Generative Adversarial Networks: Intro and Application to Bias Mitigation

Emily Black

CMU

Spring 2019

Overview

- Fairness Review
- Generative Adversarial Networks
- GANs for Removing Sensitive Attributes

Facebook Engages in Housing Discrimination With Its Ad Practices, U.S. Says

By Katie Benner, Glenn Thrush and Mike Isaac

March 28, 2019

WASHINGTON — The Department of Housing and Urban Development sued Facebook on Thursday for engaging in housing discrimination by allowing advertisers to restrict who is able to see ads on the platform based on characteristics like race, religion and national origin.

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica

May 23, 2016

Sections

The Washington Post

Democracy Dies in Darkness

Public Safety

Police are using software to predict crime. Is it a 'holy grail' or biased against minorities?

By **Justin Jouvenal**

November 17, 2016

Amazon scraps secret AI recruiting tool that showed bias against women

Jeffrey Dastin

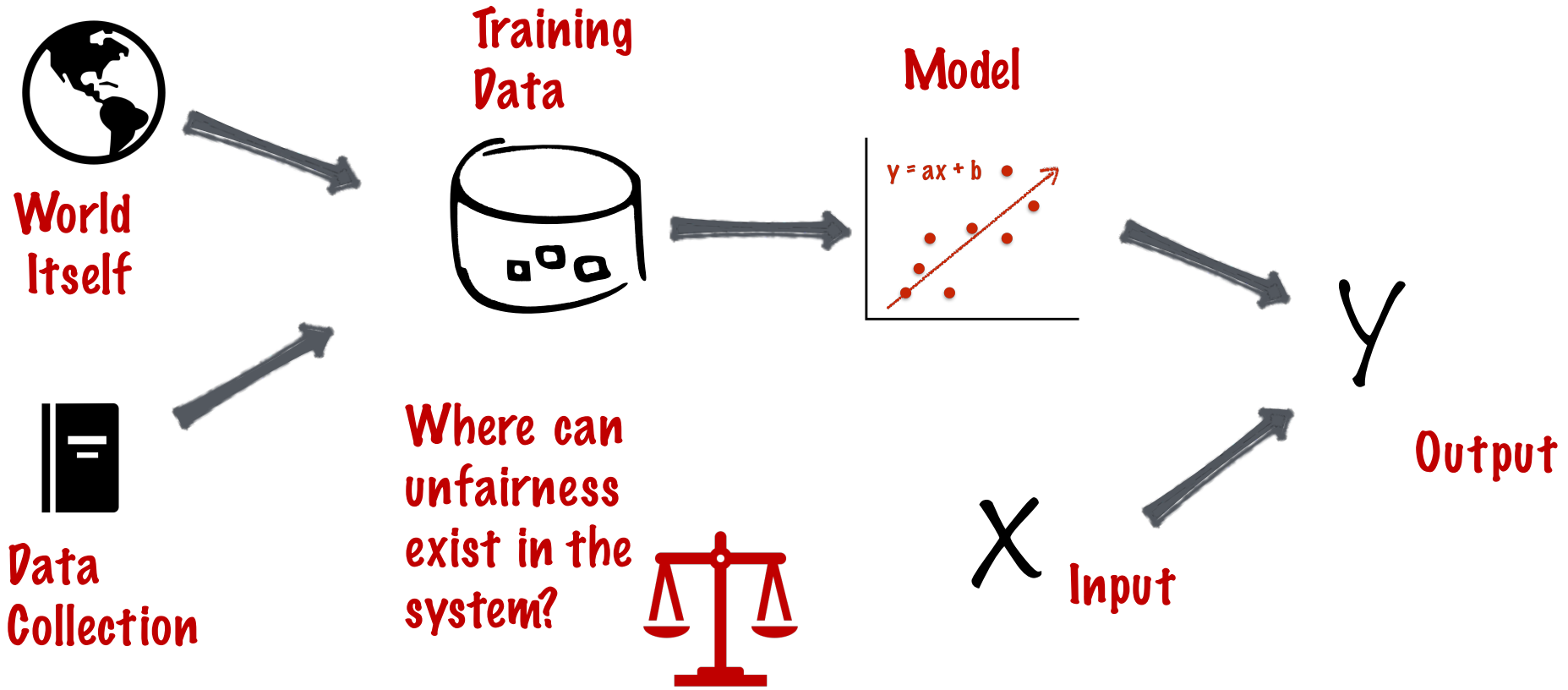
8 MIN READ

How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did

Kashmir Hill Forbes Staff

Welcome to The Not-So Private Parts where technology & privacy collide

Machine Learning Pipeline



How do We Make a Fair Model?

What do you all think?

- Try to ensure your model doesn't augment bias
- Train with a balanced dataset
- Train with fairness constraint
- Don't use protected attribute

But there may be existing bias.

