Go to **www.menti.com** and use the code **28 48 89**

CS230: Lecture 4

Attacking Networks with Adversarial Examples

-

Generative Adversarial Networks

Kian Katanforoosh

Today's outline

I. Attacking NNs with Adversarial Examples

II. Generative Adversarial Networks

I. Adversarial examples

Discovery (2013): several machine learning models, including state-of-the-art neural networks, are vulnerable to adversarial examples

A. Attacking a network with adversarial examples

B. Defenses against adversarial examples

C. Why are neural networks vulnerable to adversarial examples?

[Szegedy et al. (2013): Intriguing properties of neural networks]
[Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy (2015): Explaining and harnessing adversarial examples]

I. A. Attacking a network with adversarial examples

Goal: Given a network pretrained on ImageNet, find an input image that is not a iguana but will be classified as an iguana.

1. Rephrasing what we want:

Find x such that: $\hat{y}(x) = y_{iguana} = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$

2. Defining the loss function

$$L(\hat{y}, y) = \frac{1}{2}\left\| \hat{y}(W, b, x) - y_{iguana} \right\|_2^2$$

3. Optimize the image

$$\frac{\partial L}{\partial x} \qquad x = x - \alpha \frac{\partial L}{\partial x}$$

[Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy (2015): Explaining and harnessing adversarial examples]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

I. A. Attacking a network with adversarial examples

Question: Will the forged image **x** look like an **iguana**?

?

$$256^{32\times32\times3} \approx 10^{7400}$$

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

I. A. Attacking a network with adversarial examples

Goal: Given a network pretrained on ImageNet, find an input image that is a cat but will be classify as an iguana.

Neural network
(pretrained on ImageNet)

$$\begin{pmatrix} 0.04 \\ 0.85 \\ 0.02 \\ 0.07 \\ 0.01 \\ \vdots \\ 0.02 \end{pmatrix} \begin{matrix} \text{“car”} \\ \text{“iguana} \\ \text{“tomato”} \\ \text{“bike”} \\ \text{“cat”} \\ \\ \text{“crab”} \end{matrix}$$

1. Rephrasing what we want:

Find x such that: $\hat{y}(x) = y_{iguana} = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$

And: $x = x_{cat}$

2. Defining the loss function

$$L(\hat{y}, y) = \frac{1}{2}\left\|\hat{y}(W,b,x) - y_{iguana}\right\|_2^2$$

$$+\lambda\left\|x - x_{cat}\right\|_2^2$$

3. Optimize the image

Network
(pretrained on ImageNet)

$L(\hat{y}, y)$

After many iterations

$$\frac{\partial L}{\partial x} \qquad x = x - \alpha \frac{\partial L}{\partial x}$$

[Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy (2015): Explaining and harnessing adversarial examples]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

I. A. Attacking a network with adversarial examples

92% Cat

94% Iguana

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

I. A. Attacking a network with adversarial examples

$$256^{32\times32\times3} \approx 10^{7400}$$

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

[Alexey Kurakin, Ian J. Goodfellow, Samy Bengio (2017): Adversarial examples in the physical world]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

I. B. Defenses against adversarial examples

Menti

Types of attacks:

- Non-targeted attacks
- Targeted attacks

Knowledge of the attacker:

- White-box
- Black-box

Solution 1

- Create a SafetyNet

x =

y = cat

Solution 2

- Train on correctly labelled adversarial examples

Solution 3

- Adversarial training $L_{new} = L(W,b,x,y) + \lambda L(W,b,x_{adv},y)$
- Adversarial logit pairing $L_{new} = L(W,b,x,y) + \lambda \left\| f(x;W,b) - f(x_{adv};W,b) \right\|_2^2$

[Lu et al. (2017): SafetyNet: Detecting and Rejecting Adversarial Examples Robustly]
[Harini Kannan et al. (2018): Adversarial Logit Pairing]

I. C. Why are neural networks vulnerable to adversarial examples?

See board.

Do neural networks actually understand the data?

[Yuan et al. (2017): Adversarial Examples: Attacks and Defenses for Deep Learning]

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II. Generative Adversarial Networks (GANs)

A. Motivation

B. G/D Game

C. Training GANs

D. Nice results

E. Evaluating GANs

[Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy (2015): Explaining and harnessing adversarial examples]

II.A - Motivation

Motivation: endowing computers with an understanding of our world.

Goal: collect a lot of data, use it to train a model to generate similar data from scratch.

