Face generation using conditional generative adversarial networks

Akilesh B
IIT Hyderabad
cs13b1042@iith.ac.in

Abstract

An extension of Generative Adversarial Networks (GANs) is applied to a conditional setting. In a GAN framework, a generator network is tasked with fooling a discriminator network into believing that its own samples are real data. In this work, we add the capability for each network to condition on some attributes which describes the image being generated or discriminated. As we vary the conditional information provided to this modified GAN, we can use the resulting generative model to generate faces with specific attributes from nothing but random noise.We evaluate the likelihood of real-world faces under the generative model and explore how to deterministically control face attributes by modifying the conditional information provided to the model.

1 Introduction

Goodfellow *et al.* showed in Goodfellow et al. [2014a] that the primary cause of neural networks' vulnerability to adversarial perturbation is their linear nature. This explanation was supported by new quantitative results while giving the first explanation of the most intriguing fact about them: their generalization across architectures and training sets.

Generative adversarial nets (GANs) Goodfellow et al. [2014b] were recently introduced as an alternative framework for training generative models in order to sidestep the difficulty of approximating many intractable probabilistic computations.

The GAN framework establishes two distinct players, a generator and discriminator, and poses the two in an adversarial game. The discriminator is tasked with distinguishing between samples from the model and samples from the training data; at the same time, the generator is tasked with maximally confusing the discriminator.

Later, GAN were extended to a conditional mode (cGAN) Mirza and Osindero [2014] where both the generator and discriminator were conditioned on some extra information y. y could be any kind of auxiliary information, such as class labels or data from other modalities. Conditioning can be performed by feeding y into the both the discriminator and generator as additional input layer. In the generator the prior input noise $p_z(z)$, and y are combined in joint hidden representation, and the adversarial training framework allows for considerable flexibility in how this hidden representation is composed. The objective function is a minimax value function:

$$\min_G \max_D \left(\mathbb{E}_{\mathbf{x},\mathbf{y}\sim p_{\text{data}}(\mathbf{x},\mathbf{y})} \left[\log D(\mathbf{x},\mathbf{y}) \right] \right.$$
$$\left. + \mathbb{E}_{\mathbf{y}\sim p_{\mathbf{y}},\, \mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})} \left[\log(1 - D(G(\mathbf{z},\mathbf{y}),\mathbf{y})) \right] \right) .$$

29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

2 Related work

Popular generative models like the Restricted Boltzmann Machine and its many variants have been used successfully in confined settings such as layer-wise pretraining and some applied tasks. But the overall development of generative models as standalone tools has been largely stunted, due to generally intractable problems which arise during maximum-likelihood estimation (e.g. a very expensive normalization / partition term). RBMs, used as generative models in a layer-wise pretraining routine, led to substantial success in many deep learning tasks. They have fallen out of favor in recent years due to difficulties in training and likelihood estimation. More recent work has focused on autoencoders and their capacity as generative models. Vincent *et al.* **?** established the denoising autoencoder (DAE) model, which learns to reconstruct empirical data X from noised inputs $\widetilde{X}$. At a high level, the sampling process follows a Markov chain, where we alternate between sampling reconstructed values P (X/$\widetilde{X}$) and noise C($\widetilde{X}$/X). The Markov chain has a stationary distribution which matches the empirical density model established by the training data. This work has been further developed under the label of generative stochastic networks, where the process of noising C($\widetilde{X}$/X) is generalized to one of decoding into a hidden state using observed data. A related recent idea is the variational autoencoder, which uses neural networks to map between observed and hidden state (latent variables) during EM as a variational approximation of an expensive posterior. As an alternative to these autoencoder models, Goodfellow *et al.* **?** proposed a different approach known as the generative adversarial net (GAN). There is a clear contrast between autoencoders as generative models and the GAN approach in sampling new data. Whereas autoencoders require a special Markov chain sampling procedure, drawing new data from a learned GAN requires only real-valued noise input. Mirza and Osindero implemented a conditional extension to generative adversarial nets and demonstrated some preliminary experiments on MNIST, along with an application to image tagging.

3 Project objective:

1. To develop cGAN on a face image dataset. By varying the conditional information provided to this cGAN, we can use the resulting generative model to generate faces with specific attributes from random noise.
2. To show positive results of using the incorporated conditional data to deterministically control particular attributes of faces sampled from the model.
3. To demonstrate that the model benefits (in terms of test set likelihood) from this external input data.

4 Study Methodology

4.1 Training:

The training process for the above model would be as follows:

1. The generator outputs random RGB noise by default.
2. The discriminator learns basic convolutional filters in order to distinguish between face images and random noise.
3. The generator learns the correct bias (skin tone) and basic filters to confuse the discriminator.
4. The discriminator becomes more attuned to real facial features in order to distinguish between the simple trick images from the generator and real face images. Furthermore, the discriminator learns to use signal in the conditional data y to look for particular triggers in the image.

