Partially Conditioned Generative Adversarial Networks

DOI:
10.48550/arXiv.2007.02845

PARTIALLY CONDITIONED GENERATIVE ADVERSARIAL NETWORKS

Francisco J. Ibarrola [*†] **Nishant Ravikumar** [*†] **Alejandro F. Frangi** [*† ‡]

July 7, 2020

ABSTRACT

Generative models are undoubtedly a hot topic in Artificial Intelligence, among which the most common type is Generative Adversarial Networks (GANs). These architectures let one synthesise artificial datasets by implicitly modelling the underlying probability distribution of a real-world training dataset. With the introduction of Conditional GANs and their variants, these methods were extended to generating samples conditioned on ancillary information available for each sample within the dataset. From a practical standpoint, however, one might desire to generate data conditioned on partial information. That is, only a subset of the ancillary conditioning variables might be of interest when synthesising data. In this work, we argue that standard Conditional GANs are not suitable for such a task and propose a new Adversarial Network architecture and training strategy to deal with the ensuing problems. Experiments illustrating the value of the proposed approach in digit and face image synthesis under partial conditioning information are presented, showing that the proposed method can effectively outperform the standard approach under these circumstances.

keywords: Generative models, partial information, conditional GANs, image synthesis.

1 Introduction

Throughout the past years, much research effort has been put in generative model development ([1], [2]). On top of building a model capable of synthesising realisations from the underlying distribution of a given data set, one might want to generate samples conditioned by certain information. While this problem has been addressed for synthesising data with certain attributes ([3]), some fields, such as virtual patient synthesis for medical testing ([4], [5]) could greatly benefit from a model's ability to generate samples from an arbitrary subset within a large set of conditioning information. Building a model which can generate from partial conditioning information is the problem we shall address in what follows.

Let us consider the problem of generating synthetic data samples conditioned on some ancillary information. Assume we have a data set consisting of pairs $\{x_n, y_n\}_{n=1,\dots,N}$, where $x_n \in \mathbb{R}^M$ is the target data and $y_n \in \mathbb{R}^K$ is a vector containing available information associated to x_n. Let us further assume that the pairs $\{x_n, y_n\}$ are independently sampled realizations from an underlying probability distribution $\pi(X, Y)$. Then the goal of a conditional generative model is, given an arbitrary entry $y \in \mathbb{R}^K$, to generate representative samples from the conditional distribution $\pi(X|Y = y)$.

A recent yet widespread approach to dealing with these problems is the use of Conditional GANs ([3]) where two distinct Neural Networks, a Generator and a Discriminator, are trained in an adversarial fashion. This results in a Generator producing samples from $\pi(X|Y = y)$ from random Gaussian noise, given some conditioning information y.

[*]CISTIB Centre for Computational Imaging & Simulation Technologies in Biomedicine, School of Computing, University of Leeds, UK. (f.ibarrola@leeds.ac.uk)

[†]LICAMM Leeds Institute for Cardiovascular and Metabolic Medicine, School of Medicine, University of Leeds, UK.

[‡]Department of Cardiovascular Sciences, and Department of Electrical Engineering, ESAT/PSI, KU Leuven, Leuven, Belgium; Medical Imaging Research Center, UZ Leuven, Herestraat 49, 3000 Leuven, Belgium.

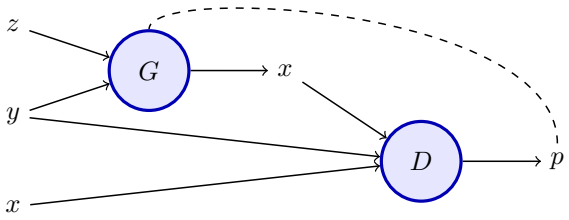

Figure 1: Diagram of the Conditional GAN network architecture. G corresponds to the generator network and D to the discriminator.

Variants of this approach have been successfully used in various applications, such as medical image synthesis ([6]), image deblurring ([7]) and parameter optimisation ([8]), to name a few. Nevertheless, open problems that are yet to be addressed might be encountered when dealing with real world scenarios.