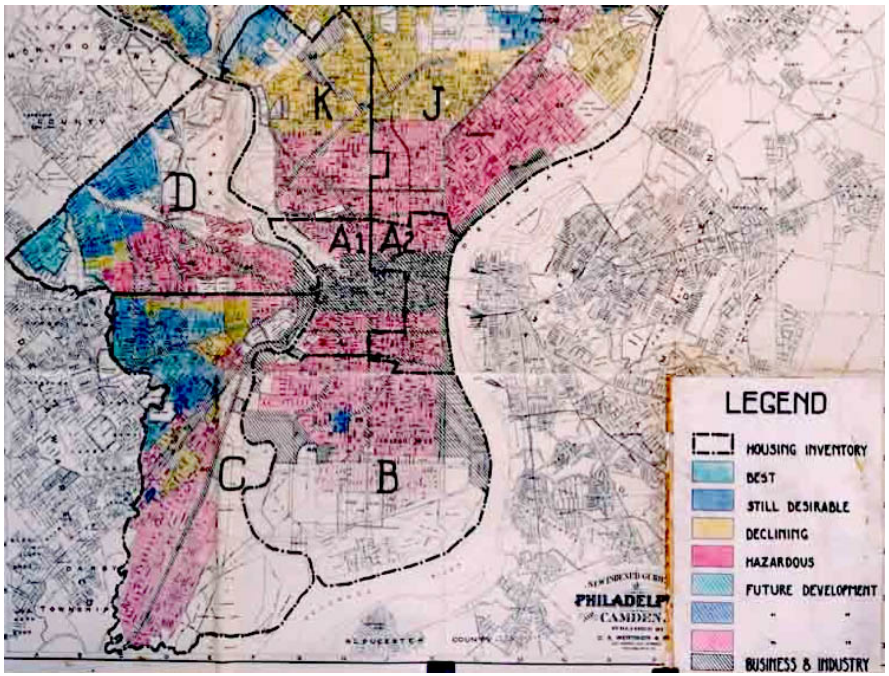
May not always be practical or sufficient.

How do we accomplish this? We will examine this today.

Still have to worry about *proxies* for protected attribute! We will also examine this today.

What Happens if We Take Out the Protected Attribute?

- Neighborhoods in America are largely racially segregated
- A race-blind model could still act in a discriminatory manner by using zipcode to e.g. deny a loan
- Even unintentional discrimination can occur in this way, given a biased prior



Some Amazon Prime services seem to exclude many predominantly black zip codes

Rafi Letzter Apr. 21, 2016, 12:36 PM

More Examples of Proxy Variables

- Purchasing history for medical conditions (pregnancy, or a disease)
- Friends on social media sites to determine sexual orientation
- Facebook currently using in the HUD case: “affinity groups” i.e. your likes on facebook

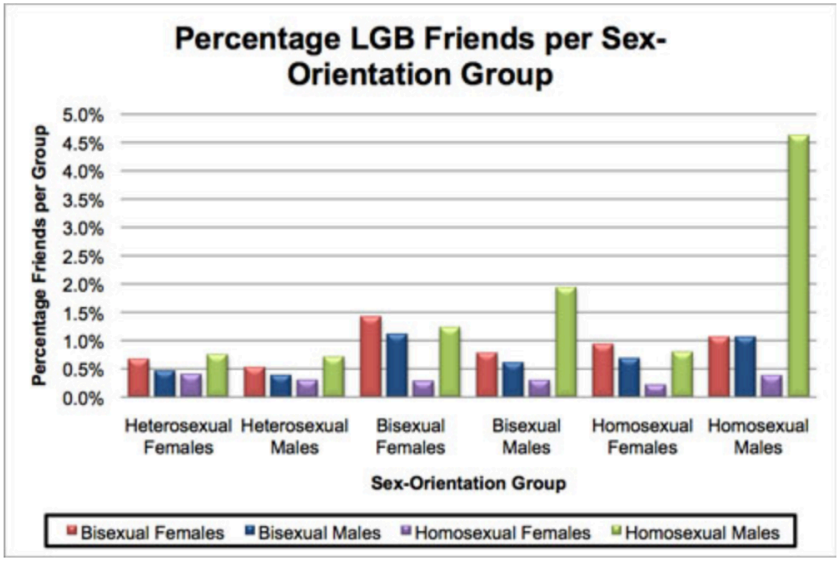


Figure 4: Percentage of LGB friends per sex orientation group.

Making a Fair Model

Proxy variables make it harder to avoid discrimination.

We would like to find a way to mitigate the effect of proxy variables and to better ensure a fair model.

But first…

Overview

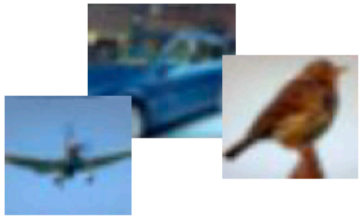
- Fairness Review
- Generative Adversarial Networks
- GANs for Removing Sensitive Attributes

Generative Models

- Collect large amount of data in some domain
- Train generative model to generate data like it

Generative Models

- Given training data generate samples from same distribution (density estimation)



Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Source: Fei-Fei Li et al.

Why Generative Models?

- Some Current applications
  - Semi-supervised learning (pretraining) in cases where labeled data is expensive
  - dataset augmentation
  - image denoising, super-resolution
  - Other thoughts?
- Fairness applications (today)

Generative Models

- Generative adversarial networks (GANs)

- Other models
  - Variational autoencoders (see also: Variational fair autoencoder)
  - Boltzmann machines

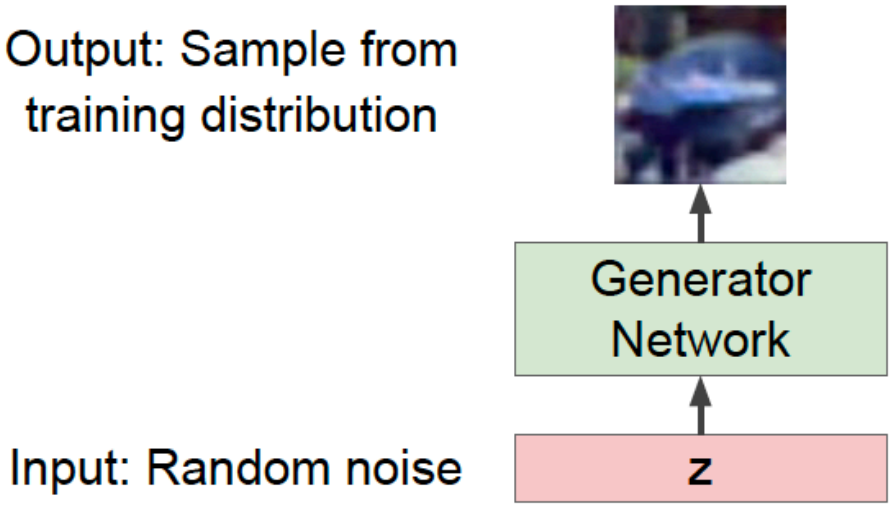
GANs

- Goal: Sample from complex, high-dimensional training distribution

- Approach
  - Sample from a simple distribution (e.g., random noise)
  - Learn transformation from noise to training distribution

- Question
  - How to represent this complex transformation?
  - A neural network!

