Interesting properties of GAN samples

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper we investigate numerical properties of samples produced with adversarial methods, specially Generative Adversarial Networks. We analyzep pixel value statistics of real and fake data and compute distances based on the marginal distribution of perceptually significant features. We provide results on MNIST, music and speech data and show that GAN generated samples have interesting signatures that can be used to identify the source of the data and detect adversarial attacks.

1 Introduction

Since the groundbreaking Generative Adversarial Networks paper [5] in 2014, GAN related publications use a grid of image samples to accompany theoretical and empirical results. GAN research focuses is expanding to other domains including language models [7] and music [15], requiring new methods of sample inspection.

Unlike variational auto encoders and other models [5], most of the evaluation of the output of Generators trained with the GAN framework is qualitative: authors normally list higher sample quality as one of the advantages of their method over other methods. Interestingly, little is mentioned about the numerical properties of GAN samples and how these properties compare to real samples.

In the context of verifiable Artificial Intelligence[14], it is hard to systematically verify the Generator because verification depends on the existence of perceptually meaningful features. For example, consider the generation of images of mamals: although it is possible to compare color histograms of fake[1] and real samples, we do not yet have robust algorithms able to verify if an image follows specifications derived from anatomy.

This paper is related to this effort and focuses on understanding the numerical properties of GAN samples. We investigate how the Generator approximate modes in the real distribution and verify if the generated samples violate specifications derived from the real distribution. We offer the following contributions in this paper:

- We show that GAN samples have universal signatures.
- We show how GAN samples approximate modes of the real distribution.
- We show significant differences between the marginal distribution of features.
- We show GAN samples that violate specifications in the real data.

2 Related work

Despite its youth, several publications ([2], [13], [16], [12]) have investigated the use of the GAN framework for generation of samples and unsupervised feature learning. Following the procedure

[1] Generated samples

Submitted to 31st Conference on Neural Information Processing Systems (NIPS 2017). Do not distribute.

described in [1] and used in [5], earlier GAN papers evaluate the quality of the Generator by fitting a Gaussian Parzen window[2] to the GAN samples and reporting the log-likelihood of the test set under this distribution. It is known that this method has some drawbacks, including its high variance and bad performance in high dimensional spaces [5].

Unlike other optimization problems where analysis of the empirical risk is a strong indicator of progress, in GANs decrease in loss is not always correlated with increase in image quality [3], and thus authors still relly on visual inspection of generated images. Based on visual inspection, authors confirm that they have not observed mode collapse or that their framework is robust to mode collapse if some criteria is met ([3], [7], [9], [12]). In practice, github issues where practicioners report mode collapse or not enough variety abound.

In their brilliant publications, [9], [3] and [7] propose alternative objective functions and algorithms that circunvemt problems that are common when using the original GAN objective. The problems addressed include instability of learning, mode collapse cand meaningful learning curves [13].

These alternatives do not eliminate the need or excitement[3] of visually inspecting GAN samples during training, nor do they provide numerical information about the generated samples. In the following sections, we will reveal some interesting properties of GAN samples. In addition to comparing the marginal distribution of features from the real and fake data, we approach these distributions as specifications that can be used to validate the output of GAN Samples. We start by enumerating the hypotheses evaluated in this paper.

In the next section we describe the hypotheses evaluated in this paper.

3 Hypotheses

Hypothesis 1 (H1): *Generative models can approximate the distribution of real data and hallucinate fake data that resembles real data and has some variety.*

Although this hypothesis is trivial for experiments that have already been conducted, it is the first condition for our experiments with polyphnic music and speech data. To our knowledge there are no publications where GANs are successful in hallucinating polyphonic music and speech data. During out experiments we prove that these hypotheses hold.

Hypothesis 2 (H2): *The real data has useful properties that can be extracted computationally.*

By useful we refer to properties that are closely related to the real data itself. For example, computing the distribution MNIST pixel values might be not useful for assessing drawing quality. However, it might be useful to evaluate if a random MNIST samples is real or fake data.

Hypothesis 3 (H3): *The fake data has properties that are hardly noticed with visual inspection of samples.*

Visual inspection of generated samples has become the norm for the evaluation of samples generated using the GAN framework. We investigate if there are properties common to all GAN samples or properties that significantly differ between the real data and the fake data. This hypothesis supports the next hypothesis related to adversarial attacks.