A straightforward data synthesis problem might entail generating samples conditioned only on partial information rather than the complete conditioning vector y. Given an incomplete conditioning vector $\bar{y}$, we want to sample from $\pi(X|Y = \bar{y})$. Furthermore, the components of y kept in $\bar{y}$ might be unknown *a priori* or they might even change depending on the available data and formulation of the problem. For instance, in the field of medical image synthesis, there is usually plenty of conditioning information with which to train a model. From a practical standpoint, however, it is usually desirable for some characteristics (conditioning variables) to be left unspecified when generating populations.

While other authors have tackled the problem of training using databases with noisy or uncertain conditioning variables ([9], [10]), to the best of our knowledge the problem of building a generator that can work under partial conditioning has yet to be addressed.

In this work we show standard Conditional GANs fail in this problem setting, thus highlighting the need to either retrain the model for every set of conditioning variables or to infer the missing inputs by learning their joint latent space, which is often non-trivial. To overcome this issue, we propose an adversarial architecture that generates samples from incomplete conditioning vectors without estimating the missing entries. We then describe a simple learning strategy and illustrate and validate our approach in two classical problems in computer vision - synthesis of handwritten digit (MNIST) and face images (CelebA).

2 Conditional GANs

Let $\{x, y\}$ be a realisation from an unknown distribution $\pi(X, Y)$, and let us consider an additional random variable $Z \sim \mathcal{N}(0, I_{J\times J})$. A Conditional GAN ([3]) consists of a generator function $G : \mathbb{R}^J \times \mathbb{R}^K \to \mathbb{R}^M$ which seeks to generate artificial realisations from $\pi(X|Y = y)$, and a discriminator function $D : \mathbb{R}^M \times \mathbb{R}^K \to [0, 1]$ that estimates the probability of an input $x \in \mathbb{R}^M$ coming from the actual training set rather than the generator, given the corresponding conditioning vector y. Hence, training a Conditional GAN equals to solving a two-player min-max game over a cost function, given by

$$V(G, D, y) \doteq \mathbb{E}_{x\sim\pi_{data}} \log D(x|y) + \mathbb{E}_{z\sim\pi_z} \log[1 - D(G(z|y)|y)].$$

Given the characteristics of the function, D should be chosen as to maximise V (which translates into good discrimination capability), while G should be such as to attempt to minimise V, thus "fooling" the discriminator. This structure is illustrated in Figure 1.

Once this network has been trained, sampling from $\pi(X|Y = y)$ amounts to evaluating $G(\cdot\,|y)$ on a realisation z from the distribution π_z. But if we want to generate samples conditioned on a subset $\bar{y}$ of the conditioning variables, $G(z|\bar{y})$ will generally not produce good results during inference.

2.1 Missing conditioning information

When working with missing entries in the conditioning vector, one can address two issues: the first is how to represent the missing data in the context of the network inputs; the second is how to synthesise data under partial conditioning information.

$y = (0, 0, 0, 1, 0, 0, 0, 0, 0, 0)$

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$

Figure 2: Examples of images generated with a standard Conditional GAN. Top row: complete conditioning information. Bottom row: incomplete conditioning information.

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1)$

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)$

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)$

Figure 3: Output examples of a standard Conditional GAN Generator under complete conditioning information (top row) and missing conditioners (middle and bottom rows).

Let us consider that the information on $\mathbb{Y}$ is categorical, *i. e.* y contains one or more characteristics describing x. Then, given L possible states, we can define $y \in \{0, 1\}^L$ as

$$y_l = \begin{cases} 1 & \text{if } x \text{ is in the } l\text{-th class,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

This kind of representation is called *one-hot-embedding* ([11]). Using this definition, missing information can simply be noted as an element of y being 0 where it should be 1.

Suppose we have trained our Conditional GAN with the aforementioned vector y, the question remains whether the generator would still work with incomplete data. In order to illustrate the ensuing issue, we trained a Conditional GAN using the MNIST dataset [12], using a vector $y \in \{0, 1\}^{10}$ where a 1 in the i-th position indicates the i-th digit. An example of the generator's output is given in the top row images on Figure 2. The bottom row, on the other hand, depicts what happens when the label is missing, which ideally should be a drawn digit, or at least resemble one.

To demonstrate this is a fundamental problem of this approach rather than a consequence of the complete lack of conditioning information given to the network, we have repeated the experiment using $y \in \{0, 1\}^{11}$, using the label $y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1)$ for indicating the digit 9 while preserving all the others.

