CasGAN: Conditional Audio Synthesis Generative Adversarial Network

Liu, Ting-Wei
B03901170@ntu.edu.tw

Department of Electrical Engineering, National Taiwan University

ABSTRACT

With the wide success of Generative Adversarial Networks (GANs), promising results on the problem of synthesizing realistic images have proven the capability of GAN models, however little progress has been made in the domain of audio synthesis. In this work, I leverage the success of SpecGAN [1], a frequency domain GAN model that generates audio using the naïve solution that operates waveforms as image-like spectrograms. This work further explores in the domain of audio synthesis based on the work of SpecGAN [1], I propose the new CasGAN model, a conditional audio synthesis generative adversarial network that generates raw audio from the condition of a given word. CasGAN uses a framework similar to DCGAN [2], a two dimensional deep convolutional model that generates spectrograms to approximate target waveforms, where I further modify the architecture into a text-conditional DCGAN model inspired by [3]. I believe that this is the first attempt on applying GANs to conditional raw audio generation in an unsupervised setting. To demonstrated that CasGAN can produce appropriate matching audios from specific words, I frame the task to be a text-to-speech (TTS) [4] problem where the CasGAN model learns to generate intelligible words from a small vocabulary set of human speech. In this work, I aim to use CasGAN to generate realistic test-to-speech audio, where machine learns to generate speech audios that are compatible and indistinguishable to human.

Index Terms **—** *GAN*, SpecGAN, CasGAN, DCGAN, Audio Synthesis, Spectrograms, Test-to-Speech.

1. INTRODUCTION

Synthesizing human-like speech is a well chased objective in the study of text-to-speech [4] domain, as machine audio applications grown popular in practice. In previous works, studies have shown end-to-end learning based approaches to eclipsed the performance of parametric approaches, however such methods depend on training with large quantities of transcribed recordings, but do not take advantage of the additional un-transcribed audio that is often available.

Generative Adversarial Networks (GANs) have shown excellent results on generating high-dimensional signals, where image generation with GAN easily results in high fidelity images, yet little work has shown GAN's capability in synthesizing audio in an unsupervised setting. In the work of SpecGAN [1], the SpecGAN model has shown promising performance on synthesizing high quality human-like speech using the DCGAN [2] method. A frequency domain approach is applied to generate appropriate spectrogram representation that allows for approximate audio inversion. This method uses a specially designed spectrogram representation that is both well-suited to GANs generative process for image and can be approximately inverted, unlike other un-invertible spectrogram approaches done in previous works. However, the work of SpecGAN [1] only shows the model's capability of synthesizing random speeches from Gaussian noise, the problem of using GAN on text-to-speech tasks still remains untouched, and I aim to solve this problem in my work.

To tackle the task of conditional text-to-speech generation, the proposed CasGAN model utilizes the same DCGAN [2] architecture as in SpecGAN [3], where a two dimensional deep convolutional model learns to generate spectrograms as in image generation, furthermore combining with the text condition constraint proposed by [3], where word vectors representing texts are added to the GAN setting. In CasGAN, spectrograms representations are now generated from a text condition, and waveforms are then approximately inversed from the generated spectrogram.

Although another audio generative model WaveGAN has also been proposed along with SpecGAN in the same paper [1], where a time domain approach is utilized to generate raw audio directly, running my own experiments showed that the SpecGAN model is more stable in training and is capable of producing compatible speech quality to the WaveGAN, hence I discard the WaveGAN architecture and only reference the SpecGAN architecture in this work. To evaluate the proposed CasGAN model, I use the standard task *Speech Commands Zero Through Nine* (SC09) proposed by [1], where the machine has to learn to generate speech of digits "zero" through "nine", but different in this setting as these speeches are now generated conditionally. In other words, given an arbitrary input text, the model has to learn to map the text into corresponding waveforms that matches the input text. This work is focused on framing the CasGAN model to generated waveforms that are indistinguishable from human

