ECEN 685-885 - Machine Learning in Cyber-security

Dr. Mahmoud Nabil

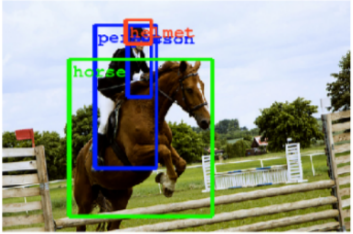
Dr. Mahmoud Nabil
mnmahmoud@ncat.edu

North Carolina A & T State University

Talk Overview

1. What are Adversarial Examples?
2. Why Adversarial Examples Exists?
3. How to Craft Adversarial Examples
4. Adversarial Examples and Transferability Attack
5. Adversarial Examples Counter Measures

Advances in Machine Learning



(Szegedy et al, 2014)

...recognizing objects and faces....

(Taigmen et al, 2013)

(Goodfellow et al, 2013)

...solving CAPTCHAS and reading addresses...

(Goodfellow et al, 2013)

Are machine learning models that intellignet?

Outline

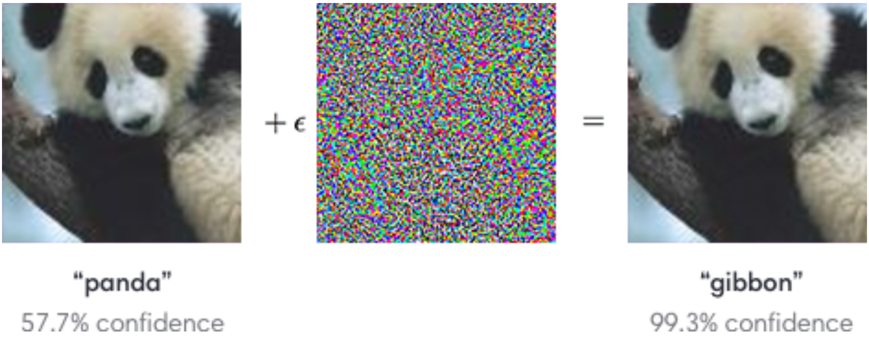
1. What are Adversarial Examples?
2. Why Adversarial Examples Exists?
3. How to Craft Adversarial Examples
4. Adversarial Examples and Transferability Attack
5. Adversarial Examples Counter Measures

What are Adversarial Examples?

Adversarial

It is an example that is carefully computed to be misclassified.

- It is indistiinguishable to human observer from the original image.



Evasion Attack

Evasion Attack

In evasion attacks, attackers deliberately manipulate the feastures within the input data during the inference stage to shift the result of a predictive model. Notethat: Nothing wrong with the model training

Evasion Example: A typical example is to change some pixels in an image to deceive object recognition system.[1].

T-shirt with adversarial pattern capable of fooling an object detector.

[1] `https://github.com/advboxes/AdvBox/blob/master/applications/StealthTshirt/README.md`

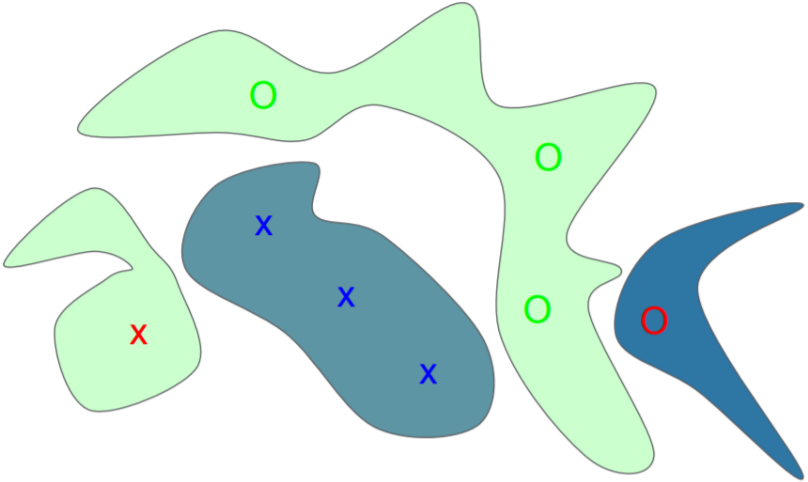
Notes on Adversarial Examples

- Applies to most machine learning models.
  - kNN, SVM, DT, LR
- In litreature, new attack techniques have been added to the literature, each with their own nuances, trade-offs, and quirks.
- But all adversarial examples share a fundamental conceptual basis: using knowledge of the model's internal state to find a small modification of input pixels that will lead to a model having the largest chance of error.