Figure 3 depicts the results obtained when the generator is given missing inputs, whilst preserving the necessary information for producing the desired output (digit 9). The alluded problem persists.

One might then seek to "fill" the missing entries on $\bar{y}$, obtaining an approximation $\hat{y} \approx y$ and then feed it to the Conditional GAN. While this could work in a toy example such as the one depicted above, when dealing with real data, "filling" y amounts to sampling from the conditional distribution $\pi(Y|Y_l = \bar{y}_l, \forall l \in \mathcal{L})$, where $\mathcal{L}$ is the set of

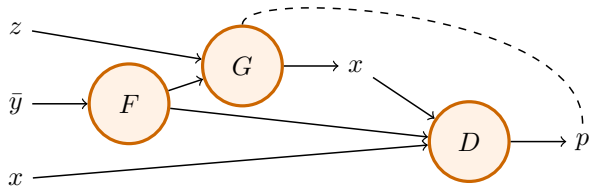

Figure 4: Diagram of PCGAN architecture. G corresponds to the generator, D to the discriminator and F to the feature extraction network.

indexes of available data. This problem can be rather simple when the random variables Y_l are independent, but in more practical and interesting scenarios, there is always some degree of correlation between the conditioning variables, which implies that sampling from the conditional distribution $\pi(Y|Y_l = \bar{y}_l, \forall l \in \mathcal{L})$ becomes a non-trivial problem.

In the next Section we propose an architecture to tackle the missing data problem in a general context that also precludes the need of estimating the aforementioned marginal distribution.

3 Architectures for dealing with missing conditioning data

From what has been observed in Figure 3, one can readily conclude that the network is not actually learning a 1 in the 10-th entry of y and a 1 in the 11-th entry as separate characteristics indicating a digit 9. Hence, the problem in this simple amounts to making the network learn an OR gate over the 10-th and 11-th entries.

The primary characteristic with which we would like to imbue our network is the ability to utilise the available information when generating data, even if incomplete. In the problem illustrated in Figure 3, this would mean learning an OR gate between the redundant information bits. In the context of a real world problem, this would mean the network be able to bypass the missing information and generate samples from whatever data is available.

Formally, given a vector $\bar{y}$ with missing entries, we want our network to be able to sample from $\pi(X|Y_l = \bar{y}_l, \forall l \in \mathcal{L})$. Note that accomplishing this would overcome the problem of estimating $\pi(Y|Y_l = \bar{y}_l, \forall l \in \mathcal{L})$.

To do this, we propose a Partially Conditioned Generative Adversarial Network (PCGAN), formulated as shown in Figure 4. In this diagram, F is a "feature extraction" neural network that extracts the underlying information from $\bar{y}$. This can be accomplished by a shallow multilayer perceptron for categorical information, but if other type of conditioning information is available, other architectures might be better suited (e.g. if the conditioner is an image, we could choose F to be a convolutional network). In our toy example, the goal of F would be to act as an OR gate.

This means that the cost function is now

$$V(G, D; \bar{y}) \doteq \mathbb{E}_{x\sim\pi_{data}} \log D(x|F(\bar{y})) + \mathbb{E}_{z\sim\pi_z} \log[1 - D(G(z|F(\bar{y}))\,|\,F(\bar{y}))].$$

Note that since F is a differentiable neural network, adapting the backpropagation process for optimisation is straightforward.

Once we have an architecture capable of information extraction (building an OR gate), the problem remains on how to embed that function into the architecture. A simple solution is that for the generator to work with missing conditioning entries, it must learn to work with missing entries. Hence, during the training process we shall remove some of the entries on the data points to emulate the expected working condition of the generator when synthesising data. A simple way to accomplish this is to assign a probability $p \in (0, 1)$ of being observed for every entry. The training process is summarized in Algorithm 1.

Note that F is trained along with G as the goal is to tune it for the generation process. The parameter p corresponds to the percentage of conditioning entries to be used for training, and does not necessarily match that to be observed when generating samples in a practical problem. Finally, while the training process is described for using a sample at a time, batch training is recommended.

The next section illustrates through examples how the proposed method works, as well as its performance.