This process continues until the discriminator is maximally confused. Since the discriminator outputs the probability that an input image was sampled from the training data, we would expect a “maximally confused” discriminator to consistently output a probability of 0.5 for inputs both from the training data and from the generator.

The conditional data y plays a key role in the learning process. If both G and D are MLPs, the framework can be trained by alternating between performing gradient-based updates on G and D.

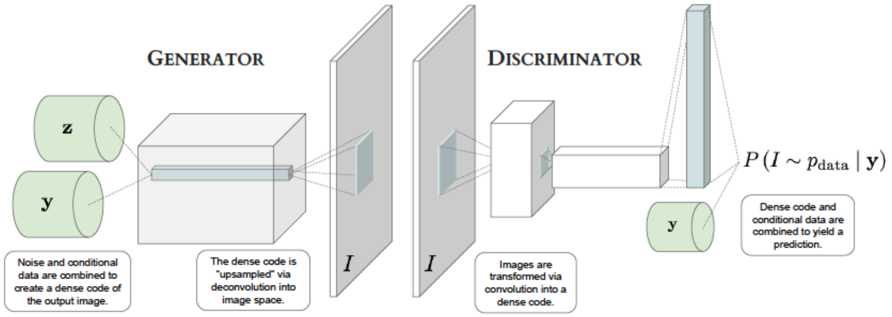
GENERATOR
DISCRIMINATOR
z
y
Noise and conditional data are combined to create a dense code of the output image.
The dense code is "upsampled" via deconvolution into image space.
I
I
Images are transformed via convolution into a dense code.
y
$P\left(I \sim p_{\text{data}} \mid \mathbf{y}\right)$
Dense code and conditional data are combined to yield a prediction.

Figure 1: Overview of the conditional generative adversarial network (cGAN) framework

Goodfellow *et al.* Goodfellow et al. [2014b] suggested training with SGD on D for k iterations (where k is small, perhaps 1) and then training with SGD on G for one iteration.

4.2 Conditional sampling:

We need to sample images from the generator at training time in order to evaluate the two players. The sampling requires both noise z and conditional data y. The random noise can be easily sampled, but care needs to be taken about generating conditional data. In case, we draw these directly from the training examples, the generator might be able to reach some spurious optimum where it learns to reproduce each training image based on the conditional data input. In order to avoid this, conditional data sampling is randomized during training. We build a kernel density estimate $p_y(y)$ (also known as a Parzen window estimate) using the conditional values $\{y_i\}_{i=1}^n$ drawn from the training data. We use a Gaussian kernel, and cross-validate the kernel width σ using a held-out validation set. Samples from this nonparametric density model are used as the inputs to the generator during training.

5 Architecture:

The generator G is a deconvolutional neural network which runs filters over its inputs and expands rather than contracts the inputs into a new representation of higher spatial dimension. Our 3D input space of dimension 2 x 1 x 4 is deconvolved into an output space of dimension 8 x 3 x 1. Each of the four available kernels are of the size 5 x 3. We deconvolve with a kernel stride of 3. The spatial dimension expands from 2 x 1 to 8 x 3, on the other hand, the depth dimension contracts from 4 to 1. This deconvolutional architecture was successfully used by Goodfellow *et al.* Goodfellow et al. [2014b] to reconstruct CIFAR-10 images. We use a single deconvolution in our model. The deconvolution is exactly the inverse of the convolution operation. The deconvolutional forward pass is calculated just as is the backward pass of a convolutional layer, where a column of a 3D input describes the coefficients for a linear combination of the available filters.

The discriminator D is a familiar convolutional neural network, similar to any recent model used in discriminative vision tasks such as image classification. Each convolutional layer has maxout activations. The final output of the convolutions is treated as a dense code describing the input image.

We provide the conditional information y as input in combination with the dense code at the start of the generator feedforward (before deconvolution) and at the end of the discriminator feedforward (after convolution).

6 Dataset

A cropped version of The Labeled Faces in the Wild images dataset Huang et al. [2007] consisting of 13,000 color images, known as LFWcrop Sanderson and Lovell [2009]. The cropping is done to avoid noisy background data and was released by Sanderson and Lovell. The cropped faces in

Table 1: Convolutional layer within discriminator D

Filter size	No of filters	Pool shape	Output volume
–	–	–	32 x 32
8 x 8	64 (x2)	4 x 4	16 x 16
8 x 8	64 (x2)	4 x 4	7 x 7
7 x 7	192 (x2)	2 x 2	5 x 5

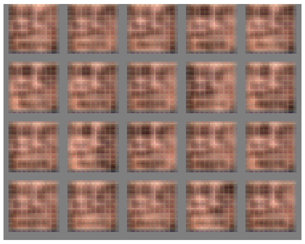

Figure 2: GAN after 110 epochs

LFWcrop exhibit real-life conditions, including misalignment, scale variations, in-plane as well as out-of-plane rotations.