Outline

Why Adversarial Examples Exists ?

Can it be an overfitting problem?



Why Adversarial Examples Exists?

Can it be an overfitting problem?

- If it was an overfitting problem each adversarial example would have unique and is a result of some randomness. However, researchers found:

Why Adversarial Examples Exists?

Can it be an overfitting problem?

- If it was an overfitting problem each adversarial example would have unique and is a result of some randomness. However, researchers found:
    1. The same adversarial example is misclassified in the same way by different models.

Why Adversarial Examples Exists?

Can it be an overfitting problem?

- If it was an overfitting problem each adversarial example would have unique and is a result of some randomness. However, researchers found:
    1. The same adversarial example is misclassified in the same way by different models.
    2. The difference between the original example and the adversarial example is direction vector. Adding this vector to any other clean example we would get another adversarial example. (It is more like a Subspace)

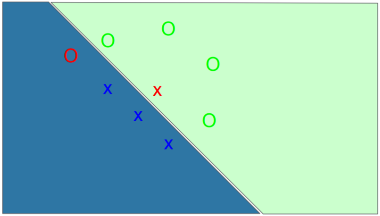
Why Adversarial Examples Exists?

Can it be an overfitting problem?

- If it was an overfitting problem each adversarial example would have unique and is a result of some randomness. However, researchers found:
  1. The same adversarial example is misclassified in the same way by different models.
  2. The difference between the original example and the adversarial example is direction vector. Adding this vector to any other clean example we would get another adversarial example. (It is more like a Subspace)
- The current researcher consensus is that adversarial examples are not a result of overfitting

Adversarial Examples from Excessive Linearity

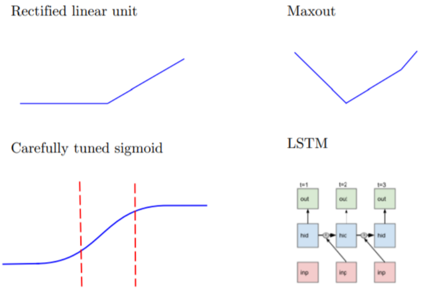
- Neural networks are discrimiative models they learn the decesion boundry between the class not the true structure of the data
- Moreover, as you move very far from the decesion boundry we are very confident of our decesion!
- The goal of the attacker is to find the direction that we could add or subtract to a clean image to get an adversarial example

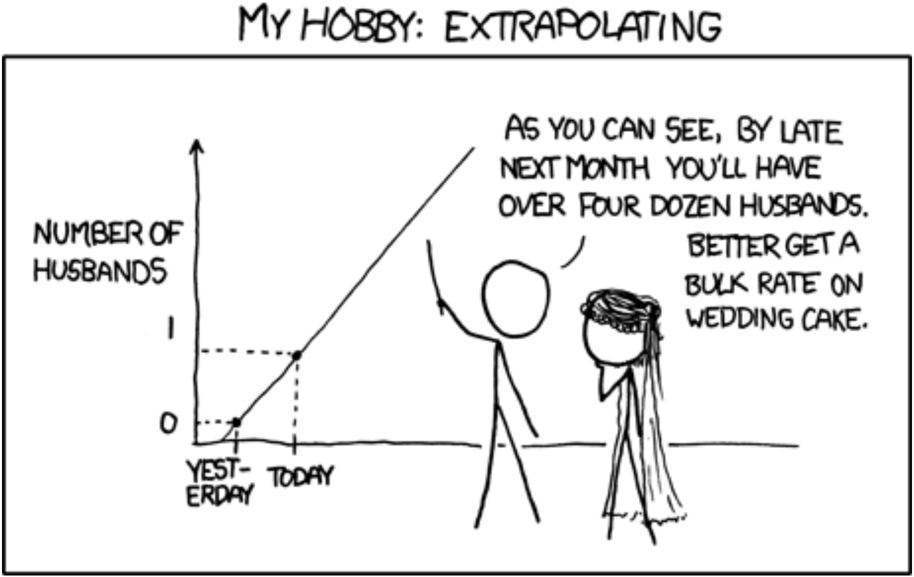


(Goodfellow 2016)

