Introduction to Machine and Deep Learning Theory

Adversarial Robustness and Attacks I: in Digital Domain

Aleksandr Petiushko

Lomonosov MSU
Faculty of Mechanics and Mathematics

March 29th, 2023

AP

Plan

1. CNN great success
2. CNN lack of robustness
3. ℓ_0-based adversaries

AP

Major approach

- Usual training:

$$\theta^{t+1} = \theta^t - \eta \nabla_{\theta^t} L(f_{\theta^t}(x), y)$$

- Adversarial perturbation:

$$x^{t+1} = x^t + \eta \nabla_{x^t} L(f_\theta(x^t), y)$$

AP

Robustness in Machine Learning

Robustness [informally]

Ability for a machine learning algorithm a to provide similar outputs on the similar data (i.e. having the same class or other invariant features)

Two types of **Robustness** in ML:

Generalization

Dataset issue: algorithm needs to be robust if the dataset to evaluate it differs (sometimes significantly: we can treat it is a distribution shift) from the training dataset

Adversarial Robustness

Noise issue: algorithm needs to provide the similar output w.r.t. both clean and noisy images (where the model of noise is the topic to consider itself)

For now we'll consider the **Adversarial Robustness**.

AP

Adversarial Robustness topics

- Perturbations (also called 'adversarial *attacks*'): how to generate noise to fool the neural net

AP

Adversarial Robustness topics

- Perturbations (also called 'adversarial *attacks*'): how to generate noise to fool the neural net
- Defense: how to diminish the influence of adversarial perturbations

AP

Adversarial Robustness topics

- Perturbations (also called 'adversarial *attacks*'): how to generate noise to fool the neural net
- Defense: how to diminish the influence of adversarial perturbations
- Certification (or verification): how to provide theoretical guarantees on the noise level not fooling the neural net

AP

Adversarial Robustness topics

- Perturbations (also called 'adversarial *attacks*'): how to generate noise to fool the neural net
- Defense: how to diminish the influence of adversarial perturbations
- Certification (or verification): how to provide theoretical guarantees on the noise level not fooling the neural net
  - Certification will be considered the next time

AP

Questions to be answered

Question1

How good is the human expert currently in comparison to Neural Nets for known datasets?

Question2

How stable are the current top neural net solutions subject to input data? How easy can they be fooled?

CNN vs Human[1]

[1]Image credit: `https://spectrum.ieee.org`

Human expert VS CNN

ImageNet[2] (1000-class image DB)

- Human expert top-5 error[3]: 5.1%
- CNN top-5 error[4]: 0.98%

Labeled Faces in the Wild[5] (famous faces DB)

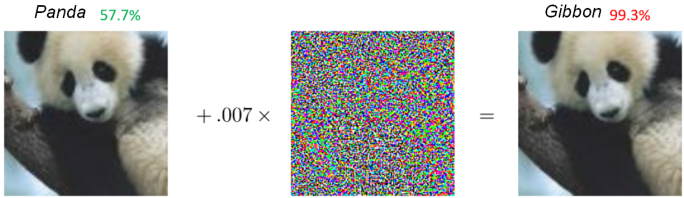
- Human expert verification error[6]: 2.47%
- CNN verification error[7]: 0.17%

LFW

[2] `http://www.image-net.org/`
[3] Andrej Karpathy blog
[4] Yuan, Lu, et al. "Florence: A new foundation model for computer vision." 2021
[5] `http://vis-www.cs.umass.edu/lfw/`
[6] Kumar N. et al. “Attribute and simile classifiers for face verification.” 2009
[7] Deng J. et al. “Arcface: Additive angular margin loss for deep face recognition.” 2018

CNN instability

- It turned out that one can add to the input almost invisible to the human eye perturbation in such a way that this perturbation completely changes the CNN output
- E.g. classification result from "Panda" changes to "Gibbon"[8]