GANs


Output: Sample from training distribution
Generator Network
Input: Random noise
z

How do we Learn a Good Generator?

- The generator's goal is to map from random noise to an instance from the training distribution.
- How do we learn this mapping?
  - Assign random index to each training point and try to learn the mapping from index to point?
  - Idea: Learn to distinguish between realistic (true) and non-realistic (generated) points to find the space of realistic points.

we could think of the noise as an index into the training distribution

probably too constrained – we don't care about any particular mapping

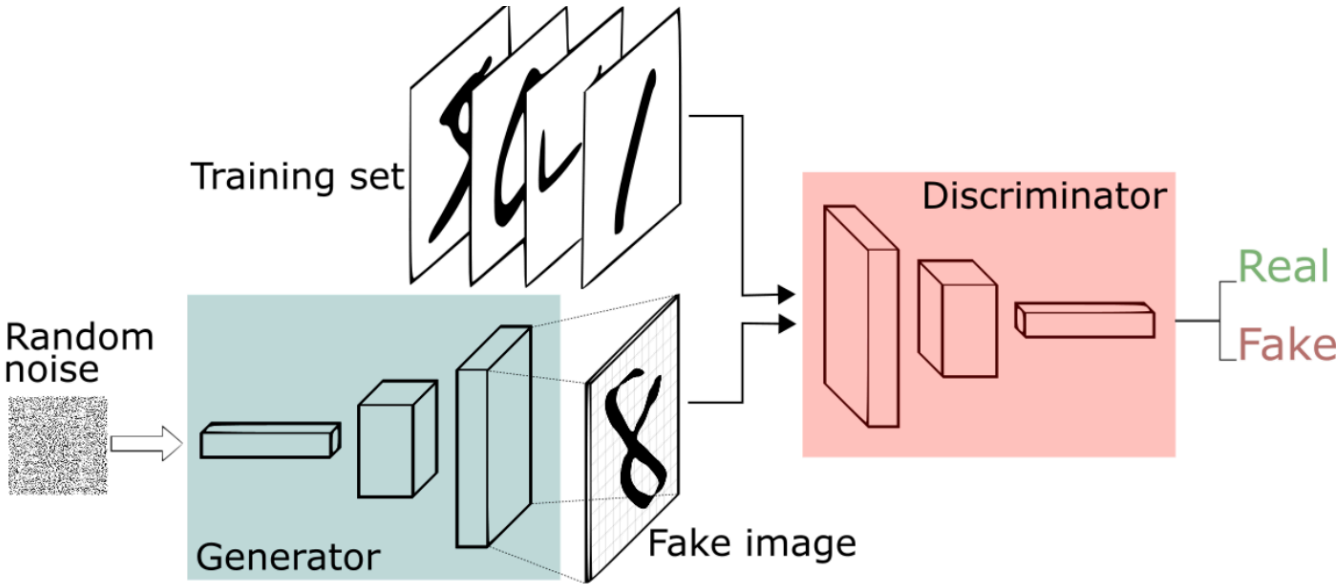
How do we Learn a Good Generator?

- The generator's goal is to map from random noise to an instance from the training distribution.
- How do we learn this mapping?
  - Idea: Learn to distinguish between realistic (true) and non-realistic (generated) points to find the space of realistic points.

A Silly Analogy…

- We can think about the GAN training process as a competition between a cop and a check counterfeiter:
  - Leo is trying to make realistic checks
  - Banks are trying to catch him, they mark all the checks they find out are his
  - As the bank learns Leo's tricks, Leo has to adapt his method to fool the bank again