Each image has confidence values for a large number of facial expression attributes and related features, detailed in which we will exploit as conditional data y in these experiments. For example, these attributes include: race (Asian, Indian, black, white), age (baby, child, senior), and emotion (frowning, smiling). There are 73 different attributes in total.

7 Results

7.1 GAN

We first train a GAN, omitting the conditional information y. The rest of the architecture remains the same. After a small number of epochs it learns to reproduce skin color, soon after it learns the positions of basic facial features and textures.

7.2 Conditional GAN

We next train the extended cGAN model on the face image dataset, now using both the image data x and the face attribute data y. In all of the following experiments we use a particular subset of the original face attributes. We eliminate attributes which do not have clear visual effects in the cropped images in our dataset. Out of available 73 attributes we retain 36 attributes.

7.3 Quantitative evaluation

We follow the method proposed in Goodfellow et al. [2014b] to determine the likelihood of a dataset under the learned generator. We establish density estimate using images sampled from the generative

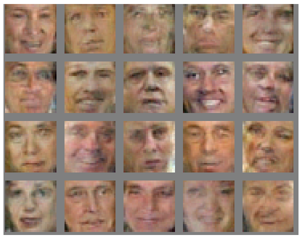

Figure 3: GAN after 430 epochs

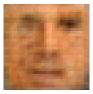

Figure 4: cGAN after 130 epochs

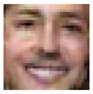

Figure 5: cGAN after 750 epochs

model of a learned cGAN. The kernel function of the Parzen estimate is Gaussian, with a width σ which we select using a validation set. We then calculate the negative log-likelihood of a held-out test set under the learned nonparametric density function.

We then compare the resulting likelihood values for a standard GAN and the conditional GAN and show that learned generative model is better able to generate samples with the conditional density model $p_y(y)$ at hand.

7.4 Comparing the performance of GAN and CGAN

As we can see from figure 4, there is a clear mosaic pattern in early images. The generator deconvolution has a stride of three pixels, while each deconvolutional kernel is a 5 x 5 map. The generator clearly struggles in coordinating these kernels in regions in which they overlap. But we can see in figure 5 that a CGAN performs much better after training for same no. of epochs. In the graph shown in figure 6, we can notice that CGAN performs marginally better than GAN because of the extra conditional information available to it.

8 Conclusion

In this work, we added the ability to condition on arbitrary external information to both the generator and discriminator components. We evaluated the model on the Labeled Faces in the Wild dataset, and demonstrated that the conditional information y could be used to deterministically control the output of the generator.

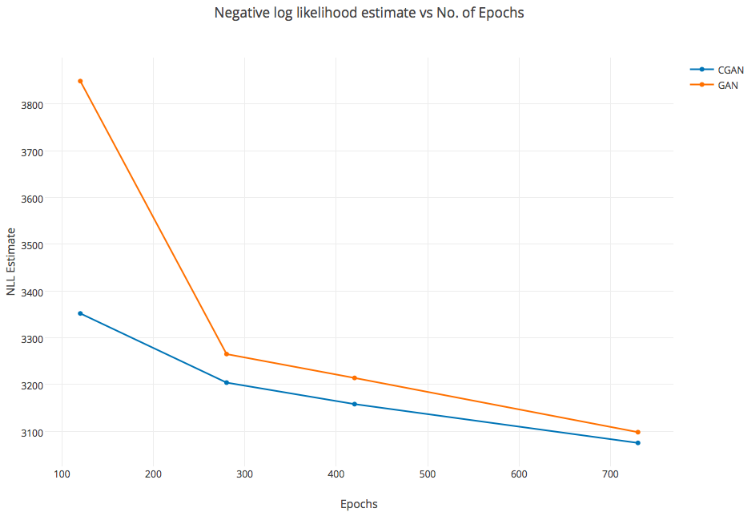

Figure 6: NLL comparison - GAN vs cGAN

The development of a deterministic control slot in the GAN model opens up exciting possibilities for new models and applications. For example, a cGAN could easily accept a multimodal embedding as conditional input y. This y could be produced by a neural language model, allowing us to generate images from spoken or written descriptions of their content.

Bibliography

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

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014b. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. URL `http://arxiv.org/abs/1411.1784`.

Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

Conrad Sanderson and Brian C Lovell. Multi-region probabilistic histograms for robust and scalable identity inference. In *Advances in Biometrics*, pages 199–208. Springer Berlin Heidelberg, 2009.