Piecewise linearity

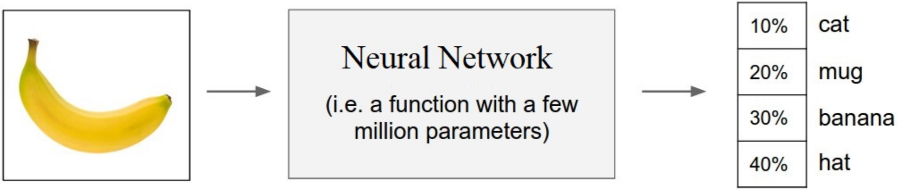
- Deep Learning models use piecewise linear functions to build up the architecture.
- This may inroduce some sort of underfitting where the model can not generalize to unseen inputs.
-
- Mapping from the input to the output is piecewise linear
- Even sigmoid and tanh fuction can be seen as piecewise linear.



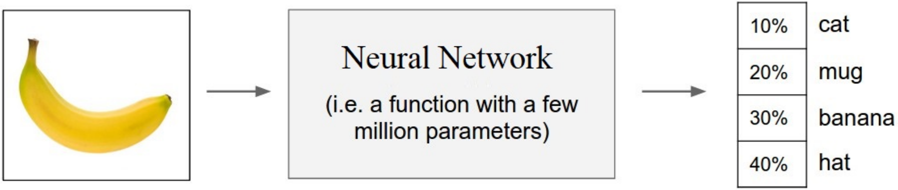

MY HOBBY: EXTRAPOLATING
NUMBER OF HUSBANDS
1
0
YEST-ERDAY
TODAY
AS YOU CAN SEE, BY LATE NEXT MONTH YOU'LL HAVE OVER FOUR DOZEN HUSBANDS.
BETTER GET A BULK RATE ON WEDDING CAKE.

Outline

How the Fooling Methods Work? (1/2)

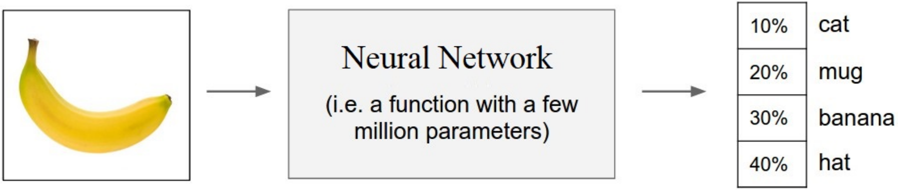


How the Fooling Methods Work? (1/2)



Normal neural network training: "What happens to the score of the correct class when I wiggle the network parameter?"

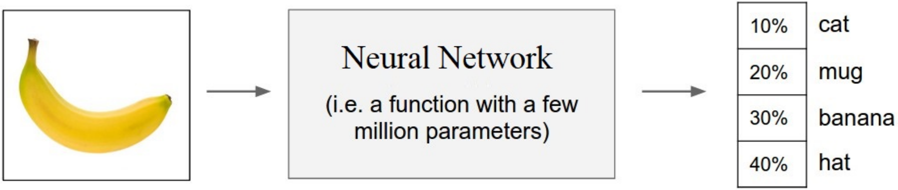
How the Fooling Methods Work? (1/2)



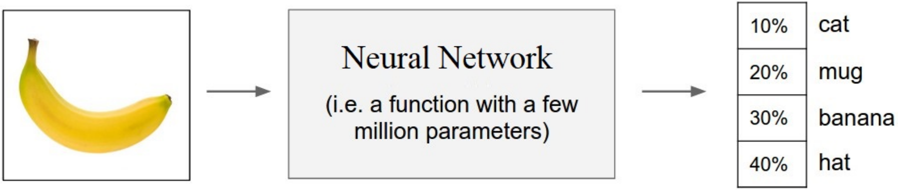
Normal neural network training: "What happens to the score of the correct class when I wiggle the network parameter?"

- Given loss defined as $L(\theta, x, y)$
- We find the gradient of the loss with respect to θ to tune the weights and minimize the loss.

How the Fooling Methods Work? (2/2)

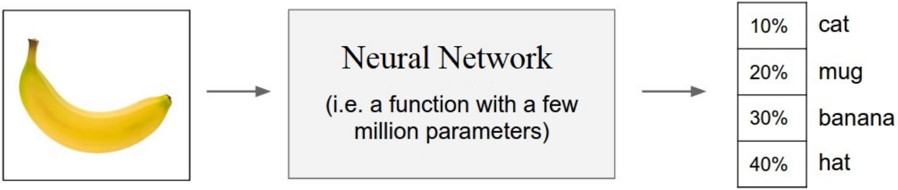


How the Fooling Methods Work? (2/2)



Normal neural network training: "What happens to the score of the correct class when I wiggle the input image?"