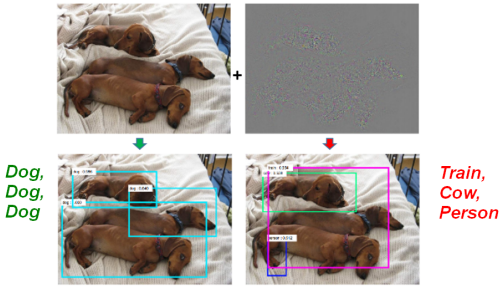


Such almost invisible perturbations leading to changing of the CNN output are called **adversarial examples** / **perturbations** (or **adversarial attacks** on CNN)

[8] Image credit: `https://arxiv.org/pdf/1412.6572.pdf`

Different types of NN to attack

Detection and segmentation[9] CNNs:



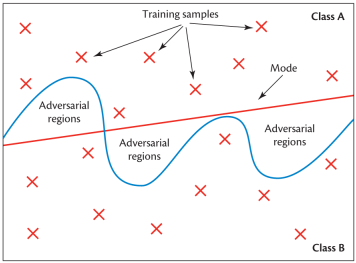
And even NN for text processing (QA, question answering systems)[10]:

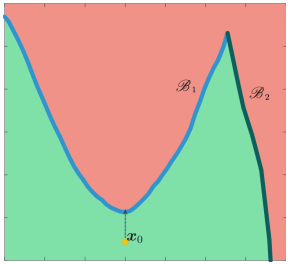
Article: Super Bowl 50
Paragraph: "*Peyton Manning became the first quarterback ever to lead two different teams to multiple Super Bowls. He is also the oldest quarterback ever to play in a Super Bowl at age 39. The past record was held by John Elway, who led the Broncos to victory in Super Bowl XXXIII at age 38 and is currently Denver's Executive Vice President of Football Operations and General Manager. Quarterback Jeff Dean had jersey number 37 in Champ Bowl XXXIV.*"
Question: "*What is the name of the quarterback who was 38 in Super Bowl XXXIII?*"
Original Prediction: John Elway
Prediction under adversary: Jeff Dean

[9] Xie C. et al. "Adversarial examples for semantic segmentation and object detection." 2017
[10] Jia R. et al. "Adversarial examples for evaluating reading comprehension systems." 2017

One of the main sources of adversarial examples to exist

- One of the main reasons for this neural nets behavior on similar inputs: **lack of NN robustness**
- Namely, the separating classification boundaries often are very close to the training examples, and it is very easy to "go abroad"[11,12]



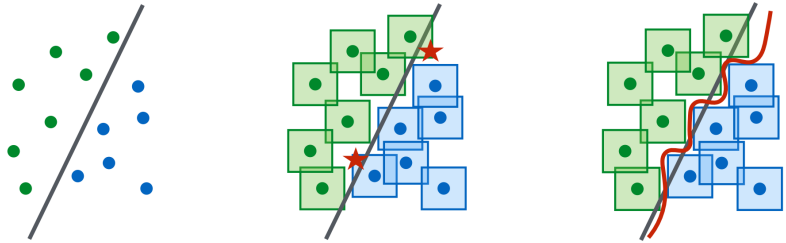


[11] Image credit: `https://secml.github.io/`

[12] Fawzi, Alhussein, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. "Robustness of classifiers: from adversarial to random noise." 2016

Easy defense method

- We can change the output of classification neural net by a subtle per-pixel perturbation ⇒ why not to add for any training example its whole per-pixel vicinity (based on some norm — e.g., ℓ_∞) during the training process[13]



[13] Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks." 2017

Easy, but not realistic defense method

- Suppose the input image size is 100×100 pixels, 3 RGB colors (8-bit color depth)
- Suppose that human eye cannot distinguish the changes in color of 2 units out of 256: at every pixel we can allow for ± 1 perturbation for every color channel
- Then for every training example we need to add into the training dataset the following number of its "pixel neighbors"

$$2^{3\times 100\times 100} = 2^{30000} = (2^{10})^{3000} \approx (10^3)^{3000} = 10^{9000}$$

