**Generative Adversarial Networks**

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio

https://arxiv.org/abs/1406.2661
https://arxiv.org/pdf/1406.2661.pdf

## TL;DR

The authors propose a new framework for generative models. Compared to traditional approaches, Generative Adversarial Networks or GANs are composed of two competing networks, which are trained simultaneously. A generative model *G*, and a discriminative model *D.* The G can be thought of a counterfeiter trying to produce fake currency, while the *D* is the police trying to detect the fake money. They compete and improve their methods until the counterfeits produced are indistinguishable from genuine currency. One application of GANs is generating realistic images.

## Introduction

The most striking successes in deep learning have involved discriminative models that map rich input to a class label. This is often accomplished with backpropagation and dropout algorithms, and piecewise linear units(like ReLU). Deep generative models have had much less success due the difficulty of approximation, many hard to control probabilistic computations, and the inability to use piecewise linear units. GANs are able to sidestep these difficulties using adversarial networks, or two competing networks; Competition drives both models to improve. The G can be thought of a counterfeiter trying to produce fake currency, while the D is the police trying to detect the fake money. They compete and improve their methods until the counterfeits produced are indistinguishable from genuine currency. GANs can be trained using backpropagation and dropout algorithms. Approximate inference and Markov chains are not necessary.

## Adversarial Nets

The adversarial framework is most straightforward when both networks are multilayer perceptrons(feed forward neural network). We train *D* to maximize the probability of assigning the correct label to both training examples and samples from *G*. $D(x)$ represents the probability that x came from training data rather than *G*. Simultaneously *G* is trained to minimize $log(1 - D(G(z)))$. $z$ is the input noise. $log(1 - D(G(z)))$ represents how bad the generator model is performing. In practice, $log(1 - D(G(z)))$ saturates. Instead we can **maximize** $log\ D(G(z))$. This function provides much stronger gradients early in learning.

**Theoretical Results**

Pseudo-Code for GAN training:

**Algorithm 1** Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, $k$, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

**for** number of training iterations **do**

  **for** $k$ steps **do**

  - Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of $m$ examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  **end for**

  - Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

**end for**

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

**Experiments**

The adversarial net was tranged on MNIST, the Toronto Face Database, and CIFAR-10.

a) b) c) d)

**Advantages and Disadvantages**

The primary disadvantages of this framework is that $D$ must be synchronized well with $G$ during training. If $G$ is trained too much without updating $D$, $G$ may collapse too many values of $z$ to the same value of $x$. There will be no diversity to the model. All outputs would be the same.

The advantages are primarily computational as mentioned previously. They can also represent sharp distributions, while methods based on Markov chains require that the distribution be somewhat blurry in order for the chains to be able to mix between modes.