Algorithm 1 PCGAN training

for $\{x_i, y_i\} \in \{X, Y\}$ **do**

Updating D
Let $z \sim \pi_z$ and $\bar{y}_i = y_i \odot \mathbf{b}_p$
$\hat{x}_i = G(z, F(\bar{y}_i))$
$e_{\text{fake}} = D(\hat{x}_i, F(\bar{y}_i))$
$e_{\text{true}} = D(x_i, F(\bar{y}_i))$
Backpropagate $e_{\text{fake}} + e_{\text{true}}$ to update the weights of D.

Updating G **and** F
Let $z \sim \pi_z$ and $\bar{y}_i = y_i \odot \mathbf{b}_p$
$\hat{x}_i = G(z, F(\bar{y}_i))$
$e_{\text{fake}} = D(\hat{x}_i, F(\bar{y}_i))$
Backpropagate e_{fake} to update the weights of G and F.

end for

$\odot$ denotes the Hadamard (pointwise) product.
$\mathbf{b}_p$ denotes a vector of independent realizations from a binomial distribution with parameter p.

4 Experiments

Three experiments will be presented. The first one for demonstrating the issue stated in the toy example in Section 2 is effectively solved. The second experiment is designed to assess performance of the method on digit synthesis with artificial conditioners, where the simplicity of the images helps make the interpretation of results easier. Finally, a third experiment on conditional face image synthesis shall help better demonstrating the performance of the approach on a real world scenario.

4.1 Digit synthesis with duplicate labels

In the first experiment we address the issue highlighted in Figure 3. To do that, we simply rerun the experiment of doubling the conditioner for digit 9, and this time we trained the architecture depicted in Figure 4, with a 30% chance of missing one of the bits corresponding to 9 during training, and defining F as a single-layer perceptron. Then, we tested the network using missing information, and obtained the results depicted in Figure 5. These results help demonstrate that the network is able to learn that either 10th OR the 11th component of the input vector correspond to the digit 9.

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1)$

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)$

$y = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)$

Figure 5: Output examples of a PCGAN Generator using incomplete conditioners.

Figure 6: Output examples of Generators, when conditioned on 30% missing conditioning entries. CGAN: Standard Conditional GAN. PCGAN: Partially Conditioned GAN, trained with 15% missing conditioning entries.

4.2 Handwritten digit synthesis from partial conditioning information

In order to perform this experiment, we built some synthetic features based upon the MNIST handwritten digits training dataset ([13]) digit labels. We took the set $\mathbb{N}_0^9 = \{0, 1, \ldots, 9\}$ and made a random partition into three sets $\{P_1^{(1)}, P_2^{(1)}, P_3^{(1)}\}$ with at least 3 elements each. Then, for every element $n \in \mathbb{N}_0^9$, we assigned the conditioner

$$y^{(1)} = \begin{cases} (1,0,0) & \text{if} \quad n \in P_1^{(1)} \\ (0,1,0) & \text{if} \quad n \in P_2^{(1)} \\ (0,0,1) & \text{if} \quad n \in P_3^{(1)} \end{cases}$$

We repeated this 10 times to produce the conditioners $y^{(1)}, y^{(2)}, \ldots, y^{(10)}$, so every element $n \in \mathbb{N}_0^9$ has an associated binary conditioning vector $y = [y^{(1)}, y^{(2)}, \ldots, y^{(10)}]$ of size 30. This procedure produces artificial conditioners which are redundant, in the sense that if $A = \cap_{k=1}^{10} P_{i_k}^{(k)}$, there exists $m: \ A = \cap_{k \neq m} P_{i_k}^{(k)}$. This means that the information can be characterised by a subset of the partitions, and hence a model well suited for the proposed problem should exhibit robustness regarding missing 1's on y.

Using these artificial features, we have tested our proposed network architecture (trained with missing information) against a standard Condtional GAN. We have taken the precaution to prevent the additional layer F from adding capacity to the proposed architecture (see Appendix 1 for details). The results generated with these networks when testing with 30% missing conditioners (chosen randomly) are illustrated in Figure 6. Some of the images generated by a standard Conditional GAN mix digits do not resemble any target digits (*i. e.* any of the original classes). The proposed PCGAN, however, is seen to produce better results in terms of image quality.