GAN Training: Two Player Game

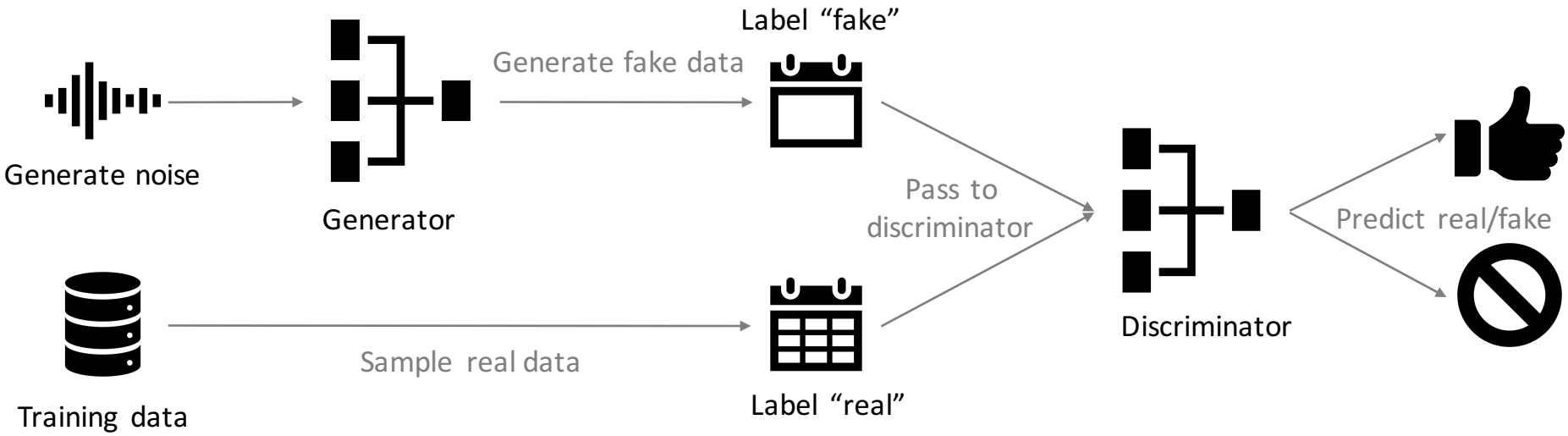


Generative Adversarial Network framework.

Discriminator's job: decipher which of its inputs are real data and which are from the generator (fake)

Generator's job: fool the discriminator by creating images that look like the real images from the training set

Illustration



Training GANs

we can think of this objective as capturing the discriminator's ability to distinguish between real examples and generated examples

Train jointly in **minimax game**

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Discriminator output for real data x

Discriminator output for generated fake data G(z)

generator is in control of these parameters only

discriminator is in control of these parameters only

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)

- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Training GANs

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

here, consider that we are "freezing" the generator's weights

2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

here, consider that we are "freezing" the discriminator's weights

Training GANs

Putting it together: GAN training algorithm

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:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]$$