Hypothesis 4 (H4): *The difference in properties can be used to identify the source (real or fake)*

The development of generative models foreshadow the iminent rise of adversarial attacks. We investigate if these differences can be used to detect the source of the data (real, GAN or adversarial attack).

We call the reader's attention that approximating the distribution over features computed on the real data does not guarantee that the real data is being approximated. Formally speaking: consider $X \sim Z$, i.e. X distributed as Z, and $f(X) \sim W$, where $f : X \mapsto Y$. If $A \sim B$ and B approximates Z, then $f(A) \sim D$ must also approximate W. However, a distribution that approximates W is not guaranteed to approximate Z.

[2] Kernel Density Estimation

[3] Despite of authors promising on twitter to never touch GANs again.

4 Method

In this section we describe our analysis method in detail, including briefly describing the datasets and features computed, as well as generative models.

4.1 Datasets

In our experiments we use the MNIST dataset, a MIDI dataset of 389 Bach Chorales downloaded from the web and a subsample of the NIST 2004 telefone conversational speech dataset with 100 speakers, multiple languages and on average of 5 minutes per speaker.

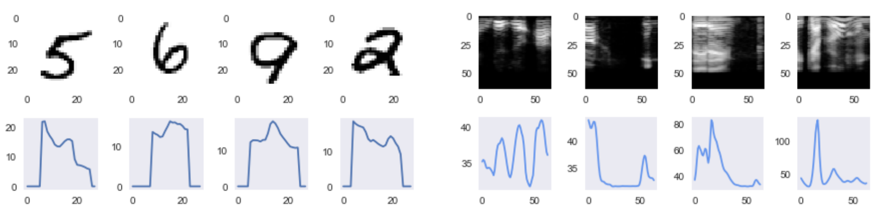
4.2 Property extraction

The properties extracted from the datasets used on this paper can be perceptually meaningful or not. We claim that both properties can be used to numerically identify the source of the sample. In the context of this paper, samples are images of fixed size.

4.2.1 Spectral Moments

The spectral centroid [11] is a feature commonly used in the audio domain, where it represents the barycenter of the spectrum. This feature can be applied to other domains and we invite the reader to visualize Figure 1 for examples on MNIST and Mel-Spectrograms [11]. For each column in an image, we transform the pixel values into row probabilities by normalizing them by the column sum, after which we take the expected row value, thus obtaining the spectral centroid.

Figure 1a shows the spectral centroid computed on sample of MNIST training data.

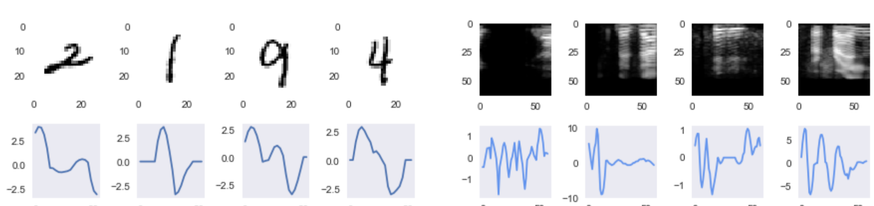


(a) MNIST samples and centroids (b) Mel-Spectrograms and centroids

Figure 1: Spectral centroids on digits and Mel-Spectrograms

4.2.2 Spectral Slope

The spectral slope is computed by applying linear regression using an overlapping sliding window of size 7. For each window, we regress the spectral centroids on the column number *mod* the window size. Figure 2 shows these features computed on MNIST and Mel-Spectrograms.



(a) MNIST samples and slopes (b) Mel-Spectrograms and slopes

Figure 2: Spectral slopes on digits and Mel-Spectrograms

4.3 Generative Models

We investigate samples produced with the DCGAN architecture using the Least-Squares GAN (LSGAN) [9] and the improved Wasserstein GAN (IWGAN) [7]. We also compare adversarial MNIST samples produced with the fast gradient sign method (FGSM) [6].

5 Experiments

5.1 MNIST

We compare the distribution of features computed over the MNIST training set to other datasets, including the MNIST test set, samples generated with GANs and adversarial samples computed using the FGSM. The training data is scaled to $[0, 1]$ and the random baseline is sampled from a Bernouli distribution with probability equal to the normalized mean value of pixel intentities in the MNIST training data, 0.13. Each GAN model is trained until the loss plateaus and the generated samples look similar to the real samples. Every dataset has 10k samples.

