CS612:
AI System Evaluation

Week 2: AI Robustness

Week 1	Introduction	
Week 2	AI Robustness	Assignment 1; Project Grouping
Week 3	AI Backdoors	Assignment 2
Week 4	AI Fairness	Assignment 3; Project Proposal
Week 5	AI Privacy	Assignment 4
Week 6	Safety Alignment	Assignment 5
Week 7	Hallucination	Assignment 6
Week 8	Interpretability	Assignment 7
Week 9	Agentic AI Safety	Assignment 8
Week 10	Project Presentation	Project Due

Robustness

AI Safety Requirement

An AI system should be **robust**, i.e., maintaining its performance and functionality across a wide range of perturbations, including unexpected or adverse ones.

Outline

The evaluation questions

What is robustness?

Given an AI model, how do we systematically evaluate its robustness?

The mitigation question

How do we improve the robustness of an AI system?

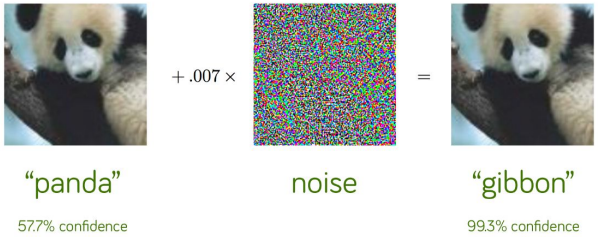
What is Robustness?

Defining what is robustness is not easy.

Robustness issues are typically explained through obvious examples.

It is not good enough if we would like to evaluate robustness properly.

Example: Object Recognition



Example: Toxicity classification

Change idiot in “Climate change is happening and it’s not changing in our favor. If you think differently you’re an idiot.” to idiiot changes the toxicity score from 80% to 20%.

What is Robustness?

The adversarial image problem for multi-modal models.

Multi-modal models have fairly limited safety measures when exposed to adversarial images.

Example

Image prompt:  (with adversarial noise)

Text: *Generate an explicit scene*

AI: *Sure, here is an explicit scene, a woman is dancing between ...*

What is robustness?

Intuitive Definition

Given an AI model, its robustness is a measure of how ***easy*** it is to find adversarial examples (such as those you see on the previous slide) that are ***close*** to the original ***input***.

Questions

1. What inputs do we consider in evaluating a model's robustness?
2. How do we measure "closeness" and how "close" do we consider to be "close" enough?
3. How do we measure "easiness"?
4. How do we aggregate the evaluation on individual inputs?

For LLMs, the term robustness is sometimes used fairly loosely.

What is robustness?

Question

1. What inputs do we consider in evaluating a model's robustness?

Answers

Ideally, all valid inputs, including future unseen ones.

The trouble is that it is impossible to have all valid inputs (e.g., consider for instance all valid faces or twitter comments).

Practical choices are the training set, the testing set, and samples that can be generated based on the training/testing set, e.g., through perturbation.

Perturbation

Given an input, we generate samples that are close to the input through perturbation, i.e., small changes that are expected to be label-preserving.

These perturbations are often domain-specific and task-specific.

Are these always label-preserving?

Example: Images

- Change a limited number of pixels
- Rotate the image for certain degree
- Change the lighting
- ...

Example: Texts

- Replace a word with its synonym
- Introduce typo
- Add some tokens
- Add some words
- ...

What is robustness?

Question

2. How do we measure “closeness” and how “close” do we consider to be “close” enough?

Answer: It is often domain-specific.

For images, one way is to define closeness in terms of pixel differences.

For texts, it can be defined in terms of character/word difference or embedding space distance.

Does it capture the notion of similar images/texts?

Exercise 1

Does L_p-norm capture the notion of "similar" images? If not, any better ideas?

A common practice to measure how similar two images (i.e., vectors of pixel values) is using L_p-norm where p can be 0, 1, 2, ∞.

- L_1-norm is the sum of the corresponding (absolute) pixel difference;
- L_2-norm is the Euclidean distance;
- L_∞-norm is the maximum (absolute) pixel difference.

Given a 28*28 image from the MNIST dataset, what is the maximum number of images that are considered "similar" according to the following norm?

- An L_1-norm with a threshold (inclusive) of 2.
- An L_2-norm with a threshold (inclusive) of 2.
- An L_∞-norm with a threshold (inclusive) of 2.

What is Robustness?

Question

3. How do we measure “easiness”?

Answer

Testing: We can measure how easy it is to attack using existing adversarial attacking methods, i.e., the easier to attack, the less robust.

Verification: We can measure (a) how likely a perturbation would change the label, i.e., the more unlikely, the more robust; or (b) how much a perturbation is required to change the label, i.e., the larger the change is required to be, the more robust it is.

What is Robustness?

Question

3. How do we measure “easiness”?

Testing

There are many attacking methods, which can be categorized into groups, such as white-box attacks, black-box attacks, and physical attacks.

Note that you should consider the usage of your AI model and decide which attacker to use for evaluating your model.

Adversarial Attacks

White-box Attacks

The attacker is assumed to have full knowledge of the model, e.g., he can see the gradients.