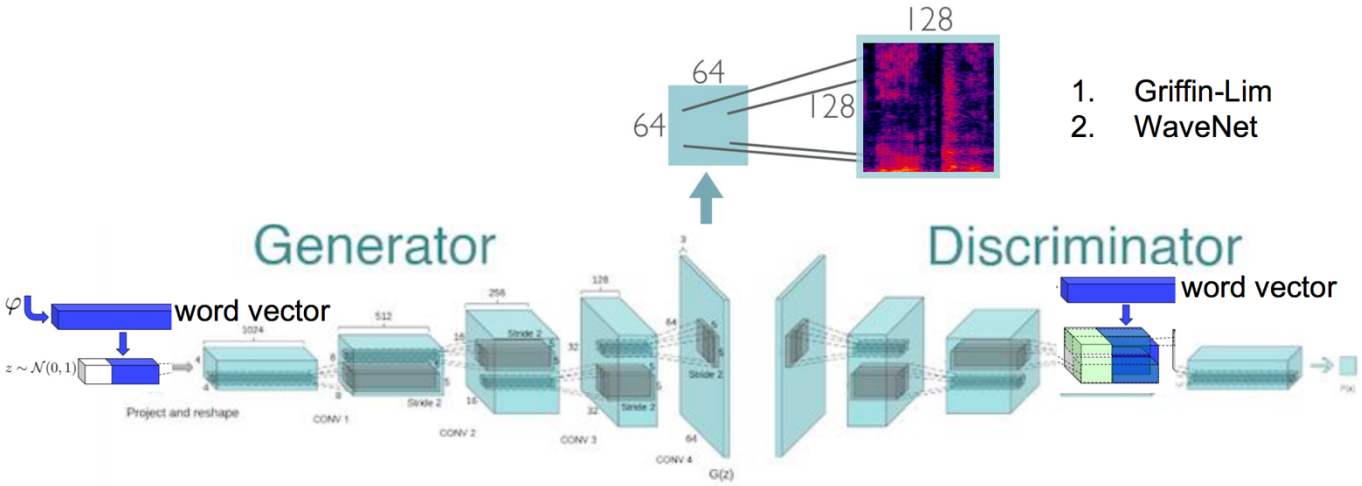

Figure 1. CasGAN text conditional convolutional architecture

speech, translating audio features from text to waveform, and we use the waveforms generated from the unconditioned SpecGAN [1] as speech quality baseline.

2. FORMULATING THE TTS PROBLEM TO THE CASGAN ARCHITECTURE

In this section, we formulate the text-to-speech [4] problem into the generative adversarial network setting, the model learns mapping from a low-dimensional vector to point into the space of human speech data X, where the generator network is denoted as $G: \mathbb{R}^Z \times \mathbb{R}^W \rightarrow \mathbb{R}^S$, and the discriminator network is denoted as $D: \mathbb{R}^S \times \mathbb{R}^W \rightarrow \{0, 1\}$. The notation W is the dimension of the word embedding, and S is the dimension of the spectrogram, and Z is the dimension of the noise input to G.

2.1 The Generator Setting

The generator is trained to generate spectrograms that can fool the discriminator into thinking its output is real, the generated spectrogram is mapped from a low-dimensional concatenated word-noise vector. A Gaussian noise prior $z \in \mathbb{R}^Z \sim \mathcal{G}(-1, 1)$ is first sampled, and we encode the target word from text into a word vector $\varphi \in \mathbb{R}^W$ using the GLOVE [10] word embedding. The word vector φ describing the target word is concatenated to the noise vector z. The concatenated vector proceeds as in a normal deconvolution network: feed-forward the concatenated vector through the generator G, and a synthetic spectrogram $\hat{x}$ is generated via $\hat{x} \leftarrow G(z, \varphi)$, where the generated spectrogram $\hat{x}$ is conditioned on the query word and a noise sample.