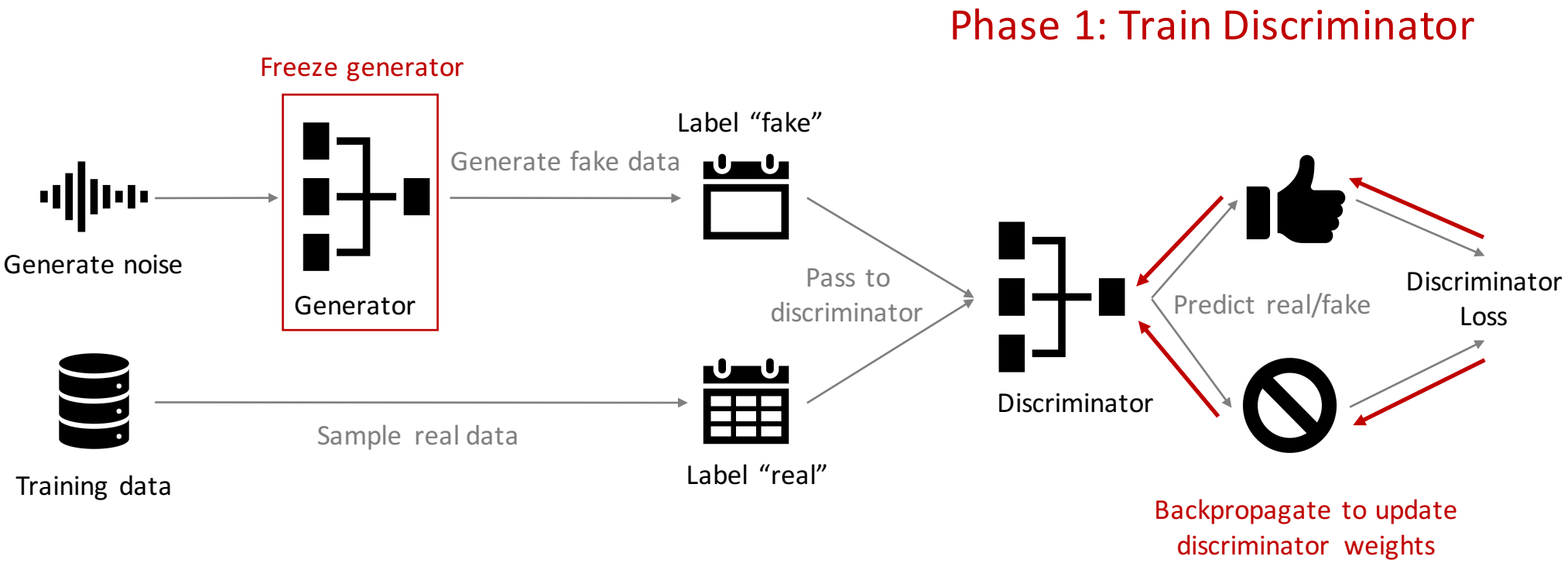
  end for

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

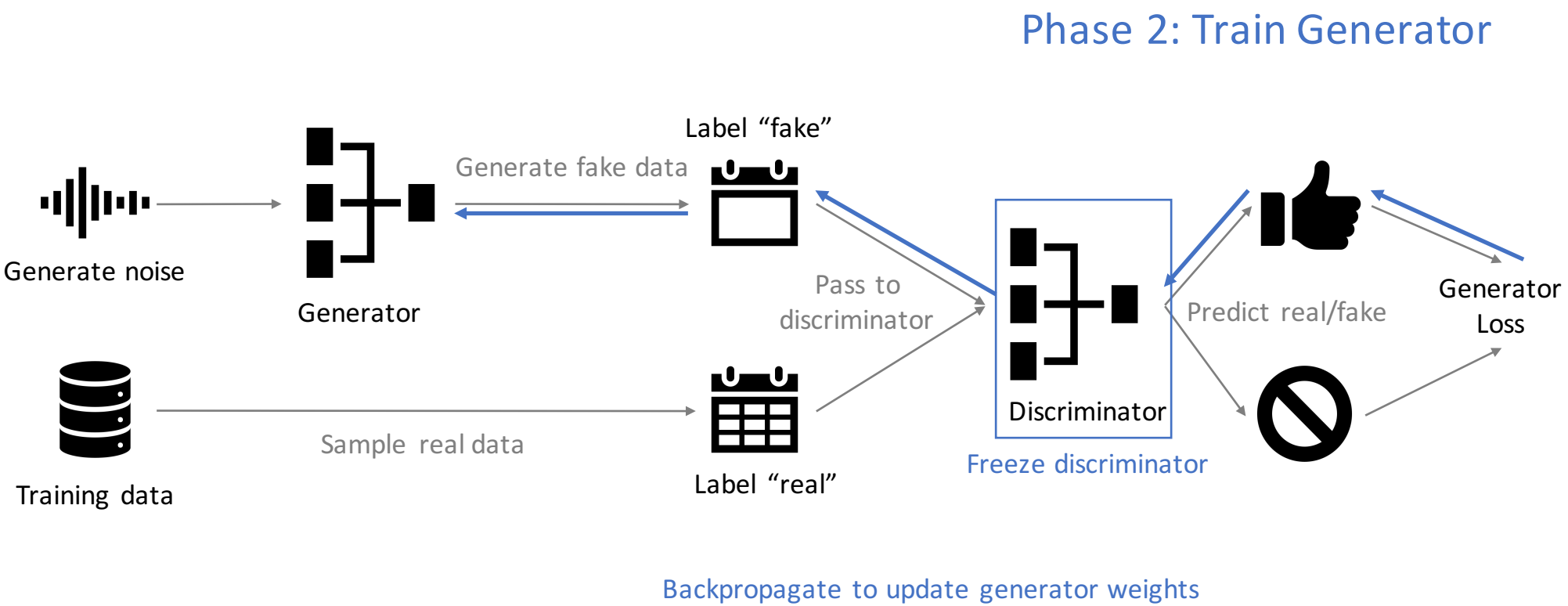
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))$$

end for

Illustration



Illustration



Training GANs

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

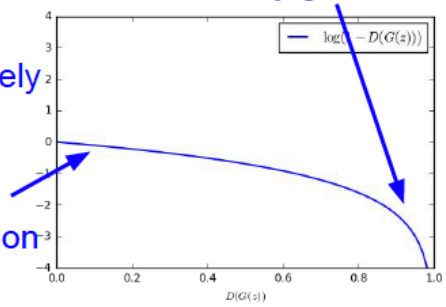
2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

In practice, optimizing this generator objective does not work well!

Gradient signal dominated by region where sample is already good

When sample is likely fake, want to learn from it to improve generator. But gradient in this region is relatively flat!



Training GANs

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Alternate between:

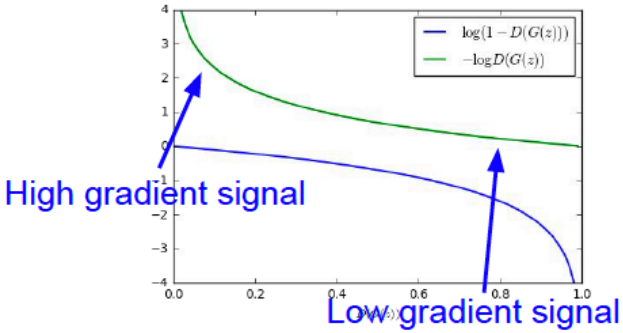
1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Instead: Gradient ascent** on generator, different objective

$$\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$$

Instead of minimizing likelihood of discriminator being correct, now maximize likelihood of discriminator being wrong.
Same objective of fooling discriminator, but now higher gradient signal for bad samples => works much better! Standard in practice.



Convergence theorem

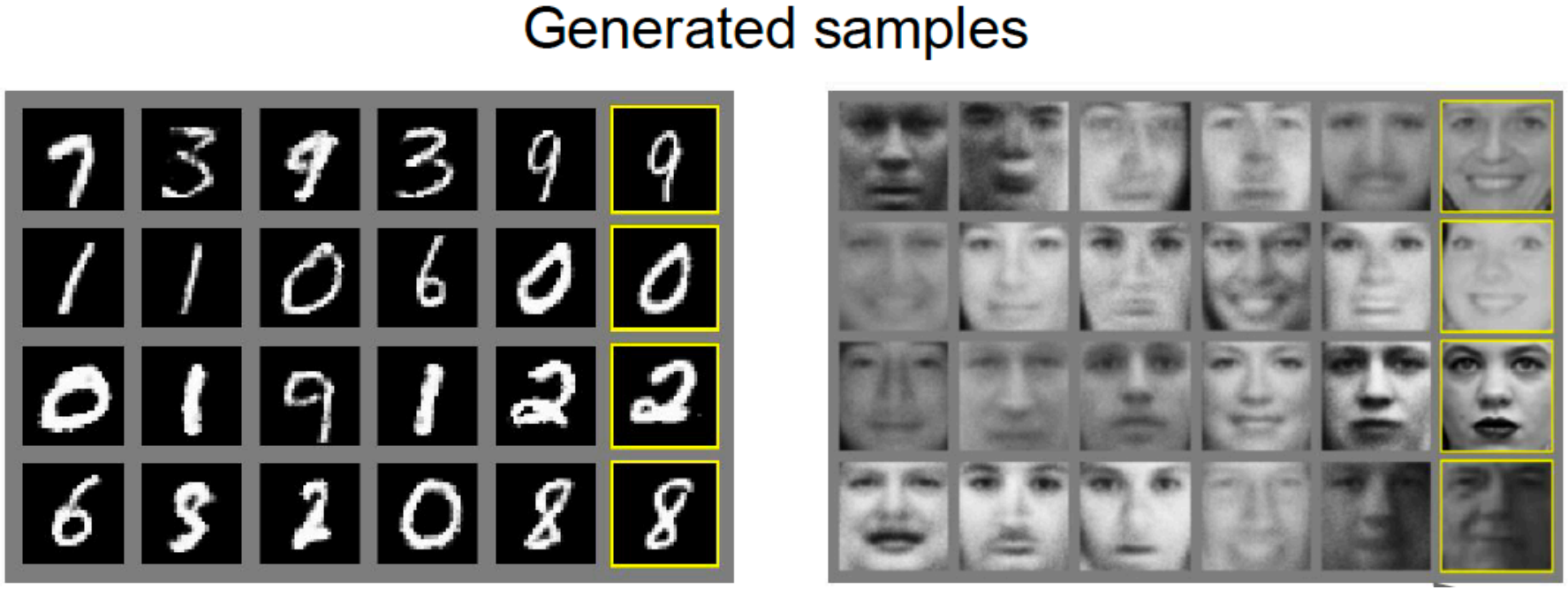
- The training criterion allows one to recover the data generating distribution as G and D are given enough capacity