As a way of quantifying the differences between the methods, we made use of the Fréchet Inception Distance (FID, [14]), where both the real images and model samples are embedded in a learnt feature space, on which the Fréchet distance is computed. This performance measure, however, does not consider the conditioning information, so we also measured the Fréchet Joint Distance (FJD, [15]), which makes use of a joint image-conditioning embedding space. The performance measures obtained when generating using 0%, 15%, 30% and 50% missing entries are depicted in Figure 7. The numbers account for similar FID/FJD values when synthesising with a Conditional GAN or PCGAN from complete conditioning information, but consistently with what was observed on Figure 6, the PCGAN exhibits greater

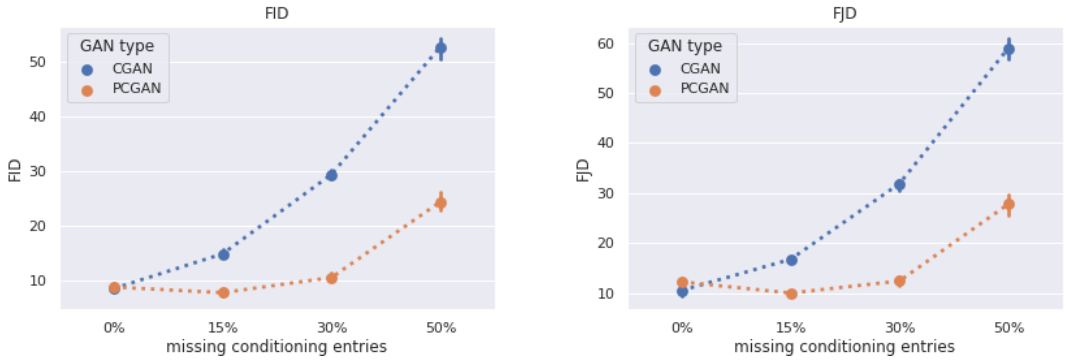

Figure 7: FID and FJD measures for digit generation. CGAN: Standard Conditional GAN. PCGAN: Partially Conditioned GAN, trained with 15% missing conditioning entries.

robustness with respect to missing conditioning entries. In fact, the difference becomes larger as more conditioners are randomly removed when synthesising.

4.3 Face image generation

In order to illustrate how the use of PCGANs translates into a real world application, we shall consider the problem of conditioned human face image generation. To do this, we utilise the CelebA dataset ([16]), which contains human face images along with 40 binary annotations associated with visual characteristics, such as hair colour, expression and skin tone.

Once the network has been trained, if one wants to generate samples from a subset of the conditioning information, it is clear the remaining conditioning features cannot be randomly guessed. Otherwise, one might end up asking the network to synthesize faces with incompatible features, such as *bald* and *brown hair*. To demonstrate that PCGAN can overcome this issue, we trained and tested a Conditional GAN and a PCGAN on the CelebA data set. Results are depicted in Figure 8, where three sets of synthetic images are shown for each method (architecture and model details are in Appendix .2). On each set, the top row corresponds to images generated with complete conditioning information, and the bottom row shows images generated with 30% missing entries (completely at random). In some cases, depending on the attributes, standard Conditional GANs produce samples with some degree of degradation, while the PCGAN exhibits more robustness to the same conditions. For fairness of comparison, the displayed $n-th$ sample of the Conditional GAN and the $n-th$ sample of the PCGAN have the same present and missing attributes.

Performance was evaluated again by measuring the FID and FJD. Figure 9 shows the mean and standard deviation of the measures for ten different trials of each network, and for 4 different degrees of missing conditioning entries (completely at random). As in the case of the previous experiments, both networks exhibit similar performance when synthesising from the complete conditioning set of variables, but performance decays more rapidly for the standard Conditional GAN when the proportion of removed conditioning entries increases. The difference in decay in this case is milder than the one using MNIST, which is consistent with the fact that the conditioning attributes are much less correlated than in the previous experiment.

5 Conclusions and future work

In this study we addressed the problem of building a generative model that can work under partial conditioning information. To do so, we proposed a network architecture that allows for generating from the features that underlay the conditioning information, as well as a strategy for properly training the network.

The experiments conducted demonstrate that the proposed approach can successfully generate images even under a significant proportion of missing conditioning information, unlike the widely used Conditional GAN architecture. Furthermore, it is immediately clear that the architecture is robust to training under some degree of missing conditioning data.