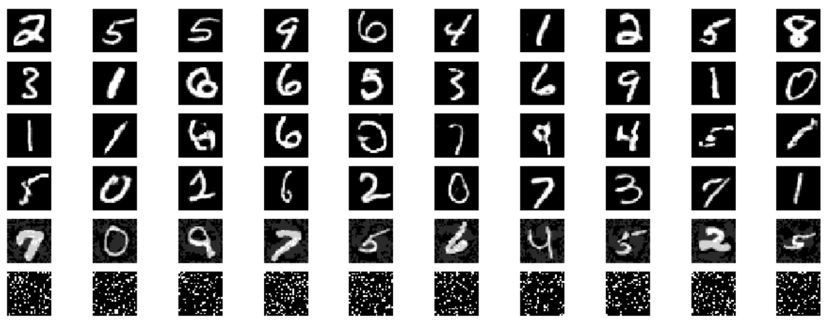


Figure 3: Samples drawn from MNIST train, test, LSGAN, IWGAN, FSGM and bernoulli respectively.

Visual inspection of these 10 generated samples in Figure 3 show that IWGAN seems to produce better samples than LSGAN. We use the MNIST training set as a reference and compare the distribution of pixel intensities. Table 1 reveals that although samples generated with LSGAN and IWGAN look similar to the training set, they are considerably different given the Kolgomorov-Smirnov (KS) Two Sample Test and the Jensen-Shannon Divergence (JSD), specially if compared to the same statistics on the MNIST test data.

	KS Two Sample Test		JSD
	Statistic	P-Value	
mnist_train	0.0	1.0	0.0
mnist_test	0.003177	0.0	0.000029
mnist_lsgan	0.808119	0.0	0.013517
mnist_iwgan	0.701573	0.0	0.014662
mnist_adversarial	0.419338	0.0	0.581769
mnist_bernoulli	0.130855	0.0	0.0785009

Table 1: Statistical comparisson over the distribution of pixel values for different samples using MNIST training set as reference.

This numerical phaenomena can be understood by investigaten the empirical CDFs in Figure 4. The pixel values of the samples generated with the GAN framework are mainly bimodal and asymptotically approaching the modes of the distribution, 0 and 1. Such behavior will be present in any gradient descent method using an asymptotically converging non-linearity, such as sigmoid and tanh, immediately preceding the output of the generating function.

In addition, Figure 5 shows that the GAN generated samples smoothly approximate the modes of the distribution. This smooth approximation is considerably different from the training and test sets.

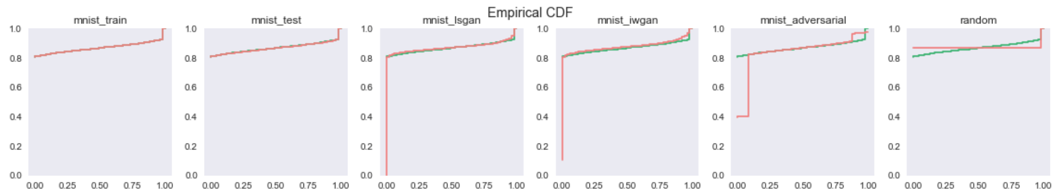


Figure 4: Pixel empirical CDF of training data as reference in green and other datasets

Although these properties are not perceptually meaningful, they can be used to identify the source of the data, hence confirming hypotheses 2, 3 and 4.

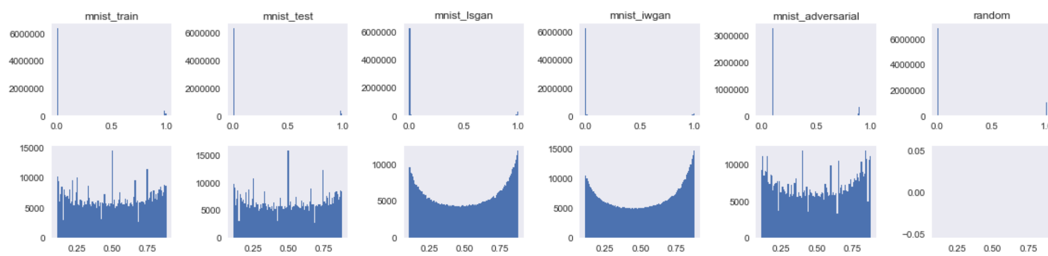


Figure 5: Histogram of pixel intensities for each dataset. First row shows histogram within the (0, 1) interval and 100 bins. Second row shows histogram between the (0.11, and 0.88) interval and 100 bins.