Proposition 2. *If G and D have enough capacity, and at each step of Algorithm 1, the discriminator is allowed to reach its optimum given G, and p_g is updated so as to improve the criterion*

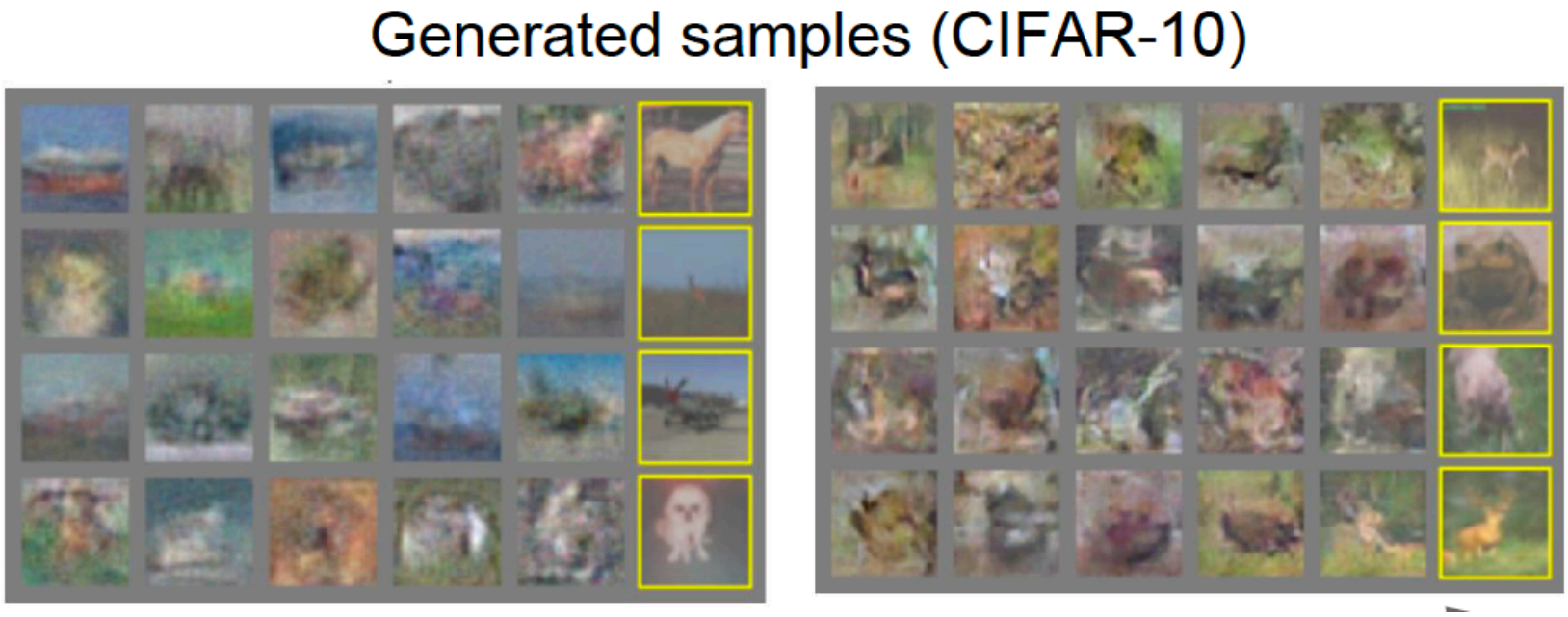
$$\mathbb{E}_{\boldsymbol{x}\sim p_{data}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}\sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))]$$