2.2 The Discriminator Setting

The discriminator learns to determine if the given sample is real or fake, however the CasGAN model uses the WGAN [5][6] training algorithm that minimizes the Wasserstein-1 distance between real and fake distributions. The following value function is used as the training loss for the discriminator:

$$L_{WGAN} = -\{ \mathrm{E}_{x \sim P_x}[D(x, \varphi)] - \mathrm{E}_{z \sim P_z}[D(\hat{x}, \varphi)] \}$$

where φ is the corresponding targeted word vector, and x is the real data sampled from the real distribution P_x, and z is the noise vector sample from the Gaussian distribution P_z, and $\hat{x}$ is the synthetic spectrogram generated from the generator $\hat{x} \leftarrow G(z, \varphi)$. With this formulation, D is not trained to identify examples as real or fake as $D: \mathbb{R}^S \times \mathbb{R}^W \rightarrow \{0, 1\}$, but instead is trained as a function that assists in computing the Wasserstein distance between real and fake distributions: $D: X \rightarrow \mathbb{R}$. In the WGAN [5][6] formulation, weight clipping is enforced on the discriminator model to ensure that D is 1-Lipschitz, however the CasModel is trained with the alternative WGAN-GP [7], where weight clipping is replaced with gradient penalty to enforce the constraint. Multi experiments have already shown that the WGAN-GP [7] strategy can successfully train a variety of different model configurations where other GAN losses may fail.

2.3 The Model Architecture

The CasGAN architecture proposed is illustrated in Figure 1, which is based on the DCGAN [2] model, but with the difference of an additional layer and two word vectors insertion to both the generator and discriminator. The CasGAN model projects the original 64 by 64 output layer to another 128 by 128 layer, resulting in a 128x128 spectrogram which yields 16384 samples. Once the spectrogram is inverted into audio, each spectrogram is slightly more than one second of audio at 16kHz. The generator takes the word vector as a generation condition, while the discriminator also takes the word vector to check for a match with text and voice. Insertion of the word vector in the discriminator setting makes D match aware, and is crucial in training the model to generate voice-text matching spectrograms.

3. TRAINING DATA

The CasGAN model is trained on the *Speech Commands Zero Through Nine* (SC09) dataset proposed by [1], where the set contains speech of digits "zero" through "nine" with text labels. This dataset is a subset selected from *Speech Commands Dataset* [8]. This dataset originally consists of many speakers recording individual words in uncontrolled recording conditions. These ten words "zero" through "nine" in SC09 represent a variety of phonemes, and each is consisted of multiple syllables. Each recording is one second in length, matching the CasGAN generation capacity of 128x128 spectrogram (16384 samples, 1 sec audio in 16kHz). There are a total of 1850 utterances for each word in the training set, resulting in 5.3 hours of speech.

3.1 Dataset Inconsistency

Since this data was originally recorded in an uncontrolled environment, background noises were highly involved, along with the un-alignment of words in time. Some audio waveforms are presented in Figure 2, showing 5 audio samples representing the same speech "eight", but with entirely different waveforms. The wide variety of alignments, speakers and recording conditions make this a challenging dataset to generate from. In some self-conducted experiments, I find that this inconsistent in training data makes the generated audio noisy and fake, resulting in audio quality unsuitable for the need of TTS [4] applications.

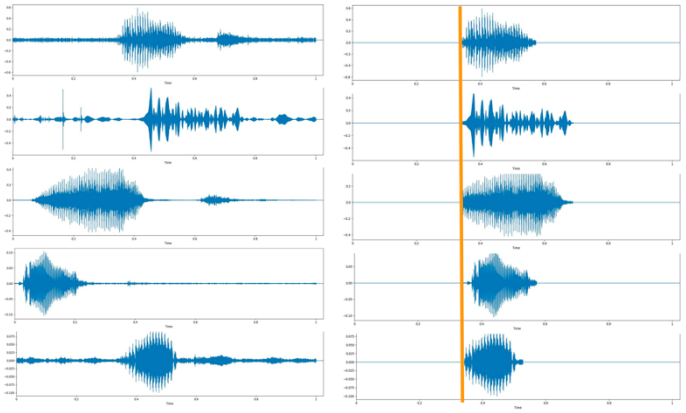

Figure 2. The originally noisy un-aligned waveform (left) and the preprocessed clean waveform (right).