- It is much more the number of atoms in the observable part of the Universe (10^{80})!
- As a consequence, not very realistic

AP

Realistic defense method

- Let us not iterate over the entire vicinity of the training example, but to use only those points from example vicinity, which are the closest to the separating surface
- This training method is called **A**dversarial **T**raining (AT)[14]

Adversarial training: pros

- No need in iterating over the vicinity of huge cardinality
- In general, protects against the method of adversarial example generation

Adversarial training: cons

- Procedure of *good* adversarial examples generation usually works slowly (significantly slower than 1 backprop iteration)
- Protects **only** against the method of generating adversarial examples used for adversarial training

[14] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." 2014

Definitions

- $x \in B = [0,1]^{C \times M \times N}$ — input image $C \times M \times N$, where C — number of color channels (1 for grayscale, 3 for RGB)
- y — correct class label for input x
- θ — parameters of CNN-classifier
- $L(\theta, x, y)$ — loss function
- $f(x)$ — output of classifier (recognized class), and we are trying to make $f(x) = y$ when training
- $r \in B = [0,1]^{C \times M \times N}$ — the additive perturbation for the input x

AP

Definition of adversarial example and robustness

Goal of adversarial attack

To change the output of the classifier f from the correct class label to the incorrect one by means of minimal in terms of some norm ℓ_p perturbation r:

1. $||r||_p \to \min$ so as:
2. $f(x) = y$ (initially the output is correct)
3. $f(x + r) \neq y$ ("break" the CNN output with perturbation r)
4. $x + r \in B$ (still in the space of correct images)

Classifier robustness

To find the perturbation class $S(x, f) \subseteq B$ so as the classifier will not change its output:

$$f(x + r) = f(x) = y \quad \forall r \in S(x, f)$$

AP

ℓ_p norms

Let us remind the most used ℓ_p norms for $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$:

- ℓ_2: $||x||_2 = \sqrt{\sum_{i=1}^n x_i^2}$
- ℓ_1: $||x||_1 = \sum_{i=1}^n |x_i|$
- ℓ_∞: $||x||_\infty = \max_i |x_i|$
- ℓ_0: $||x||_0 = \sum_{i=1}^n \mathbf{1}_{x_i \neq 0}$

Remark. For $0 < p < 1$ the functional ℓ_p such as $||x||_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$ is not a norm[15]

[15] **Exercise**: Prove it

Adversarial examples taxonomy (1)

Target

- **Untargeted**: only need to change the classifier's output
- **Targeted**: need to change the classifier's output to the specific class y_t

Awareness

- **White-box**: adversary knows everything about the classifier (architecture and its weights)
- **Black-box**: adversary knows only partial details about the classifier (usually only classifier's output)

Application

- **Digital**: for digital record application only (e.g., photo of an object)
- **Real-world**: to be applied in the real environment (e.g., the object itself)

Adversarial examples taxonomy (2)

Universality

- **Input-aware**: the perturbation r depends on the input x
- **Universal**: the perturbation r is to be applied for any input x

Transfer

- **Non-transferable**: successfully working only for a small range of classifiers
- **Transferable**: successfully working for a wide range of classifiers (but at the same time can be non-universal ones)

- The hardest case — targeted black-box real-world universal transferable adversarial examples
- This talk will be devoted to white-box adversarial examples

AP

Adversarial examples: success criteria

Let us introduce the success criteria $S(A, Z)$ of an algorithm A for generating adversarial examples $r_A(x)$ using the set $Z \ni (x^i, y^i)$:

- For untargeted adversarial examples:

$$S(A, Z) = \frac{\sum_i \mathbf{1}\{f(x^i) = y^i\} \cdot \mathbf{1}\{f(x^i + r_A(x^i)) \neq y^i\}}{\sum_i \mathbf{1}\{f(x^i) = y^i\}}$$