In the future, we intend to extend the proposed model to be able to work with continuous conditioning variables and to find ways to optimally choose the architecture parameters. Additionally, we envisage the use of the concepts proposed herein, for the purpose of anatomical shape synthesis, conditioned on patient meta data.

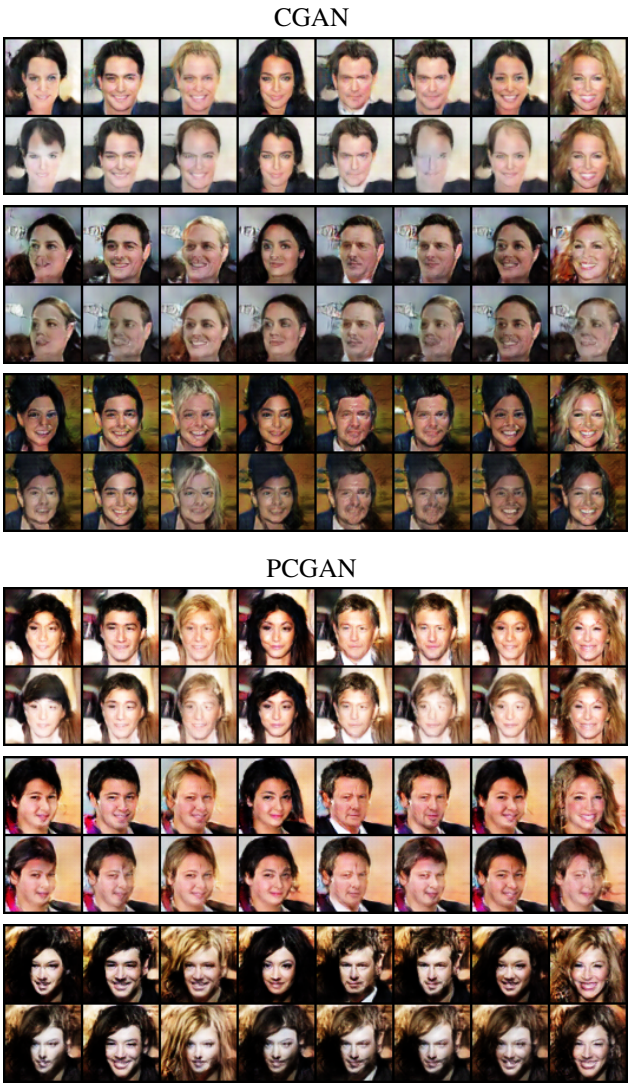

Figure 8: Synthetic face images generated using Conditional GAN and PCGAN. On every set of images, the top row corresponds to complete conditioning information, and bottom row has 30% missing inputs (at random). For fairness in the comparisons, the conditioning entries and missing entries are the same for both networks.

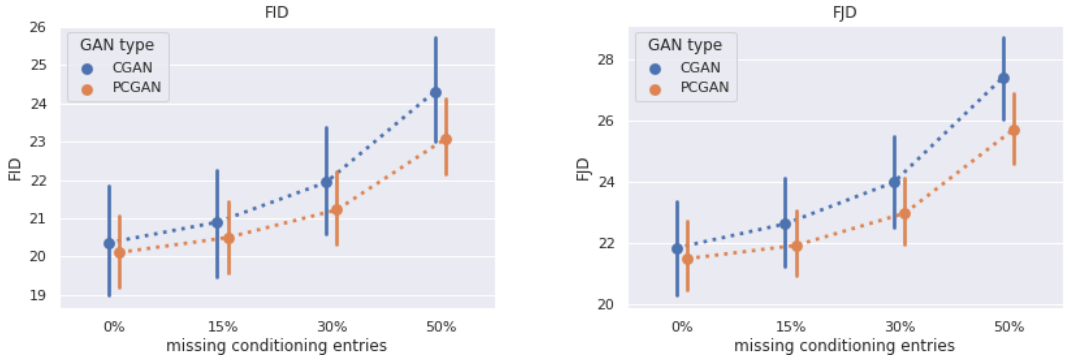

Figure 9: FID and FJD measures for face generation. CGAN: Standard Conditional GAN. PCGAN: Partially Conditioned GAN, trained with 15% missing conditioning entries.