3.2 Dataset Preprocessing

For this reason, it is necessary to process the waveform into an aligned and clean version. I use the LibROSA toolkit [9], a python package for music and audio analysis, to align and clean the training data. In this preprocessing process, I use libROSA [9] to implement a cleanse and alignment algorithm which runs through all training data. The algorithm first detects the useful portion of each waveform that contains the main speech by estimating the noise threshold using iteration. Then the leading and tailing noisy portion of the audio is simply removed, then the separated main speech portion of the audio is aligned to start from the 0.33 second out of the 1 second audio length. Finally, the algorithm pads zeros to the start and end of the remaining 1 second audio length to represent silent. The preprocessing result is shown in Figure 2, where the originally noisy un-aligned waveform (left) is now cleaned and aligned (right). Through this preprocessing process, the useful part of the audio is conserved and aligned, while the noisy part which does not contain speech is removed. The preprocessed dataset is named SC09-Clean and the CasGAN model is trained using this data, allowing the model to learn from a better source of voice.

4. IMPLEMENTATION DETAILS

The CasGAN model is implemented in Tensorflow [11] using Python, the generator consists of 6 layers, including a dense layer that projects the concatenated vector of the Gaussian noise $z \in \mathbb{R}^Z \sim \mathcal{G}(-1, 1)$ and word vector $\varphi \in \mathbb{R}^W$ to a higher dimensional space. After reshaping the projected vector, it is feed-forward through a total of 6 two-dimensional transpose convolutional layers, and finally outputting a 128x128 image-like spectrogram. On the other hand, the discriminator too has 6 layers in total, with 6 two-dimensional convolutional layers that reduce the input spectrogram to a lower dimensional space, and finally passing it through a dense vector to compute the Wasserstein distance. Since the model is trained with the WGAN-GP [7] loss, no batch normalization layer is used. The architecture descripted is summarized below:

Operation	Kernel Size	Output Shape
Input $z \sim \mathcal{G}(-1, 1)$		(n, 100+100)
Dense 1	(100, 256d)	(n, 256d)
Reshape		(n, 4, 4, 16d)
ReLU		(n, 4, 4, 16d)
Trans Conv2D (stride=2)	(5, 5, 16d, 8d)	(n, 8, 8, 8d)
ReLU		(n, 8, 8, 8d)
Trans Conv2D (stride=2)	(5, 5, 8d, 4d)	(n, 16, 16, 4d)
ReLU		(n, 16, 16, 4d)
Trans Conv2D (stride=2)	(5, 5, 4d, 2d)	(n, 32, 32, 2d)
ReLU		(n, 32, 32, d)
Trans Conv2D (stride=2)	(5, 5, 2d, d)	(n, 64, 64, d)
ReLU		(n, 64, 64, d)
Trans Conv2D (stride=2)	(5, 5, d, c)	(n, 128, 128, c)
Tanh		(n, 128, 128, c)