Intuition: number of parameters of the model << amount of data

II.A - Motivation

Probability distributions:

Samples from the “real data distribution”

Samples from the “generated distribution”

[Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas (2017): StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks]

II.B - G/D Game

100-d
random code

$$\begin{pmatrix} 0.47 \\ \vdots \\ 0.19 \end{pmatrix}$$

z

Generator "G"
(Neural Network)

(64,64,3)
generated image

$\neq$

How can we train G to generate images from the true data distributions?

[Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas (2017): StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks]

II.B - G/D Game

100-d
random code

$$\begin{pmatrix} 0.47 \\ \vdots \\ 0.19 \end{pmatrix}$$

z

Generator "G"
(Neural Network)

(64,64,3)
generated image

Run Adam simultaneously on two minibatches
(true data / generated data)

x

Discriminator "D"
(Neural Network)

Gradients

Binary classification

$$\begin{cases} y = 0 & if \quad x = G(z) \\ y = 1 & otherwise \end{cases}$$

Real images
(database)

Probability distributions

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II.B - G/D Game

100-d
random code

$$\begin{pmatrix} 0.47 \\ \vdots \\ 0.19 \end{pmatrix}$$

z

Generator "G" (Neural Network)

(64,64,3)
generated image

x

Discriminator "D" (Neural Network)

Gradients Binary classification

$$\begin{cases} y = 0 & if \quad x = G(z) \\ y = 1 & otherwise \end{cases}$$

End goal: G is outputting images that are indistinguishable from real images for D

Real images (database)

Probability distribution

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II.B - G/D Game

Training procedure, we want to minimize:

Labels: $\begin{cases} y_{real} & \text{is always 1} \\ y_{gen} & \text{is always 0} \end{cases}$

- The cost of the discriminator

$$J^{(D)} = \underbrace{-\frac{1}{m_{real}}\sum_{i=1}^{m_{real}} y_{real}^{(i)}.\log(D(x^{(i)}))}_{\substack{\text{cross-entropy 1:} \\ \text{“D should correctly label real data as 1”}}} \underbrace{-\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}(1-y_{gen}^{(i)}).\log(1-D(G(z^{(i)})))}_{\substack{\text{cross-entropy 2:} \\ \text{“D should correctly label generated data as 0”}}}$$

- The cost of the generator

$$J^{(G)} = -J^{(D)} = \frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(1-D(G(z^{(i)})))$$

“G should try to fool D: by minimizing the opposite of what D is trying to minimize”

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II.C - Training GANs

Saturating cost for the generator:

$$\min\left[\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(1-D(G(z^{(i)})))\right] \Leftrightarrow \max\left[\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D(G(z^{(i)})))\right] \Leftrightarrow \min\left[-\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D(G(z^{(i)})))\right]$$

Non-saturating cost

$$J^{(G)} = -\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D(G(z^{(i)})))$$

Saturating cost

$$J^{(G)} = \frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(1-D(G(z^{(i)})))$$

[Ian Goodfellow (2014): NIPS Tutorial: GANs]

II.C - Training GANs

Note that: $$\min\left[\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(1-D(G(z^{(i)})))\right] \Leftrightarrow \max\left[\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D(G(z^{(i)})))\right] \Leftrightarrow \min\left[-\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D(G(z^{(i)})))\right]$$

New training procedure, we want to minimize:

$$J^{(D)} = \underbrace{-\frac{1}{m_{real}}\sum_{i=1}^{m_{real}} y_{real}^{(i)}.\log(D(x^{(i)}))}_{\substack{\text{cross-entropy 1:} \\ \text{“D should correctly label real data as 1”}}} \underbrace{-\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}(1-y_{gen}^{(i)}).\log(1-D(G(z^{(i)})))}_{\substack{\text{cross-entropy 2:} \\ \text{“D should correctly label generated data as 0”}}}$$

$$J^{(G)} = -\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D(G(z^{(i)})))$$ “G should try to fool D: by minimizing this”

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

Table 1: Generator and discriminator loss functions. The main difference whether the discriminator outputs a probability (MM GAN, NS GAN, DRAGAN) or its output is unbounded (WGAN, WGAN GP, LS GAN, BEGAN), whether the gradient penalty is present (WGAN GP, DRAGAN) and where is it evaluated. We chose those models based on their popularity.