- For adversarial examples targeted towards the class y_t:

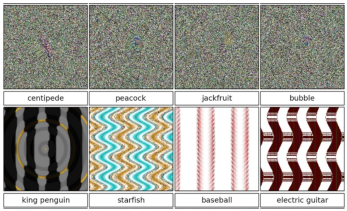
$$S(A, Z, y_t) = \frac{\sum_i \mathbf{1}\{f(x^i) = y^i\} \cdot \mathbf{1}\{f(x^i + r_A(x^i)) = y_t\}}{\sum_i \mathbf{1}\{f(x^i) = y^i\}}$$

Remark. Apparently $S(A, Z, y_t) \leq S(A, Z)$

AP

Adversarial examples on images: forerunner

- Initially CNN robustness was studied towards how adequate the output of CNN using different inputs
- It was turned out that some examples (structured ot not) exist that produce as the output of CNN any class with any probability
- Such examples were called “fooling images”[16] and were generated by evolutionary algorithms



[16] Nguyen, Anh, Jason Yosinski, and Jeff Clune. “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images.” 2014

Generation method: L-BFGS-B (1)

- The first proposed method[17] of generation of adversarial examples used ℓ_2-norm
- Examples targeted towards class $y_t \neq y$ were used
- Functional to minimize with constraints $x + r \in B$, $c = const$:

$$c||r||_2 + L(\theta, x + r, y_t) \rightarrow \min_r$$

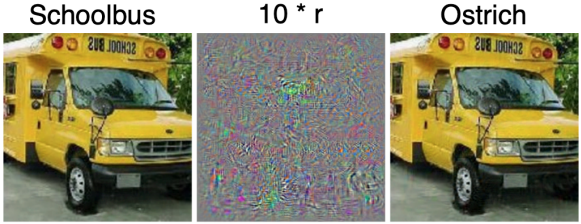
- As an optimization method the L-BFGS-B[18] (**L**imited memory **B**royden–**F**letcher–**G**oldfarb–**S**hanno algorithm with **B**ox constraints) algorithm was used — quasi-Newtonian minimization method with memory/variables constraints
- Generated examples were transferable to some extent to some other architectures

[17] Szegedy, Christian, et al. "Intriguing properties of neural networks." 2013

[18] Byrd, Richard H., et al. "A limited memory algorithm for bound constrained optimization." 1995

Generation method: L-BFGS-B (2)

Adversarial example:



Transferability:

	FC10(10^{-4})	FC10(10^{-2})	FC10(1)	FC100-100-10	FC200-200-10	AE400-10
FC10(10^{-4})	100%	11.7%	22.7%	2%	3.9%	2.7%
FC10(10^{-2})	87.1%	100%	35.2%	35.9%	27.3%	9.8%
FC10(1)	71.9%	76.2%	100%	48.1%	47%	34.4%
FC100-100-10	28.9%	13.7%	21.1%	100%	6.6%	2%
FC200-200-10	38.2%	14%	23.8%	20.3%	100%	2.7%
AE400-10	23.4%	16%	24.8%	9.4%	6.6%	100%

AP

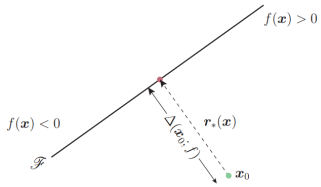
Generation method: CW

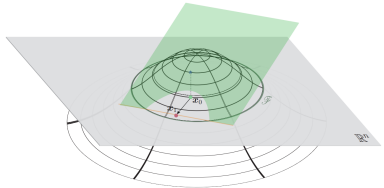
- Recall the functional to minimize: $c||r||_2 + L(\theta, x + r, y_t) \to \min_r$
- Let us change the functional for targeted adversarial examples: $f(x) = y_t \Leftrightarrow g(x) \leq 0$
  - E.g., $g(x) = \max_{i \neq t} F(x)_i - F(x)_t$
  - where F(x) — the SoftMax (probability) output, or logits before the SoftMax
- Then consider $||r||_p + c \cdot g(x + r) \to \min_r$, where $x + r \in B$ and $c > 0$ is some constant
- This method is called CW[19] (**C**arlini-**W**agner) algorithm and can be applied for any ℓ_p
- Here we're optimizing not the proxy-function (like loss function), but the real output (probabilities / logits)