Operation	Kernel Size	Output Shape
Input x *or* $G(z, \varphi)$		(n, 128, 128, c)
Conv2D (stride=4)	(5, 5, c, d)	(n, 64, 64, d)
LReLU ($\alpha = 0.2$)		(n, 64, 64, d)
Conv2D (stride=4)	(5, 5, c, 2d)	(n, 32, 32, 2d)
LReLU ($\alpha = 0.2$)		(n, 32, 32, 2d)
Conv2D (stride=4)	(5, 5, 2d, 4d)	(n, 16, 16, 4d)
LReLU ($\alpha = 0.2$)		(n, 16, 16, 4d)
Conv2D (stride=4)	(5, 5, 4d, 8d)	(n, 8, 8, 8d)
LReLU ($\alpha = 0.2$)		(n, 8, 8, 8d)
Conv2D (stride=4)	(5, 5, 8d, 16d)	(n, 4, 4, 16d)
LReLU ($\alpha = 0.2$)		(n, 4, 4, 16d)
Reshape		(n, 256d)
Dense	(256d, 1)	(n, 1)

The full architectures for the CasGAN generator and discriminator is listed in the two tables above respectively, where n is the batch size, d is the modifiable model size, and c is the number of channels in the examples. All dense and convolutional layers include biases.

5. RESULTS AND CONCLUSION

5.1 Baseline Model

To evaluate the CasGAN model, an audio quality baseline is first set using the SpecGAN architecture, where audio of speech is generated un-conditionally, with the SpecGAN model being set using the same architecture as the CasGAN model except without the conditional word vector. The random generation results from SpecGAN is shown in Figure 3, where I draw waveforms to visualize the generated spectrograms. One can easily observe that the synthetic audio is well aligned and clean without noise, very similar to the training data: SC09-Clean.

5.2 Current Problem and Solution

However, once listened to, it is not hard to notice that these synthesized audio sounds very mechanic and un-human, which is an undesirable result for TTS [4] applications. Therefor currently one branch of my work includes working to generate a better sound quality using the SpecGAN model, since the training data is clean and aligned, I suspect that it is the model architecture that needs to be modified. I assume that it is the spectrogram-audio inversion process that causes the machinery sound, and I aim to solve this by changing the generator model so that it directly outputs a two channel spectrogram which contains both real and imaginary values, resulting in a (2, 128, 128) vector. In this setting phase estimation will no longer be needed in the inversion process, and one can obtain waveform by performing inverse STFT (ISTFT) on the complex spectrogram, as Google did in [12].

5.3 Progress

Another branch of my work is focused on perfecting the training algorithm of CasGAN, as currently I have completed the implementation of all the major parts including model architecture, data loader, and the WGAN-GP [7] training algorithm. However, more work will need to be done on the testing phase of the model. As a final conclusion, this work proposes to use a generative adversarial model – the CasGAN, to generate human speech conditionally given a word in text. To achieve a performable text-to-speech quality, this model needs to acquire the ability of generating human-like audio of speech. This is achieved through a text-conditional complex spectrogram generation from a DCGAN architecture. As for future work, I shall complete the modification of the SpecGAN model, and test its performance with the complex spectrogram, and hopefully setting a better baseline for sound generation quality. Once the improvement on SpecGAN is done, the same complex spectrogram method will be transferred and applied to the CasGAN model.