GAN	DISCRIMINATOR LOSS	GENERATOR LOSS
MM GAN	$\mathcal{L}_D^{GAN} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_G^{GAN} = \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$
NS GAN	$\mathcal{L}_D^{NSGAN} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_G^{NSGAN} = -\mathbb{E}_{\hat{x}\sim p_g}[\log(D(\hat{x}))]$
WGAN	$\mathcal{L}_D^{WGAN} = -\mathbb{E}_{x\sim p_d}[D(x)] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$	$\mathcal{L}_G^{WGAN} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
WGAN GP	$\mathcal{L}_D^{WGANGP} = \mathcal{L}_D^{WGAN} + \lambda\mathbb{E}_{\hat{x}\sim p_g}[(\|\|\nabla D(\alpha x + (1-\alpha\hat{x})\|\|_2 - 1)^2]$	$\mathcal{L}_G^{WGANGP} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
LS GAN	$\mathcal{L}_D^{LSGAN} = -\mathbb{E}_{x\sim p_d}[(D(x)-1)^2] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})^2]$	$\mathcal{L}_G^{LSGAN} = -\mathbb{E}_{\hat{x}\sim p_g}[(D(\hat{x}-1)^2]$
DRAGAN	$\mathcal{L}_D^{DRAGAN} = \mathcal{L}_D^{GAN} + \lambda\mathbb{E}_{\hat{x}\sim p_d+\mathcal{N}(0,c)}[(\|\|\nabla D(\hat{x})\|\|_2 - 1)^2]$	$\mathcal{L}_G^{DRAGAN} = \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$
BEGAN	$\mathcal{L}_D^{BEGAN} = \mathbb{E}_{x\sim p_d}[\|\|x - \mathrm{AE}(x)\|\|_1] - k_t\mathbb{E}_{\hat{x}\sim p_g}[\|\|\hat{x} - \mathrm{AE}(\hat{x})\|\|_1]$	$\mathcal{L}_G^{BEGAN} = \mathbb{E}_{\hat{x}\sim p_g}[\|\|\hat{x} - \mathrm{AE}(\hat{x})\|\|_1]$

[Lucic, Kurach et al. (2018): Are GANs Created Equal? A Large-Scale Study]

II.C - Training GANs

Simultaneously training G/D?

for num_iterations:

for k iterations:
update D

update G

Non-saturating cost

$$J^{(G)} = -\frac{1}{m_g}\sum_{i=1}^{m_g}\log(D(G(z^{(i)})))$$

Saturating cost

$$J^{(G)} = \frac{1}{m_g}\sum_{i=1}^{m_g}\log(1-D(G(z^{(i)})))$$

[Ian Goodfellow (2014): NIPS Tutorial: GANs]

BatchNorm with GANs:

Generated images (batch 1)

Generated images (batch 2)

[Ian Goodfellow (2014): NIPS Tutorial: GANs]

II.C - Training GANs

BatchNorm with GANs:

Assume no batchnorm

Assume batchnorm

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II.C - Training GANs

BatchNorm with GANs:

BatchNorm

$$Z = \{z^{(1)}, \ldots, z^{(m)}\}$$

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} z^{(i)}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (z^{(i)} - \mu_B)^2$$

$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

$$\tilde{z}^{(i)} = \gamma z_{norm}^{(i)} + \beta$$

Reference BatchNorm

$$R = \{r^{(1)}, \ldots, r^{(m)}\}$$

$$Z = \{z^{(1)}, \ldots, z^{(m)}\}$$

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} r^{(i)}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m} (r^{(i)} - \mu_B)^2$$

$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

$$\tilde{z}^{(i)} = \gamma z_{norm}^{(i)} + \beta$$

Virtual BatchNorm

$$R = \{r^{(1)}, \ldots, r^{(m)}\}$$

$$Z = \{z^{(1)}, \ldots, z^{(m)}\}$$

For k = 1….m

$$\mu_B = \frac{1}{m+1}\left(z^{(k)} + \sum_{i=1}^{m} r^{(i)} \right)$$

$$\sigma_B^2 = \frac{1}{m+1}\left((z^{(k)} - \mu_B)^2 + \sum_{i=1}^{m} (r^{(i)} - \mu_B)^2 \right)$$

$$z_{norm}^{(k)} = \frac{z^{(k)} - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

$$\tilde{z}^{(k)} = \gamma z_{norm}^{(k)} + \beta$$

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II.C - Training GANs

Recap: GANs' training tips

- Modification of the cost function
- Keep D up-to-date with respect to G (k update for D / 1 update for G)
- Use Virtual Batchnorm
- (not presented but important) One-sided label smoothing

And a lot more, GANs are hard to train!

