	Output Size
Input Spectrogram	-	256 × 128 × 1
ResBlock	D	128 × 64 × 1. ch
Non-local block	-	128 × 64 × 1. ch
ResBlock	-	64 × 32 × 1. ch
ResBlock	D	32 × 16 × 2. ch
ResBlock	D	16 × 8 × 4. ch
ResBlock	D	8 × 4 × 8. ch
ResBlock	D	4 × 2 × 16. ch
ResBlock (No Shortcut)	-	4 × 2 × 16. ch
ReLU	-	4 × 2 × 16. ch
Global sum pooling	-	1 × 1 × 16. ch
Dense (z_{x_u}), Dense (z'_{x_u})	-	128, 128

TABLE 21. Architecture of the Decoder for the GAAE.

Layer Name	RS	SN	Output Size
Input latent vector	-	-	128
Dense	-	-	4 × 2 × 16. ch
ResBlock	U	SN	8 × 4 × 16. ch
ResBlock	U	SN	16 × 8 × 16. ch
ResBlock	U	SN	32 × 16 × 16. ch
ResBlock	U	SN	64 × 32 × 16. ch
ResBlock	U	SN	128 × 64 × 16. ch
Non-local block	-	-	128 × 64 × 16. ch
ResBlock	U	SN	256 × 128 × 1. ch
BN, ReLU	-	-	256 × 128 × 1
Conv [3, 3, 1]	-	-	256 × 128 × 1
Tanh	-	-	256 × 128 × 1

use multi dense layers, and the architecture is given in table 23.

The learning rates for both Discriminators are 0.0002, and for other networks, the learning rate is 0.00005. We set the number of channels to 16 for all the experiment carried out with the GAAE.

TABLE 22. Architecture of the Classifier for the GGAN.

Layer Name	Output Size
Input latent vector	128
Dense	128
ReLU	128
Dense	10

TABLE 23. Architecture of the Latent Discriminator for the GGAN.

Layer Name	Output Size
Input latent vector	128
Dense	128
ReLU	128
Dense	128
ReLU	128
Dense	1

TABLE 24. Architecture of the Simple Spectrogram Classifier.

Layer Name	Output Size
Input Spectrogram	256 × 128 × 1
Convolution [3, 3, 32]	256 × 128 × 32
Maxpool [2, 2]	128 × 64 × 32
Convolution [3, 3, 64]	128 × 64 × 64
Maxpool [2, 2]	64 × 32 × 64
Convolution [3, 3, 128]	64 × 32 × 128
Maxpool [2, 2]	32 × 16 × 128
Convolution [3, 3, 256]	32 × 16 × 256
Maxpool [2, 2]	16 × 8 × 256
Dense	c

E. SIMPLE CLASSIFIER

For many classification tasks, we mention a Simple Classifier throughout the paper. The architecture of these classifiers are as in table 24. Here, c is the number of outputs according to the classification categories. The learning rates is used as 0.0001 for this classifier network.

REFERENCES

[1] F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. F. Bachem, "Challenging common assumptions in the unsupervised learning of disentangled representations," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4114–4124.

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2014, pp. 2672–2680.

[3] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, pp. 1–16, Nov. 2015.

[4] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.

[5] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using WaveNet autoencoders," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 12, pp. 2041–2053, Dec. 2019.

[6] J. Donahue and K. Simonyan, "Large scale adversarial representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 10541–10551.