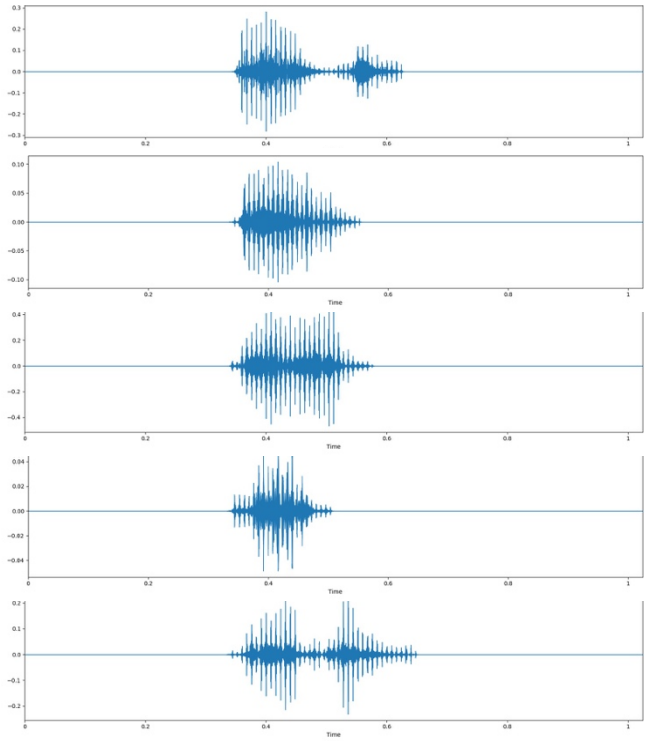

Figure 3. Generated waveforms from the SpecGAN baseline.

6. REFERENCES

[1] Chris Donahue, Julian McAuley, and Miller Puckette, 2017. Synthesizing Audio with Generative Adversarial Networks. In arXiv:1802.04208v1.

[2] Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[3] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee, 2016. Generative Adversarial Text to Image Synthesis. In arXiv:1605.05396v2.

[4] Shen, Jonathan, Pang, Ruoming, Weiss, Ron J, Schuster, Mike, Jaitly, Navdeep, Yang, Zongheng, Chen, Zhifeng, Zhang, Yu, Wang, Yuxuan, Skerry-Ryan, RJ, et al. Natural TTS synthesis by conditioning WaveNet on Mel spectrogram predictions. In *ICASSP, 2018*.

[5] Salimans, Tim, Goodfellow, Ian, Zaremba, Wojciech, Cheung, Vicki, Radford, Alec, and Chen, Xi. Improved techniques for training GANs. In *NIPS*, 2016.

[6] Arjovsky, Martin, Chintala, Soumith, and Bottou, Le´on. Wasserstein GAN. In *ICML*, 2017.

[7] Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Du- moulin, Vincent, and Courville, Aaron. Improved training of Wasserstein GANs. In *NIPS*, 2017.

[8] Warden, Pete. Speech commands dataset. https://research.googleblog.com/2017/08/launching-speech-commands-dataset.html, 2017. Accessed: 2018-07-11.

[9] Librosa development team. LibROSA. https://librosa.github.io/librosa/, 2013. Accessed: 2018-07-11.

[10] Jeffrey Pennington, Richard Socher, Christopher D. Manning. GloVe: Global Vectors for Word Representation. https://nlp.stanford.edu/projects/glove/, 2014. Accessed: 2018-07-11.

[11] Abadi, Mart´ın, Agarwal, Ashish, Barham, Paul, Brevdo, Eugene, Chen, Zhifeng, Citro, Craig, Cor- rado, Gregory S., Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Goodfellow, Ian J., Harp, Andrew, Irving, Geoffrey, Isard, Michael, Jia, Yangqing, Józefowicz, Rafal, Kaiser, Lukasz, Kudlur, Manjunath, Levenberg, Josh, Mane´, Dan, Monga, Rajat, Moore, Sherry, Murray, Derek Gordon, Olah, Chris, Schuster, Mike, Shlens, Jonathon, Steiner, Benoit, Sutskever, Ilya, Talwar, Kunal, Tucker, Paul A., Vanhoucke, Vincent, Vasudevan, Vijay, Vie´gas, Fernanda B., Vinyals, Oriol, Warden, Pete, Wattenberg, Martin, Wicke, Martin, Yu, Yuan, and Zheng, Xiao- qiang, 2016. "Tensorflow: Large- scale machine learning on heterogeneous distributed systems," In arXiv: 1603.04467v2.

[12] Ariel Ephrat, Inbar Mosseri, Oran Lang, Tali Dekel, Kevin Wilson, Avinatan Hassidim, William T. Freeman, Michael Rubinstein, 2019. Looking to Listen at the Cocktail Party: A Speaker-Independent Audio-Visual Model for Speech Separation. In arXiv:1804.03619v1.