GANs

Machine Learning: Jordan Boyd-Graber
University of Maryland
SLIDES ADAPTED FROM GRAHAM NEUBIG
Generative Models Ain’t Perfect

- Over-emphasis of common outputs, fuzziness
- Note: this is probably a good idea if you are doing maximum likelihood!

Real MLE Adversarial

(Lotter et al. 2015)

- Fitting conventional prob models focuses on common input
- Can be “fuzzy”
- Still better for smaller amounts of data or if true objective is ML
Adversarial Training

- It’s time for some game theory
Adversarial Training

- It’s time for some game theory
- Create "discriminator" that criticizes generated output
  - Is this example real or not
- Generator is trained to fool discriminator to say it’s real
Adversarial Training

- It’s time for some game theory
- Create “discriminator” that criticizes generated output
  - Is this example real or not
- Generator is trained to fool discriminator to say it’s real
- Contrast with encoder / decoder:
Adversarial Training

- It’s time for some game theory
- Create “discriminator” that criticizes generated output
  - Is this example real or not
- **Generator** is trained to fool discriminator to say it’s real
- Contrast with encoder / decoder: no fixed representation
Training GAN

sample minibatch

sample latent vars.

predict w/ discriminator

\( x_{real} \)

\( x_{fake} \)

\( y \)

discriminator loss
(higher if fail predictions)

generator loss
(higher if make predictions)
Training Equations

Discriminator

\[ \ell_D(\theta_D, \theta_G) = -\mathbb{E}_{x \sim P_{\text{data}}} [\log D(x)] - \mathbb{E}_z [\log (1 - D(G(z)))] \]

- Real data should get high score
- Fake data should get low score
Training Equations

Discriminator

\[ \ell_D(\theta_D, \theta_G) = -\mathbb{E}_{x \sim P_{\text{data}}} \left[ \log D(x) \right] - \mathbb{E}_z \left[ \log(1 - D(G(z))) \right] \]

- Real data should get high score
- Fake data should get low score
Training Equations

Discriminator

\[ \ell_D(\theta_D, \theta_G) = \]
\[ -\mathbb{E}_{x \sim P_{\text{data}}} [\log D(x)] \]
\[ -\mathbb{E}_z [\log(1 - D(G(z)))] \]

- Real data should get high score
- Fake data should get low score
Training Equations

Discriminator

\[ \ell_D(\theta_D, \theta_G) = \]
\[ - \mathbb{E}_{x \sim P_{\text{data}}} [\log D(x)] - \mathbb{E}_z [\log (1 - D(G(z)))] \]

- Real data should get high score
- Fake data should get low score

Generator

\[ \ell_G(\theta_D, \theta_G) = -\ell_D(\theta_D, \theta_G) \]

- If discriminator is very accurate, sometimes better to focus on non-saturating loss
- Focus on where you can confuse discriminator

\[ \mathbb{E}_z [\log D(G(z))] \quad (1) \]
Problems with Training

- GANs are great, but training very hard
- Mode Collapse: generator maps all $z$ to single $x$
- Over-confident discriminator
Problems with Training

- GANs are great, but training very hard
- Mode Collapse: generator maps all $z$ to single $x$ (other examples as side information)
- Over-confident discriminator
Problems with Training

- GANs are great, but training very hard
- Mode Collapse: generator maps all $z$ to single $x$ (other examples as side information)
- Over-confident discriminator (smoothing)
Problems with Discrete Data

- Sample minibatch
- Sample latent vars.
- Convert with generator
- Predict with discriminator

Discrete!
Can’t backprop