then p_g converges to p_{data}

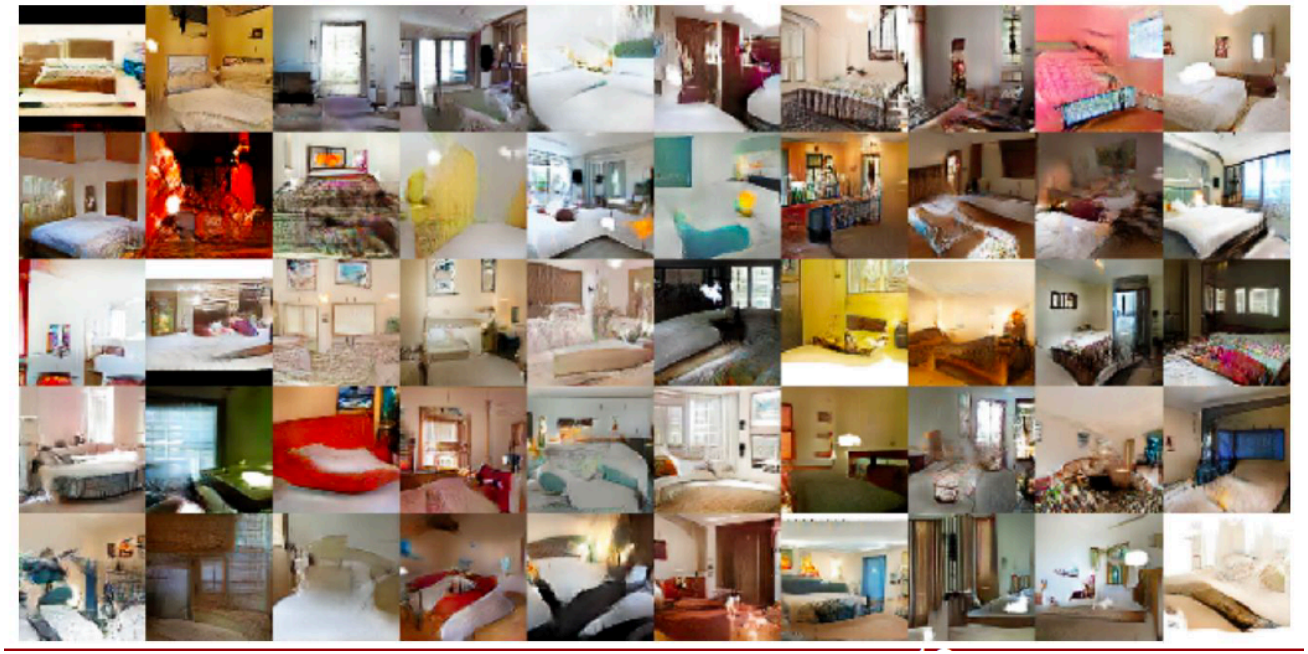
Generated Samples



Generated Samples



GANs: Convolutional architectures



Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Overview

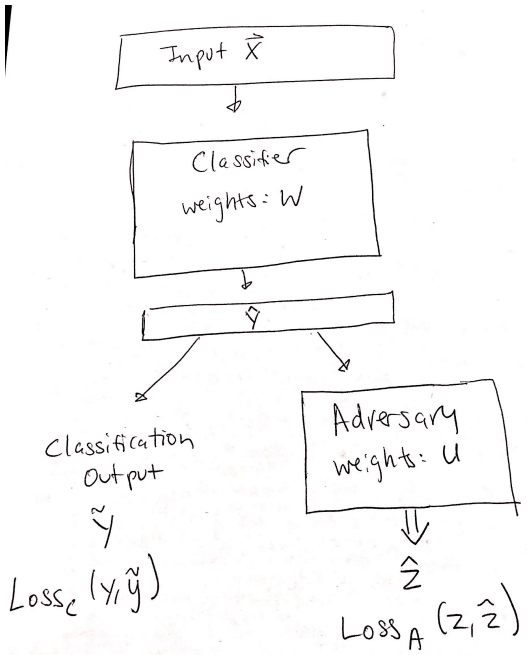
- Fairness Review
- Generative Adversarial Networks
- GANs for Removing Sensitive Attributes

GANs for Fairness

Setup

- Classifier: tries to predict some classification task, such as predicting income based on census data
- Adversary: tries to predict a given protected attribute *z* given the output layer of the classifier
  - The adversary here is likened to the discriminator in a GAN: in a GAN, the generator tries to obfuscate whether a sample is real or fake. Here, the classifier is trying to obfuscate the protected attribute of an input.

GANs for Fairness: Model



Idea: Adversary and Classifier are competing

- Classifier wants to predict the correct output, while also keeping the adversary from predicting the protected attribute
- The Adversary wants to predict the protected attribute
- We give the adversary different information depending upon which fairness objective we want

Difference from GAN

- But the competition here is directly in the gradient updates, not the loss functions
- Idea: remove the part of the gradient update in the classifier that helps the adversary achieve its goal!

Weight Updates

Regular gradient of the weights of the classifier

Projection of the gradient of the *classifier's objective* onto the gradient of the *adversary's objective* w.r.t. the classifier's weights

The negative gradient of the weights of the classifier with respect to the loss of the adversary

$$\nabla_W L_P - \text{proj}_{\nabla_W L_A} \nabla_W L_P - \alpha \nabla_W L_A \qquad (1)$$

Update of the Classifier Weights

$$\nabla_U L_A$$

Update of the Adversary Weights

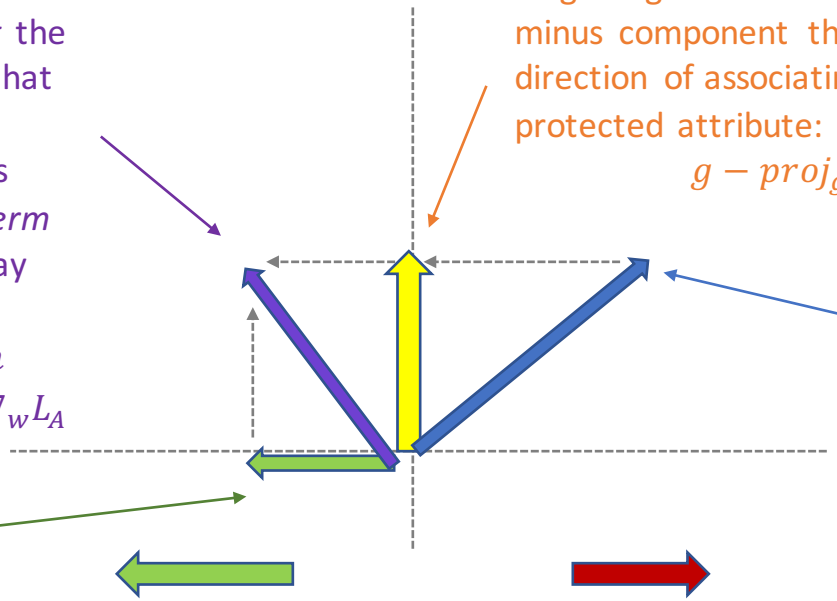
What do all these Vectors Mean?