[19] Carlini, Nicholas, and David Wagner. "Towards evaluating the robustness of neural networks." 2016

Generation method: DeepFool (1)

- **Idea**: to project the point x_0 onto class separating surface
- In case of binary classifier $\operatorname{sign} f(x) = \operatorname{sign}(w^T x + b)$:
  - Direction: $-\operatorname{sign} f(x_0) \frac{w}{||w||_2}$
  - Length: $\frac{|f(x_0)|}{||w||_2}$
  - $\Rightarrow$ Perturbation: $r = -\frac{f(x_0)}{||w||_2^2} w$
- In case of non-linear separating surface $f(x)$:
  - applying Taylor formula: $f(x) \approx f(x_0) + \nabla f(x_0)(x - x_0)$
  - and substitute in the formula for r the expression $w = \nabla f(x_0)$





Generation method: DeepFool (2)

- DeepFool[20] iterative algorithm for any classifier

Algorithm 1 DeepFool for binary classifiers

input: Image $\boldsymbol{x}$, classifier f.
output: Perturbation $\hat{\boldsymbol{r}}$.
Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
while $\text{sign}(f(\boldsymbol{x}_i)) = \text{sign}(f(\boldsymbol{x}_0))$ **do**
    $\boldsymbol{r}_i \leftarrow -\frac{f(\boldsymbol{x}_i)}{\|\nabla f(\boldsymbol{x}_i)\|_2^2} \nabla f(\boldsymbol{x}_i)$,
    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$,
    $i \leftarrow i + 1$.
end while
return $\hat{\boldsymbol{r}} = \sum_i \boldsymbol{r}_i$.

- There is a natural generalization in case of multi-class classifier

[20] Moosavi-Dezfooli, Seyed-Mohsen, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks." 2015

Generation method: FGSM

- L-BFGS-B method is not very fast and requires the external (in relation to NN under attack) optimizer
- **Proposition**: to use only the linear part of loss function in the vicinity of x and walk by its gradient — FGSM[21] (**F**ast **G**radient **S**ign **M**ethod):

$$r = \epsilon \cdot \text{sign}(\nabla_x L(\theta, x, y_t))$$

  where $0 < \epsilon < 1$ — some constant
- **Reminder**: for NN weights optimization the backpropagation method is used where the gradient is taken w.r.t. NN weights, i.e. $\nabla_\theta L(\theta, x, y)$
- For most CNNs the norm ℓ_∞ is used because it is correlated with the process of how a human eye perceive the visual information

[21] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." 2014

Generation method: I-FGSM (PGD)

- Often the linear approximation of the loss function is quite rough so one step of FGSM sometimes is not enough for generating a good adversarial example
- To overcome it the so called I-FGSM[22] (**I**terative FGSM) is used, which allows to move to the classification boundary more accurately
- If Π_B — projection to B, then in case of untargeted adversarial examples

$$x^{n+1} = \Pi_B(x^n + \alpha \cdot \operatorname{sign} \nabla_x L(\theta, x, y)), \quad x^0 = x, \alpha = \frac{\epsilon}{T}, n \leq T - 1$$

- Taking into account $||x - x_{adv}||_\infty \leq \epsilon$, authors proposed to make $T = \min(256\epsilon + 4, 320\epsilon)$ steps
- This method is also called PGD[23] (**P**rojected **G**radient **D**escent)

[22] Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. "Adversarial examples in the physical world." 2016