[7] S. Zhao, J. Song, and S. Ermon, "InfoVAE: Information maximizing variational autoencoders," 2017, *arXiv:1706.02262*. [Online]. Available: http://arxiv.org/abs/1706.02262
[8] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," 2015, *arXiv:1511.05644*. [Online]. Available: http://arxiv.org/abs/1511.05644
[9] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4401–4410.
[10] J. Engel, K. Krishna Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "GANSynth: Adversarial neural audio synthesis," 2019, *arXiv:1902.08710*. [Online]. Available: http://arxiv.org/abs/1902.08710
[11] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–16.
[12] A. Marafioti, N. Holighaus, N. Perraudin, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," 2019, *arXiv:1902.04072*. [Online]. Available: http://arxiv.org/abs/1902.04072
[13] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," *CoRR*, vol. abs/1809.11096, pp. 1–35, Sep. 2018.
[14] K. Nazmul Haque, R. Rana, J. H. L. Hansen, and B. Schuller, "Guided generative adversarial neural network for representation learning and high fidelity audio generation using fewer labelled audio data," 2020, *arXiv:2003.02836*. [Online]. Available: http://arxiv.org/abs/2003.02836
[15] M. Lucic, M. Tschannen, M. Ritter, X. Zhai, O. Bachem, and S. Gelly, "High-fidelity image generation with fewer labels," 2019, *arXiv:1903.02271*. [Online]. Available: http://arxiv.org/abs/1903.02271
[16] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, New York, NY, USA, vol. 1. Red Hook, NY, USA: Curran Associates, 2013, pp. 899–907.
[17] R. Zhang, P. Isola, and A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis.*, vol. 9907, Oct. 2016, pp. 649–666.
[18] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 577–593.
[19] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1422–1430.
[20] K. Nazmul Haque, M. Abu Yousuf, and R. Rana, "Image denoising and restoration with CNN-LSTM encoder decoder with direct attention," 2018, *arXiv:1801.05141*. [Online]. Available: http://arxiv.org/abs/1801.05141
[21] X. Zhan, X. Pan, Z. Liu, D. Lin, and C. C. Loy, "Self-supervised learning via conditional motion propagation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1881–1889.
[22] Z. Feng, C. Xu, and D. Tao, "Self-supervised representation learning by rotation feature decoupling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10364–10374.
[23] S. Liu, A. Davison, and E. Johns, "Self-supervised generalisation with meta auxiliary learning," in *Proc. Adv. Neural Inf. Process. Syst.*, pp. 1679–1689, 2019.
[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*. [Online]. Available: http://arxiv.org/abs/1810.04805
[25] J. Wu, X. Wang, and W. Yang Wang, "Self-supervised dialogue learning," 2019, *arXiv:1907.00448*. [Online]. Available: http://arxiv.org/abs/1907.00448
[26] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai, "VL-BERT: Pre-training of generic visual-linguistic representations," 2019, *arXiv:1908.08530*. [Online]. Available: http://arxiv.org/abs/1908.08530
[27] H. Wang, X. Wang, W. Xiong, M. Yu, X. Guo, S. Chang, and W. Y. Wang, "Self-supervised learning for contextualized extractive summarization," 2019, *arXiv:1906.04466*. [Online]. Available: http://arxiv.org/abs/1906.04466
[28] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," *CoRR*, vol. abs/1803.07728, pp. 1–16, Oct. 2018.
[29] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*. [Online]. Available: http://arxiv.org/abs/1807.03748
[30] F. de Chaumont Quitry, M. Tagliasacchi, and D. Roblek, "Learning audio representations via phase prediction," 2019, *arXiv:1910.11910*. [Online]. Available: http://arxiv.org/abs/1910.11910
[31] A. Nagrani, J. S. Chung, S. Albanie, and A. Zisserman, "Disentangled speech embeddings using cross-modal self-supervision," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 6829–6833.
[32] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "Wav2vec: Unsupervised pre-training for speech recognition," 2019, *arXiv:1904.05862*. [Online]. Available: http://arxiv.org/abs/1904.05862
[33] Y.-A. Chung and J. Glass, "Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech," 2018, *arXiv:1803.08976*. [Online]. Available: http://arxiv.org/abs/1803.08976
[34] Y.-A. Chung, C.-C. Wu, C.-H. Shen, H.-Y. Lee, and L.-S. Lee, "Audio Word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder," 2016, *arXiv:1603.00982*. [Online]. Available: http://arxiv.org/abs/1603.00982
[35] K. Kawakami, L. Wang, C. Dyer, P. Blunsom, and A. van den Oord, "Learning robust and multilingual speech representations," 2020, *arXiv:2001.11128*. [Online]. Available: http://arxiv.org/abs/2001.11128
[36] M. Riviere, A. Joulin, P.-E. Mazare, and E. Dupoux, "Unsupervised pretraining transfers well across languages," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 7414–7418.
[37] A. Baevski, S. Schneider, and M. Auli, "Vq-wav2vec: Self-supervised learning of discrete speech representations," 2019, *arXiv:1910.05453*. [Online]. Available: http://arxiv.org/abs/1910.05453
[38] A. Baevski, M. Auli, and A. Mohamed, "Effectiveness of self-supervised pre-training for speech recognition," 2019, *arXiv:1911.03912*. [Online]. Available: http://arxiv.org/abs/1911.03912
[39] S. Latif, R. Rana, S. Khalifa, R. Jurdak, J. Qadir, and B. W. Schuller, "Deep representation learning in speech processing: Challenges, recent advances, and future trends," 2020, *arXiv:2001.00378*. [Online]. Available: http://arxiv.org/abs/2001.00378
[40] S. Amiriparian, M. Freitag, N. Cummins, and B. Schuller, "Sequence to sequence autoencoders for unsupervised representation learning from audio," in *Proc. Workshop (DCASE)*, 2017, pp. 1–5.
[41] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1096–1104.
[42] Y. Xu, Q. Huang, W. Wang, P. Foster, S. Sigtia, P. J. B. Jackson, and M. D. Plumbley, "Unsupervised feature learning based on deep models for environmental audio tagging," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 6, pp. 1230–1241, Jun. 2017.
[43] M. Neumann and N. T. Vu, "Improving speech emotion recognition with unsupervised representation learning on unlabeled speech," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 7390–7394.
[44] S. Ghosh, E. Laksana, L.-P. Morency, and S. Scherer, "Learning representations of affect from speech," 2015, *arXiv:1511.04747*. [Online]. Available: http://arxiv.org/abs/1511.04747
[45] W.-N. Hsu, Y. Zhang, and J. Glass, "Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2017, pp. 16–23.
[46] W.-N. Hsu, Y. Zhang, R. J. Weiss, Y.-A. Chung, Y. Wang, Y. Wu, and J. Glass, "Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 5901–5905.
[47] D. P Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*. [Online]. Available: http://arxiv.org/abs/1312.6114
[48] J. Chang and S. Scherer, "Learning representations of emotional speech with deep convolutional generative adversarial networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 2746–2750.
[49] H. Yu, Z.-H. Tan, Z. Ma, and J. Guo, "Adversarial network bottleneck features for noise robust speaker verification," 2017, *arXiv:1706.03397*. [Online]. Available: http://arxiv.org/abs/1706.03397
[50] S. Sahu, R. Gupta, G. Sivaraman, W. AbdAlmageed, and C. Espy-Wilson, "Adversarial auto-encoders for speech based emotion recognition," 2018, *arXiv:1806.02146*. [Online]. Available: http://arxiv.org/abs/1806.02146
[51] E. Principi, F. Vesperini, S. Squartini, and F. Piazza, "Acoustic novelty detection with adversarial autoencoders," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 3324–3330.
[52] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. C. Courville, and Y. Bengio, "Char2wav: End-to-end speech synthesis," in *Proc. ICLR*, 2017, pp. 1–6.