Our update:
Original gradient *without* the portion (i.e. projection) that goes in the direction of improving the adversary's loss, and *with an extra term* pushing the gradient away from that direction:

$$g - proj_g \nabla_w L_A + h$$
$$= g - proj_g \nabla_w L_A - \alpha \nabla_w L_A$$

Original gradient of the classifier loss minus component that moves in the direction of associating output with protected attribute:

$$g - proj_g \nabla_w L_A$$



Original gradient of the classifier loss w.r.t weights (i.e. weight update) of the classifier,

$$g = \nabla_w L_P$$

$$-\alpha \nabla_w L_A$$

Opposite direction of the adversary's loss
$-\nabla_w L_A$

The direction of the gradient of the adversary's loss with respect to the weights of the classifier:

$$\nabla_w L_A$$

Moving in this direction *helps* the adversary because it moves in the direction of *associating* the protected attribute with the outcome (Dem Parity case)

Fairness Definitions

In other words, you cannot predict Z from $\hat{Y}$: the two are not correlated, i.e. independent

Definition 1. DEMOGRAPHIC PARITY. A predictor $\hat{Y}$ satisfies *demographic parity* if $\hat{Y}$ and Z are independent.

This means that $P(\hat{Y} = \hat{y})$ is equal for all values of the protected variable Z: $P(\hat{Y} = \hat{y}) = P(\hat{Y} = \hat{y}|Z = z)$.

E.g. if the decision is binary, in order for this condition to be true, we need the probability of the positive ($\hat{Y} = 1$) outcome to be the same for all subgroups, as well as the probability of the negative outcome ($\hat{Y} = 0$) to be the same for all subgroups

Fairness Definitions

Definition 3. EQUALITY OF OPPORTUNITY. If the output variable Y is discrete, a predictor $\hat{Y}$ satisfies *equality of opportunity* with respect to a class y if $\hat{Y}$ and Z are independent conditioned on $Y = y$.

This means that, for a *particular* value of the true label Y, $P(\hat{Y} = \hat{y})$ is the same for all values of the protected variable: $P(\hat{Y} = \hat{y}|Y = y) = P(\hat{Y} = \hat{y}|Z = z, Y = y)$

Again, consider the binary Y case, e.g. loan decisions: given that a person is *truly* qualified for a loan, i.e. has (Y= 1), you should not be able to predict their protected attribute from their outcome $\hat{Y}$. Similarly, given that (Y= 0), you should not be able to predict a person's protected attribute from $\hat{Y}$.

Idea: protected attribute does not affect the outcome beyond "underlying state of the world".

BUT, if there truly is a difference—e.g. truly fewer qualified female loan applicants—outcome of the model and protected attribute would still be correlated to reflect this difference

Demographic Parity/ Equal Opportunity

Demographic Parity:

- Feed the adversary $\hat{Y}$
- Intuition: We want Z, $\hat{Y}$ to be independent.
- This is another way of saying we don't want to predict Z from $\hat{Y}$
- The goal of training is that the adversary shouldn't be able to predict Z from $\hat{Y}$: Demographic Parity!

Equal Opportunity

- Constrain the inputs to the adversary to only be from a particular class
- Feed the adversary $\hat{Y}$
- Intuition: We want conditional independence of Z, $\hat{Y}$ given Y. If we restrict what the adversary sees to be just examples from one class, it should not be able to determine Z.
- The goal of training is that this should happen: we achieve Eq Opp!

Results Snapshot: Debiasing Word Embeddings!

biased		debiased	
neighbor	similarity	neighbor	similarity
nurse	1.0121	nurse	0.7056
nanny	0.9035	obstetrician	0.6861
fiancée	0.8700	pediatrician	0.6447
maid	0.8674	dentist	0.6367
fiancé	0.8617	surgeon	0.6303
mother	0.8612	physician	0.6254
fiance	0.8611	cardiologist	0.6088
dentist	0.8569	pharmacist	0.6081
woman	0.8564	hospital	0.5969

Table 1: Completions for `he : she :: doctor : ?`

- Paper reports that non-biased analogies remain intact, i.e. man: woman = he: she
- Perhaps something of a happy medium–certainly debiased, but since performance on the original classification task is also a part of the objective, we can retain acceptable accuracy for some tasks

UCI Adult

Without Debiasing			With Debiasing		
Female	Pred 0	Pred 1	*Female*	Pred 0	Pred 1
True 0	4711	120	True 0	4518	313
True 1	265	325	True 1	263	327
Male	Pred 0	Pred 1	*Male*	Pred 0	Pred 1
True 0	6907	697	True 0	7071	533
True 1	1194	2062	True 1	1416	1840

Table 3: Confusion matrices on the UCI Adult dataset, with and without equality of odds enforcement.

	FNR (before)	FNR (after)	FPR (before)	FPR (after)
Males	.367	.435	.092	.072
Females	.449	.446	.025	.065

- Accuracy goes down from 86% to 84.5%, but results are less biased
- Note that False Negative Rates and False Positive Rates are roughly equal across sex groups

References and acknowledgments

- Fei-Fei Li et al.: Generative models
- OpenAI blog post on Generative Models
- Goodfellow et al.: Generative Adversarial Networks