How the Fooling Methods Work? (2/2)



Normal neural network training: “What happens to the score of the correct class when I wiggle the input image?”

- Given loss defined as $L(\theta, x, y)$
- We find the gradient of the loss with respect to x to tune the input and maximize the loss.

Fast Gradient Sign Method

$$x_{adv} = x + \epsilon \text{ sign } (\nabla_x \text{ L}(\theta, x, y))$$

- x_{adv}: Adversarial image.
- x : Original input image.
- y : Original input label.
- ϵ : Multiplier to ensure the perturbations are small.
- θ : Model parameters.
- L: Loss.

Fast Gradient Sign Method

$$x_{adv} = x + \epsilon \, \mathrm{sign} \left(\nabla_x \, \mathrm{L}(\theta, x, y) \right)$$

- x_{adv}: Adversarial image.
- x : Original input image.
- y : Original input label.
- ϵ : Multiplier to ensure the perturbations are small.
- θ : Model parameters.
- L: Loss.

Notes

- The gradients are taken with respect to the input image.

Fast Gradient Sign Method

$$x_{adv} = x + \epsilon \ \mathrm{sign}\ (\nabla_x \ \mathrm{L}(\theta, x, y))$$

- x_{adv}: Adversarial image.
- x : Original input image.
- y : Original input label.
- ϵ : Multiplier to ensure the perturbations are small.
- θ : Model parameters.
- L: Loss.

Notes

- The gradients are taken with respect to the input image.
- This is done because the objective is to create an image that maximises the loss.

Fast Gradient Sign Method

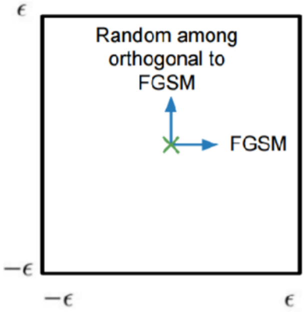
$$x_{adv} = x + \epsilon \text{ sign } (\nabla_x \text{ L}(\theta, x, y))$$

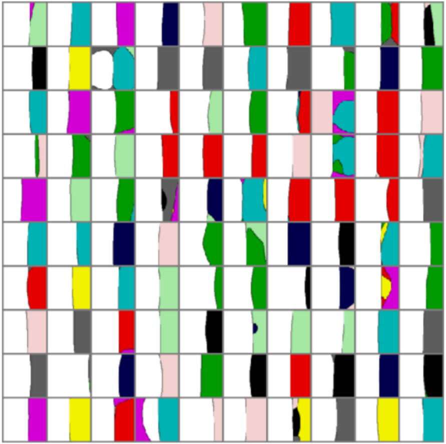
- x_{adv}: Adversarial image.
- x : Original input image.
- y : Original input label.
- ϵ : Multiplier to ensure the perturbations are small.
- θ : Model parameters.
- L: Loss.

Notes

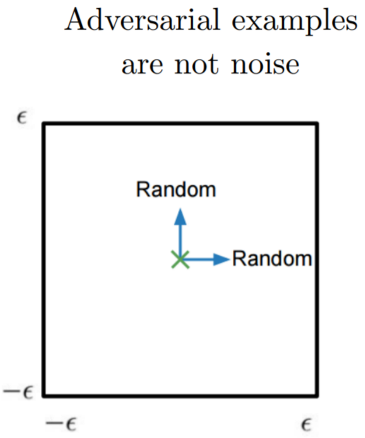
- The gradients are taken with respect to the input image.
- This is done because the objective is to create an image that maximises the loss.
- We are finding how much each pixel in the image contributes to the loss value