[Soumith et al. (2016): GanHacks]
[Lucic, Kurach et al. (2018): Are GANs Created Equal? A Large-Scale Study]

II.D - Nice results

Operation on codes

Code 1

$$\begin{pmatrix} 0.12 \\ \vdots \\ 0.92 \end{pmatrix}$$

Generator "G" (Neural Network)

(64,64,3) generated image

Code 2

$$\begin{pmatrix} 0.47 \\ \vdots \\ 0.19 \end{pmatrix}$$

Generator "G" (Neural Network)

(64,64,3) generated image

Code 3

$$\begin{pmatrix} 0.42 \\ \vdots \\ 0.07 \end{pmatrix}$$

Generator "G" (Neural Network)

(64,64,3) generated image

Code 1 Code 2 Code 3

$$\begin{pmatrix} 0.12 \\ \vdots \\ 0.92 \end{pmatrix} - \begin{pmatrix} 0.47 \\ \vdots \\ 0.19 \end{pmatrix} + \begin{pmatrix} 0.42 \\ \vdots \\ 0.07 \end{pmatrix}$$

Generator "G" (Neural Network)

Man with glasses - man + woman = woman with glasses

[Radford et al. (2015): UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS]

II.D - Nice results

Image Generation:

Samples from the “generated distribution”

[Zhang et al. (2017): StackGAN++]

II.D - Nice results

Pix2Pix:

https://affinelayer.com/pixsrv/ by Christopher Hesse.

[Isola et al. (2017): Image-to-Image Translation with Conditional Adversarial Networks]

II.D - Nice results

Super-resolution image:

Generator "G" (Neural Network)

[Ledig et al. (2016): Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network]

II.D - Nice results

CycleGANs:

[Zhu, Park et al. (2017): Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks]

II.D - Nice results

Goal: Convert horses to zebras on images, and vice-versa.

Data?

Unpaired images

Horse images

Zebra images

Architecture?

Cost?

[Zhu, Park et al. (2017): Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks]

II.D - Nice results

Architecture?

[Zhu, Park et al. (2017): Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks]

II.D - Nice results

Loss to minimize?

$$J^{(D1)} = -\frac{1}{m_{real}}\sum_{i=1}^{m_{real}}\log(D1(h^{(i)})) - \frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(1 - D1(G1(H^{(i)})))$$

$$J^{(G1)} = -\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D1(G1(H^{(i)})))$$

$$J^{(D2)} = -\frac{1}{m_{real}}\sum_{i=1}^{m_{real}}\log(D2(z^{(i)})) - \frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(1 - D2(G2(Z^{(i)})))$$

$$J^{(G2)} = -\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\log(D2(G2(Z^{(i)})))$$

$$J^{cycle} = -\frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\| G2(G1(H^{(i)}) - H^{(i)} \|_1 - \frac{1}{m_{gen}}\sum_{i=1}^{m_{gen}}\| G1(G2(Z^{(i)}) - Z^{(i)} \|_1$$

$$\boxed{J = J^{(D1)} + J^{(G1)} + J^{(D2)} + J^{(G2)} + \lambda J^{cycle}}$$

Kian Katanforoosh, Andrew Ng, Younes Bensouda Mourri

II.D - Nice results

CycleGANs:

Face2ramen

[成冨 志優: Face2Ramen using CycleGAN]
[Zhu, Park et al. (2017): Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks]

II.D - Nice results

Other applications:

- Beaulieu-Jones et al., Privacy-preserving generative deep neural networks support clinical data sharing.
- Hwang et al., Learning Beyond Human Expertise with Generative Models for Dental Restorations.
- Gomez et al., Unsupervised cipher cracking using discrete GANs.
- Many more…

II.E - Evaluating GANs

How to evaluate GANs?

Human annotators

Inception Score (IS)

$$IS(G) = \exp\left(\mathrm{E}_{x \sim p_g}\left[KL(p(y \mid x) \,\|\, p(y))\right]\right)$$

Measure of image quality

Measure of image diversity

$$IS(G) = \exp\left(\frac{1}{m_b}\sum_{i=1}^{m_b}\sum_{j=1}^{n_y}\hat{y}_j^{(i)}\left(\log \hat{y}_j^{(i)} - \log\left(\frac{1}{m_b}\sum_{k=1}^{m_b}\hat{y}_j^{(k)}\right)\right)\right)$$

Also: Fréchet Inception Distance (FID)

[Lucic, Kurach et al. (2018): Are GANs Created Equal? A Large-Scale Study]
[Salimans et al. (2016): Improved Techniques for Training GANs]

Announcements

For Wednesday 10/24, 11am:

C2M3

- **Quiz: Hyperparameter tuning, Batch Normalization, Programming Frameworks**
- **Programming assignment: Tensorflow**

C3M1 and C3M2

- **Quiz: Bird recognition in the city of Peacetopia (case study)**
- **Quiz: Autonomous driving (case study)**

This Friday 10/19:

- **Hands-on session this Friday**

Check out the project example code! (cs230-stanford.github.io)

Meet with your mentor TA, to go over project proposal feedback (attendance required.)