5.2 Bach Chorales

We investigate the properties of Bach Chorales generated with the GAN framework and verify if they satisfy musical specifications. Bach Chorales are polyphonic pieces of music, normally written for 4 or 5 voices, that follow a set of specifications/rules[4]. For example, a global specification could assert that only a set of durations can be used; a local specification could assert that only certain transitions between states (notes) are valid depending on the current harmony.

For this experiment, we convert the dataset of Bach chorales to piano rolls. The piano roll is a representation in which the rows represent note numbers, the columns represent time steps and the cell values represent note intensities. We compare the distribution of features computed over the training set, test set, gan generated samples and a random baseline sampled from a Bernoulli distribution with probability equal to the normalized mean value of intensities in the training data. After scaling, the intensities in the training and test data are strictly bimodal and equal to 0 or 1. Figure 6 below shows training, test, IWGAN and Bernoulli samples, thus confirming hypothesis 1. Each dataset has roughly 1000 image patches.

Figure 7 shows a behavior that is similar to our previous MNIST experiments: the IWGAN asymptoptically approximates the modes of the distribution of the distribution of intensity values. In the interest of space, we refer the reader to the appendix for statistics and other relevant information.

Following, we investigate if the generated samples violate the specifications of Bach chorales. For doing so, we first convert the all data to boolean by thresholding at 0.5 such that values above the threshold are set to 1. We use these piano rolls to compute boolean Chroma [11] feature and to compute an Chroma empirical transition matrix, where the positive entries represent existing and valid transitions. The transition matrix built on the training data is taken as the reference specification, i.e. anything that is not included is a violation of the specification. Table 2 shows the number of violations given each dataset. Compared to the test set that has only 429 violations, the IWGAN samples have over 5000 violations, 10 times more than the test set! We use these facts to confirm hypotheses 2, 3 and 4.

[4]The specifications define the characteristics of the style.

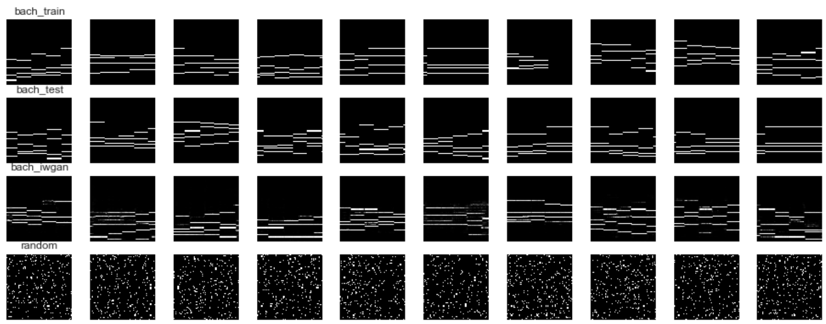


Figure 6: Samples drawn from Bach Chorales train, test, IWGAN, and Bernoulli respectively.

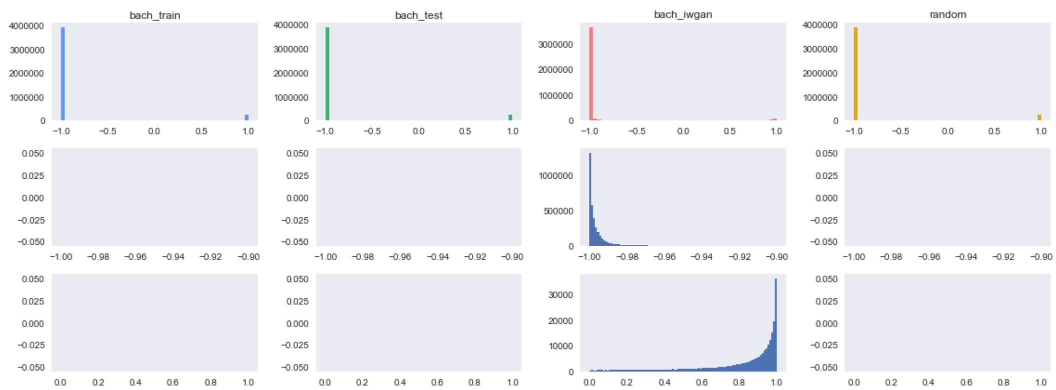
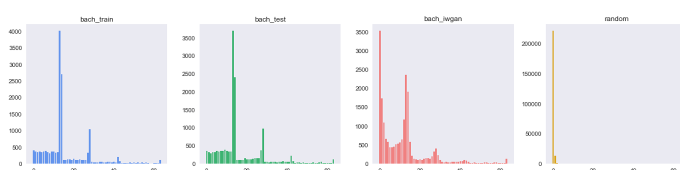