[53] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous, "Tacotron: Towards End-to-End speech synthesis," 2017, *arXiv:1703.10135*. [Online]. Available: http://arxiv.org/abs/1703.10135

[54] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," 2016, *arXiv:1606.00704*. [Online]. Available: http://arxiv.org/abs/1606.00704

[55] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *CoRR*, vol. abs/1605.09782, pp. 1–18, May 2016.

[56] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8110–8119.

[57] D. Griffin and J. Lim, "Signal estimation from modified short-time Fourier transform," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 32, no. 2, pp. 236–243, Apr. 1984.

[58] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs, "The large time-frequency analysis toolbox 2.0," in *Sound, Music, and Motion*, M. Aramaki, O. Derrien, R. Kronland-Martinet, and S. Ystad, Eds. Cham, Switzerland: Springer, 2014, pp. 419–442.

[59] A. Spurr, E. Aksan, and O. Hilliges, "Guiding InfoGAN with semi-supervision," in *Machine Learning and Knowledge Discovery in Databases*, M. Ceci, J. Hollmén, L. Todorovski, C. Vens, and S. Džeroski, Eds. Cham, Switzerland: Springer, 2017, pp. 119–134.

[60] J. Tobias Springenberg, "Unsupervised and semi-supervised learning with categorical generative adversarial networks," 2015, *arXiv:1511.06390*. [Online]. Available: http://arxiv.org/abs/1511.06390

[61] K. Sricharan, R. Bala, M. Shreve, H. Ding, K. Saketh, and J. Sun, "Semi-supervised conditional GANs," 2017, *arXiv:1708.05789*. [Online]. Available: http://arxiv.org/abs/1708.05789

[62] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.

[63] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *CoRR*, vol. abs/1804.03209, pp. 1–11, Apr. 2018.

[64] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 5206–5210.

[65] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. Int. Conf. Mach. Learn. (PMLR)*, 2017, pp. 1068–1077.

[66] A. Marafioti, N. Holighaus, N. Perraudin, and P. Majdak, "Adversarial generation of time-frequency features with application in audio synthesis," *CoRR*, vol. abs/1902.04072, pp. 4352–4356, May 2019.

[67] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen, "Improved techniques for training GANs," in *Proc. Adv. Neural Inf. Process. Syst.*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2016, pp. 2234–2242.

[68] 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 *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6626–6637.