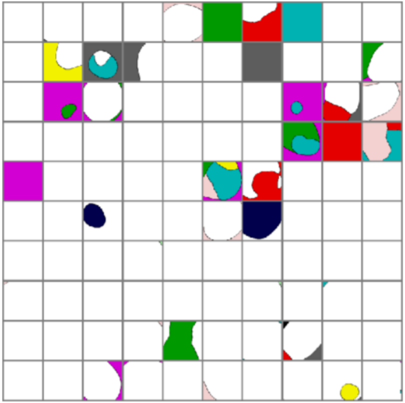
Adversarial Subspace





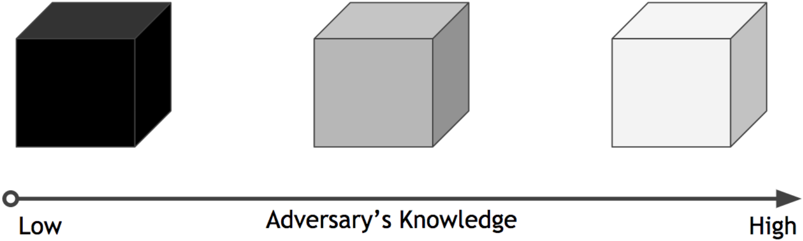
Maps of Random Subspace



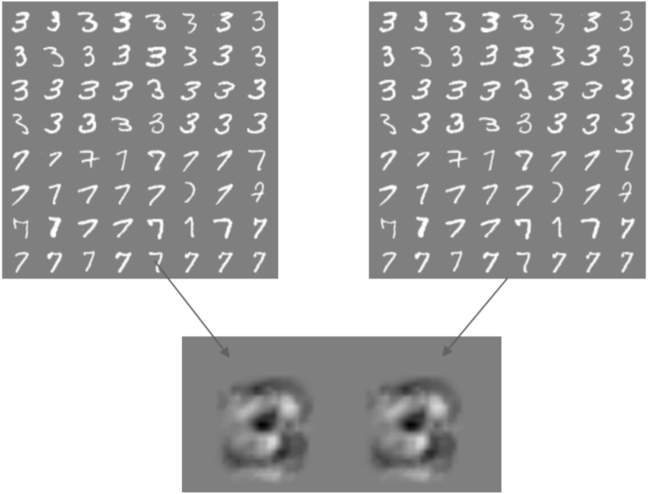


Fast Gradient Sign Method (Revisted)

$$x_{adv} = x + \epsilon \, \nabla_x \, \text{sign} \, (\text{L}(\theta, x, y))$$

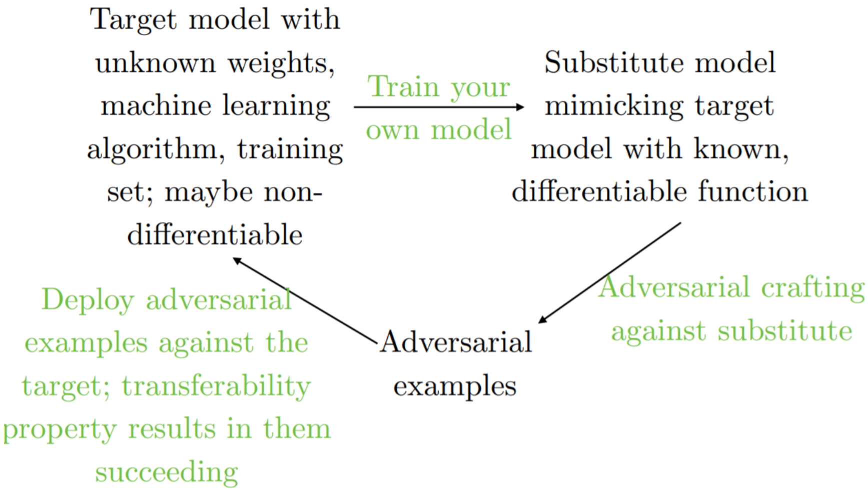


Cross-model, cross-dataset generalization

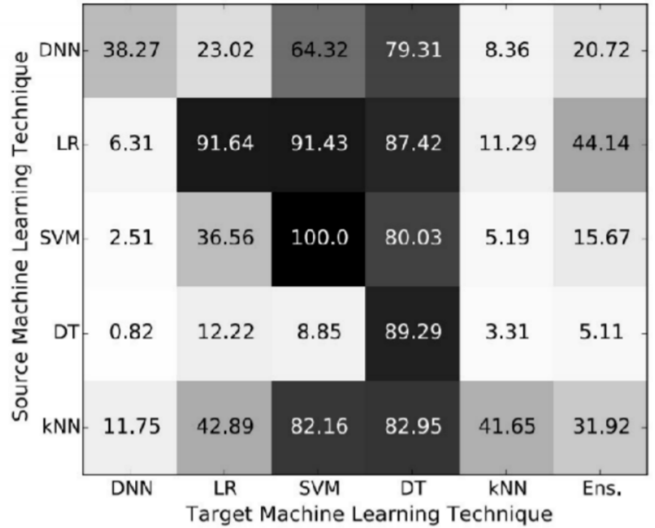


Outline

Transferability Attack (1/2)