[23] Madry A. et al. "Towards deep learning models resistant to adversarial attacks." 2017

Generation method: MI-FGSM

- **Remark**: Generation methods are becoming more and more similar to NN weights optimization steps
- **Idea**: let's use the gradient smoothing — MI-FGSM[24] (**M**omentum I-FGSM):

$$g^{t+1} = \mu \cdot g^t + \frac{\nabla_x L(\theta, x^t, y)}{||\nabla_x L(\theta, x^t, y)||_1}$$

$$x^{t+1} = \Pi_B(x^t + \alpha \cdot \text{sign}(g^{t+1}))$$

$$x^0 = x, g^0 = 0, \alpha = \frac{\epsilon}{T}, t \leq T - 1$$

[24] Dong, Yinpeng, et al. "Boosting adversarial attacks with momentum." 2017

Comparison of FGSM-like attacks

	Attack	Inc-v3	Inc-v4	IncRes-v2	Res-152	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{ens}$
Inc-v3	FGSM	72.3*	28.2	26.2	25.3	11.3	10.9	4.8
	I-FGSM	**100.0***	22.8	19.9	16.2	7.5	6.4	4.1
	MI-FGSM	**100.0***	**48.8**	**48.0**	**35.6**	**15.1**	**15.2**	**7.8**
Inc-v4	FGSM	32.7	61.0*	26.6	27.2	13.7	11.9	6.2
	I-FGSM	35.8	**99.9***	24.7	19.3	7.8	6.8	4.9
	MI-FGSM	**65.6**	**99.9***	**54.9**	**46.3**	**19.8**	**17.4**	**9.6**
IncRes-v2	FGSM	32.6	28.1	55.3*	25.8	13.1	12.1	7.5
	I-FGSM	37.8	20.8	**99.6***	22.8	8.9	7.8	5.8
	MI-FGSM	**69.8**	**62.1**	99.5*	**50.6**	**26.1**	**20.9**	**15.7**
Res-152	FGSM	35.0	28.2	27.5	72.9*	14.6	13.2	7.5
	I-FGSM	26.7	22.7	21.2	**98.6***	9.3	8.9	6.2
	MI-FGSM	**53.6**	**48.9**	**44.7**	98.5*	**22.1**	**21.7**	**12.9**

Based on it, MI-FGSM is one of the most successful ones.

AP

Generation method: JSMA

- ℓ_∞-based adversaries are imperceptible, but require all pixels to change. In the physical world it is not realistic — we can only change a part of the scene
- JSMA[25] (**J**acobian-based **S**aliency **M**ap **A**ttack) — one of the first ℓ_0-based methods of generation where the number of changed pixels is more important than its deltas
- **Idea**: to change the pixels with the major impact to the gradient w.r.t. the input for the needed class for targeted attack
- It can be done iteratively while gradually adding pixels into perturbation area r
- **Remark**: $F(x)$ — SoftMax layer output, pixels are added by pairs (empirically shown that it's simpler)

Algorithm 3 Increasing pixel intensities saliency map
$\nabla \mathbf{F}(\mathbf{X})$ is the forward derivative, Γ the features still in the search space, and t the target class

Input: $\nabla \mathbf{F}(\mathbf{X})$, Γ, t

for each pair $(p, q) \in \Gamma$ **do**
  $\alpha = \sum_{i=p,q} \frac{\partial \mathbf{F}_t(\mathbf{X})}{\partial \mathbf{X}_i}$
  $\beta = \sum_{i=p,q} \sum_{j \neq t} \frac{\partial \mathbf{F}_j(\mathbf{X})}{\partial \mathbf{X}_i}$
  if $\alpha > 0$ and $\beta < 0$ and $-\alpha \times \beta >$ max **then**
    $p_1, p_2 \leftarrow p, q$
    $max \leftarrow -\alpha \times \beta$
  end if
end for
return p_1, p_2