[69] S. Barratt and R. Sharma, "A note on the inception score," 2018, *arXiv:1801.01973*. [Online]. Available: http://arxiv.org/abs/1801.01973

[70] K. Shmelkov, C. Schmid, and K. Alahari, "How good is my GAN?" in *Computing Vision—ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham, Switzerland: Springer, 2018, pp. 218–234.

[71] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[72] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.

[73] T. van Erven and P. Harremos, "Rényi divergence and Kullback-Leibler divergence," *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 3797–3820, Jul. 2014.

[74] D. C. Dowson and B. V. Landau, "The Fréchet distance between multivariate normal distributions," *J. Multivariate Anal.*, vol. 12, no. 3, pp. 450–455, Sep. 1982.

[75] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, pp. 1–15, Dec. 2015.

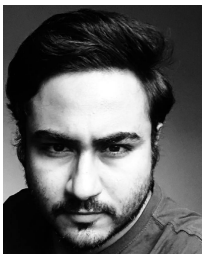
[76] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu, "Mixed precision training," 2017, *arXiv:1710.03740*. [Online]. Available: http://arxiv.org/abs/1710.03740

[77] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," 2018, *arXiv:1809.11096*. [Online]. Available: http://arxiv.org/abs/1809.11096

[78] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[79] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," 2019, *arXiv:1904.08779*. [Online]. Available: http://arxiv.org/abs/1904.08779

[80] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," 2017, *arXiv:1710.10196*. [Online]. Available: http://arxiv.org/abs/1710.10196



KAZI NAZMUL HAQUE received the master's degree in information technology from Jahangirnagar University, Bangladesh. He is currently pursuing the Ph.D. degree with the University of Southern Queensland, Australia. He has been working professionally in the field of machine learning for more than five years. His research interest includes building machine learning models to solve diverse real-world problems. The current focus of his research work is unsupervised representation learning for audio data.

RAJIB RANA (Member, IEEE) received the B.Sc. degree in computer science and engineering from Khulna University, Bangladesh, with the Prime Minister and President's Gold Medal for outstanding achievements, and the Ph.D. degree in computer science and engineering from the University of New South Wales, Sydney, Australia, in 2011. He received his Postdoctoral Training with the Autonomous Systems Laboratory, CSIRO, before joining the University of Southern Queensland, as a Faculty Member, in 2015. He is currently an Experimental Computer Scientist, an Advance Queensland Research Fellow, and a Senior Lecturer with the University of Southern Queensland. He is also the Director of the IoT Health Research Program with the University of Southern Queensland. His research work aims to capitalize on advancements in technology along with sophisticated information and data processing to better understand disease progression in chronic health conditions and develop predictive algorithms for chronic diseases, such as mental illness and cancer. His current research interest includes unsupervised representation learning. He was a recipient of the Prestigious Young Tall Poppy QLD Award, in 2018, as one of the Queensland's most outstanding scientists for achievements in the area of scientific research and communication.

BJÖRN W. SCHULLER, JR. (Fellow, IEEE) received the diploma degree, the Ph.D. degree in automatic speech and emotion recognition, and the habilitation and an Adjunct Teaching Professorship in signal processing and machine intelligence from Technische Universität München (TUM), Munich, Germany, in 1999, 2006, and 2012, respectively, all in electrical engineering and information technology. He is currently a Professor of Artificial Intelligence with the Department of Computing, Imperial College London, U.K., where he heads the Group on Language, Audio and Music (GLAM), a Full Professor and the Head of the Chair of Embedded Intelligence for Health Care and Wellbeing with the University of Augsburg, Germany, and a Founding CEO/CSO of audEERING. He was previously a Full Professor and the Head of the Chair of Complex and Intelligent Systems with the University of Passau, Germany. He has (co-)authored five books and more than 900 publications in peer-reviewed books, journals, and conference proceedings leading to more than overall 30 000 citations (H-index=82). He was an Elected Member of the IEEE Speech and Language Processing Technical Committee. He is a Golden Core Member of the IEEE Computer Society, a Fellow of the ISCA, a Senior Member of the ACM, and the President-Emeritus of the Association for the Advancement of Affective Computing (AAAC). He was the General Chair of ACII 2019, a Co-Program Chair of Interspeech, in 2019, and ICMI, in 2019, a repeated Area Chair of ICASSP, next to a multitude of further Associate and a Guest Editor roles and functions in Technical and Organisational Committees. He is the Field Chief Editor of the *Frontiers in Digital Health* and a former Editor in Chief of the IEEE TRANSACTIONS ON AFFECTIVE COMPUTING.