Transferability Attack (2/2)

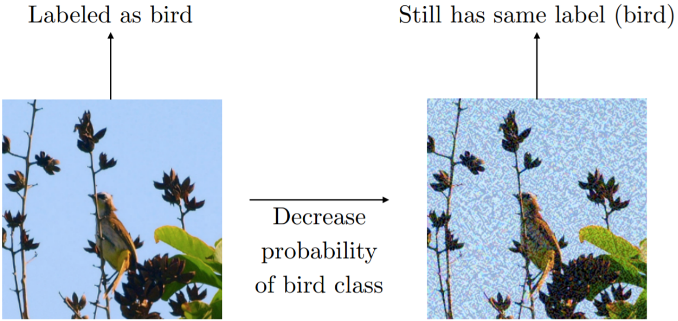


Outline

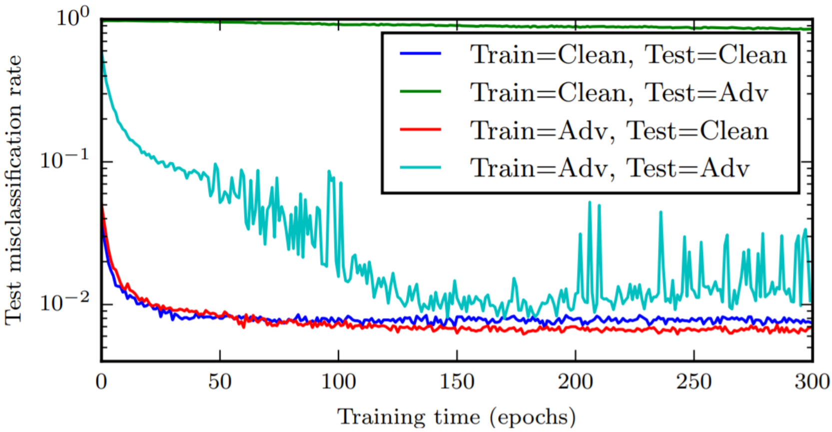
Adversarial Training (1/2)

Adversarial Training

Adversarial training, in which a network is trained on adversarial examples, is one of the few defenses against adversarial attacks that withstands strong attacks.



Adversarial Training (2/2)

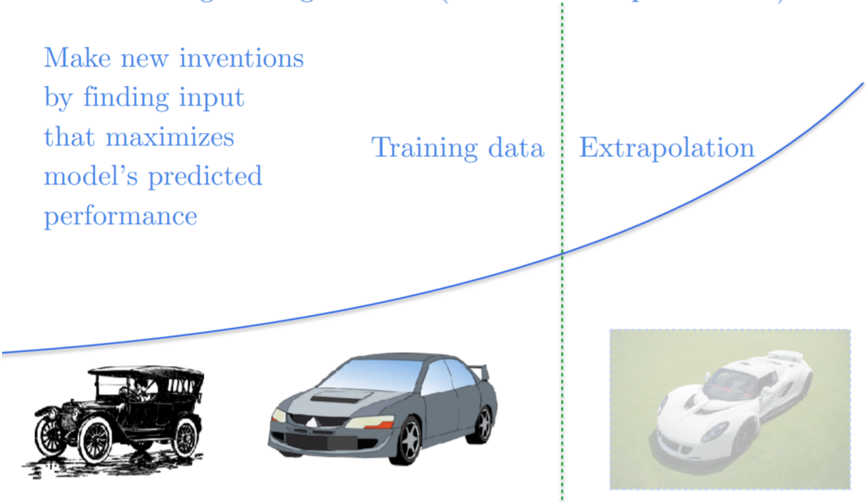


Adversarial training provides regularization

Universal Engineering Machine

Universal engineering machine (model-based optimization)

Make new inventions
by finding input
that maximizes
model's predicted
performance



Pytorch and FGSM

https://pytorch.org/tutorials/beginner/fgsm_tutorial.html

Questions