[25] Papernot, Nicolas, et al. "The limitations of deep learning in adversarial settings." 2015

Generation method: One pixel

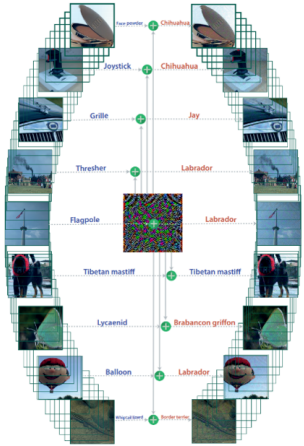
- One pixel adversarial example[26] — extreme case of ℓ_0-based generation method
- **Idea**: to apply the evolutionary algorithm (differential evolution[27])
- Population consists of 400 examples each defined by 5 numbers: 2 coordinates and 3 color channels
- Offspring generation — the linear combination of 3 random parents

[26] Su, Jiawei, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks." 2017

[27] Storn, Rainer, and Kenneth Price. "Differential evolution — a simple and efficient heuristic for global optimization over continuous spaces." 1997

Universal adversarial examples

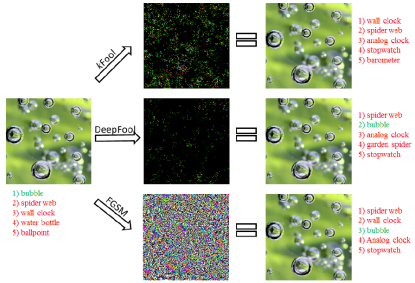
- Until now all adversarial examples have been constructed as a function of input x
- It turns out that it is possible to build so called "universal" example[28] which is the function of the whole training dataset X
- **Idea**: to find perturbation r approximately equally far away from all classes from X
- To the right is the universal perturbation for every input changing the output of the classifier

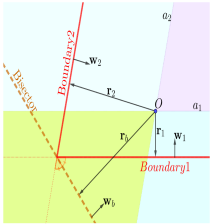


[28] Moosavi-Dezfooli, Seyed-Mohsen, et al. "Universal adversarial perturbations." 2016

Adversarial examples: multi-class case

- When recognizing multiple classes (usual case) then just "movement down" of the correct class from the first place by probability is not enough for some tasks
- It is the case to use so called "*top-k* adversarial examples"[29], where $k > 1$
- The work uses nice but simple geometric concepts (bisectors for top-k class boundaries)

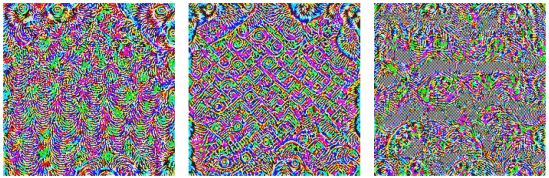




[29] Tursynbek N. et al. "Geometry-Inspired Top-k Adversarial Perturbations". 2020

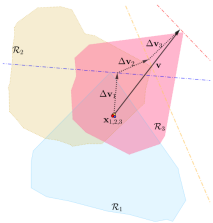
Universal adversarial examples: multi-class case

- It is possible to construct the universal adversarial example having the same property: “shifting” the correct class out of the top-k predicted classes by probability



(a) ResNet-18 (b) VGG-16 (c) MobileNetV2

Figure 7. Result of kUAP ($k = 3$) to different deep neural networks for ILSVRC2012



Taxonomy of generation methods

In general, the adversarial examples generation methods (in Computer Vision) can be divided into the following types:

- Using ℓ_2-based norm (including geometric ones): most convenient for classical optimization
- Using ℓ_∞-based norm: correspond to perception process by the human eye of any visual information
- Using ℓ_0-based norm: the area of perturbation is minimized, but not the delta value per pixel

Anyway it is not enough for generation of physically plausible adversarial examples.

AP

Takeway notes

- NNs for now are much better than human expert in controlled conditions
- NNs are unstable w.r.t. its input

AP

Thank you!