Network structures

In what follows, we present a description of the Network architectures used for the experiments. Should the reader want any additional details, a Python notebook to reproduce the presented experiments can be found at https://github.com/fibarrola/pcgan

.1 Architecture for MNIST

- *Parameters* - Learning rate = 1×10^{-4}. Latent space dimension = 10. Batch size = 128.
- *Generator* - A fully connected layer F is applied to the conditioning vector y, then concatenated with the noise vector z. Then a second fully connected layer and two transposed convolutional layers are applied. Batch normalization is used after every convolutional layer. Activation functions are sigmoid for the last layer and ReLU for all the others.
- *Discriminator for Conditional GAN* - Images go through two consecutive convolutional layers and batch normalization. A fully connected layer F_2 is applied to y. Then the outputs are flattened, concatenated, and put through two more fully connected layers. Activation functions are sigmoid for the last layer and ReLU for all the others.
- *Discriminator for PCGAN* - Same architecture as discriminator for standard Conditional GAN, but $F_2 = F$.

.2 Architecture for CelebA

- *Parameters* - Learning rate = 2×10^{-4}. Latent space dimension = 100. Batch size = 64.
- *Generator* - A transposed convolutional layer F is applied to the conditioning vector y, then concatenated with the output of putting the noise vector z through another transposed convolutional layer. Four transposed convolutional layers are applied afterwards. Batch normalisation is used after every convolutional layer. Activation functions are hyperbolic tangent for the last layer and ReLU for all the others.
- *Discriminator for Conditional GAN* - A fully connected layer is applied to y, and the result is stacked as a 4-th image channel. Then the outputs are fed through five convolutional layers. Activation functions are sigmoid for the last layer and Leaky ReLU (negative slope = 0.2) for all the others.
- *Discriminator for PCGAN* - The same architecture than the discriminator for the standard Conditional GAN, but it is fed $F(y)$ instead of y.

Acknowledgement

AFF is supported by the RAEng Chair in Emerging Technologies (CiET1819/19).

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[2] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

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

[4] T. Lassila, A. Sarrami-Foroushani, S. Hejazi, and A. F. Frangi, "Population-specific modelling of between/within-subject flow variability in the carotid arteries of the elderly," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 36, no. 1, p. e3271, 2020.

[5] E. Abadi, W. P. Segars, B. M. Tsui, P. E. Kinahan, N. Bottenus, A. F. Frangi, A. Maidment, J. Lo, and E. Samei, "Virtual clinical trials in medical imaging: a review," *Journal of Medical Imaging*, vol. 7, no. 4, p. 042805, 2020.

[6] K. Xu, J. Cao, K. Xia, H. Yang, J. Zhu, C. Wu, Y. Jiang, and P. Qian, "Multichannel residual Conditional GAN-leveraged abdominal pseudo-ct generation via dixon mr images," *IEEE Access*, vol. 7, pp. 163 823–163 830, 2019.

[7] J. Choi, K. J. Noh, S. W. Cho, S. H. Nam, M. Owais, and K. R. Park, "Modified conditional generative adversarial network-based optical blur restoration for finger-vein recognition," *IEEE Access*, vol. 8, pp. 16 281–16 301, 2020.

[8] S. Alonso-Monsalve and L. H. Whitehead, "Image-based model parameter optimization using model-assisted generative adversarial networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[9] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh, "Robustness of conditional gans to noisy labels," in *Advances in neural information processing systems*, 2018, pp. 10 271–10 282.

[10] K. K. Thekumparampil, S. Oh, and A. Khetan, "Robust conditional gans under missing or uncertain labels," *arXiv preprint arXiv:1906.03579*, 2019.

[11] D. Harris and S. Harris, *Digital design and computer architecture*. Morgan Kaufmann, 2010.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[13] D. Lee, J. Kim, W.-J. Moon, and J. C. Ye, "Collagan: Collaborative gan for missing image data imputation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2487–2496.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[15] T. DeVries, A. Romero, L. Pineda, G. W. Taylor, and M. Drozdzal, "On the evaluation of conditional gans," *arXiv preprint arXiv:1907.08175*, 2019.

[16] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.