Figure 7

	bach_train	bach_test	bach_iwgan	bach_bernoulli
Number of Violations	0	429	5029	58284

Table 2: Number of specification violation with respect to training data as reference.

In addition to experiments with Chroma features, we computed the distribution of note durations on the boolean piano roll described above. Figure 8a shows the distribution of note durations within each dataset. The train and test data are approximatelly bimodal and, again, the improved WGAN smoothly approximates the dominating modes of the distribution. Table 8b provides a numerical comparisson between datasets.



(a) Duration distribution

	KS Two Sample Test		JSD
	Statistic	P-Value	
train	0.0	1.0	0.0
test	0.09375	0.929	0.002
iwgan	0.21875	0.080	0.084
bernoulli	0.93750	0.0	0.604

(b) Test statistics

5.3 Speech

Within the speech domain, we investigate dynamic compressed Mel-Spectrogram samples produced with GANs trained on a subset of the NIST 2004 dataset, with 100 speakers. We divide the NIST 2004 dataset into training and test set, generate samples with the GAN framework and use a random baseline sampled from a Exponential distribution with parameters chosen using heuristics. The generated samples can be seen in Figure 9, thus confirming hypothesis 1. We obtain the Mel-Spectrogram by projecting a spectrogram onto a mel scale, which we do with the python library librosa [10]. More specifically, we project the spectrogram onto 64 mel bands, with window size equal to 1024 samples and hop size equal to 160 samples, i.e. frames of 100ms long. Dynamic range compression is computed as described in [8], with $log(1 + C * M)$, where C is the compression constant scalar set to 1000 and M is the matrix representing the Mel-Spectrogram. Each dataset has approximately 1000 image pataches and the GAN models are trained using DCGAN with the improved Wasserstein GAN algorithm.

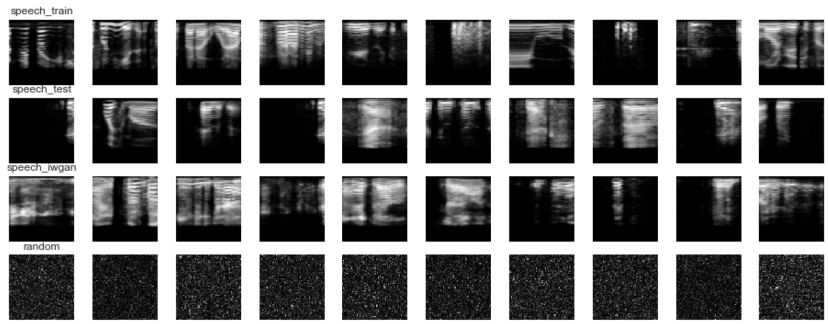
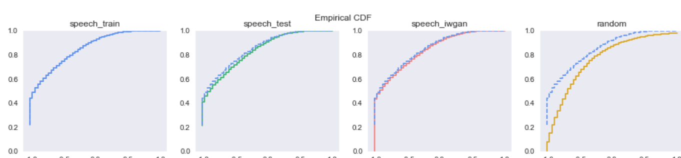


Figure 9: Samples drawn from Mel-Spectrogram Speech train, test, IWGAN, and exponential respectively.

In Figure 10a we show the empirical CDFs of intensity values. Unlike our previous experiments where intensity (Bach Chorales) or pixel value (MNIST) was linear, in this experiment intensities are compressed using the log function. This considerably reduces the distance between the empirical CDFs of the training data and GAN samples, specially around the saturating points of the non-linearity, -1 and 1 in this case. In Table 10b we show numerical analysis of the differences and confirm hypotheses 2 and 3.



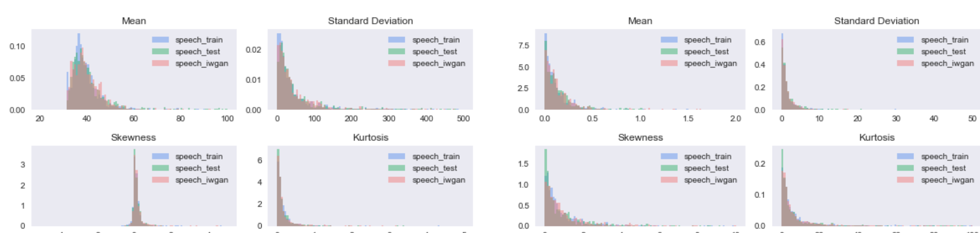
(a) Intensity empirical CDF of training data in blue and other datasets.

	KS Two Sample Test		JSD
	Statistic	P-Value	
train	0.0	1.0	0.0
test	0.03685	0.0	0.00080
iwgan	0.22149	0.0	0.00056
bernoulli	0.36205	0.0	0.11423

(b) Test statistics

Figure 10: Empirical CDF and statistical tests of speech intensity

Figure 11 shows the distribution of moments computed on spectral centroids and slope. The distributions from different sources considerably overlap, indicating that the generator has efficiently approximated the real distribution of these features.

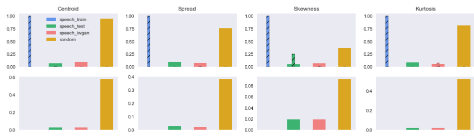


(a) Spectral Centroid Moments

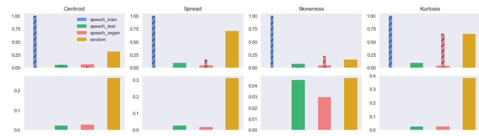
(b) Spectral Slope Moments

Figure 11: Moments of spectral centroid (left) and slope(right)

Figure 12 shows statistics used to compare the reference (training data) and other datasets. The difference between KS-Statistics and JSD of the test data and generated samples are negligible. Interestingly, the p-values of the spectral slope of the improved WGAN are considerably higher than the test data. For these reasons and although Table 10b shows a significant difference between the KS-Statistic of test data and generated data with respect to the training data, we refrain from confirming hypothesis 4). An adversary can easily manipulate the generated data to considerably decrease this difference and still keeping the high similarity in features harder to simulate such as moments of spectral centroid or slope.



(a) Spectral Centroid Statistics



(b) Spectral Slope Statistics

Figure 12: Statistics of spectral centroid (left) and slope(right)

6 Conclusions

In this paper we investigated numerical properties of samples produced with adversarial methods, specially Generative Adversarial Networks. We showed that GAN samples have universal signatures that are dependent on the choice of non-linearity on the last layer of the generator. In addition, we showed that adversarial examples produced with the FSGM have properties that can be used to identify an adversarial attack. Following, we showed that GAN samples smoothly approximate the dominating modes of the distribution and that this information can be used to identify the source of the data. Finally, we showed that samples generated with GANs do not provide guarantees on satisfaction of simple specifications. With this we hope to call attention to our community to the necessity of developing a theory of verifiable artificial intelligence.

Acknowledgments

References

[1] Quickly generating representative samples from an rbm-derived process.

[2] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *NIPS 2016 Workshop on Adversarial Training. In review for ICLR*, volume 2016, 2017.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[4] David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[7] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017.

[8] Yanick Lukic, Carlo Vogt, Oliver Dürr, and Thilo Stadelmann. Speaker identification and clustering using convolutional neural networks. In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016.

[9] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. *arXiv preprint ArXiv:1611.04076*, 2016.

[10] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, 2015.

[11] Geoffroy Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. 2004.

[12] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[13] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016.

[14] Sanjit A. Seshia and Dorsa Sadigh. Towards verified artificial intelligence. *CoRR*, abs/1606.08514, 2016.

[15] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation using 1d and 2d conditions. *CoRR*, abs/1703.10847, 2017.

[16] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.