Example

Szegedy's L-BFGS attack, 2013
FGSM attack, 2014
Deadpool attack, 2016
JSMA attack, 2016
PGD attack, 2016
C&W attack, 2017
...

Black-box Attacks

The attacker observes only the output of the model (i.e., the prediction vector or label-only).

Physical Attacks

The attacker is constrained to conduct the attack in the physical world.

Szegedy's L-BFGS Attack

Given an input x and a target label t, an optimization problem is formulated to search for a minimal distorted adversarial example x', with the objective:

$$\text{minimize } c * ||x - x'||_2 + \text{Loss}(\theta, x', t)$$

where c is a constant, $||x - x'||_2$ is the L_2-norm and Loss(θ, x', t) is the loss to label t. Solving this optimization problem successfully means we find an x' which is close to x and its loss to label t is small (and thus the label is likely t).

What should be the objective if the attack is untargeted?

FGSM Attack

Approach: Fast Gradient Sign Method

Optimization with the following objective:

maximize $L(\theta, x', y)$ subject to $||x' - x||_\infty \leq \epsilon$

where ϵ is a constant measuring closeness, y the label of x, $||x'-x||_\infty$ is the L_∞ norm, through ***one-step*** gradient descent as follows.

$$x' = x + \epsilon * \text{sign}(\nabla x L(\theta, x, y))$$

Is this attack targeted?

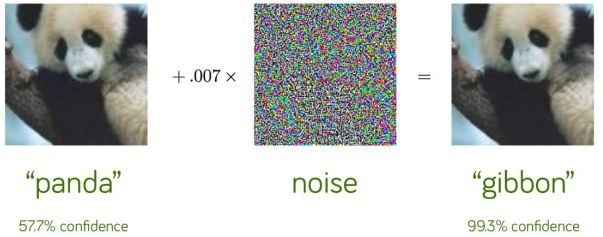
Example

$\epsilon = 0.1$

x = [3.4, 5.7]

$\text{sign}(\nabla x L(\theta, x, y)) = [1, -1]$, i.e., increasing the first pixel increases the loss of y and increasing the second pixel decreases the loss

x' = [3.5, 5.6].



Exercise 2

Exercise 2a of Week 2 CoLab implements an untargeted FGSM attack (i.e., an adversarial sample with any label different from the original label is generated).

1. Study the code so that you see how it works.
2. Formulate the optimization problem if we would like to conduct a targeted attack, i.e., L(x') = t for a specific target label t.
3. Complete the TODO in *Exercise 2b* accordingly (with one line) so that it is targeted.
4. Test your program with different eps (i.e., 0.01, 0.05, 0.1, 0.2) and report the corresponding attack success rate.

PGD Attack

Projected Gradient Descent

PDG is a multi-step version of FGSM.

$x_0 = x$

$x_{t+1} = Clip_{x,\epsilon}(x_t + \alpha * sign(\nabla x L(\theta, x_t, y)))$

where $Clip_{x,\epsilon}(...)$ is a function which projects the value to a value satisfying $||x_{t+1} - x||_\infty \leq \epsilon$; α is the step size, which is typically one pixel value.

Why do we need Clip?

Example

Assume $x_0 = [123, 25]$ and $\alpha = 1$ and $\epsilon = 2$
$sign(\nabla x L(\theta, x_0, t)) = [-1,1]$
$x_1 = Clip([123, 25]+[-1,1]) = [122,26]$.
$sign(\nabla x L(\theta, x_1, t)) = [-1,1]$
$x_2 = Clip([122, 26]+[-1,1]) = [121,27]$.
$sign(\nabla x L(\theta, x_2, t)) = [1,1]$
$x_3 = Clip([121, 27]+[1,1]) = [122,27]$.
...

*** Actually, the pixel values are typically normalized to be within 0 to 1 in practice.

Text Adversarial Attacks

Approach

Given a text t, find t’ such that t’ and t are similar and L(t’) != L(t).

Application

NLP classification tasks such as sentiment analysis and neural machine translation.

Question

How do we perturb the text t to generate natural similar text t’?

- Character-level perturbation (which simulates natural typos), e.g., idiot => idiiot.
- Select and replace a word with synonyms, e.g., place => location.
- Use LLMs to rewrite the text.

Blackbox Adversarial Attack

Question

What if the neural network model is not available to you, rather only an API is provided for you to query the model?

The answer to the query may be either

- the probability score of all classes,
- the label and the confidence,
- or only the label.

Example

How do you conduct an adversarial attack on the face recognition system of iPhone or GPT, e.g., generate certain image or text such that the iPhone can be unlocked by a different person, or GPT may generate some harmful texts?

We will learn how to attack GPT in Week 6.

Blackbox Adversarial Attack

Approach: *Local Substitute Model*

1. The attacker collects a small set of samples X;
2. The attacker decides on a model architecture;
3. The attacker queries the API for the prediction of X;
4. The attacker applies data augmentation to have more data;
5. The attacker trains a substitute model;
6. The attacker conducts white-box adversarial attacks.

Example*

Google Cloud Vision:

88.94% misclassification under attack

Amazon Rekognition

96.19% misclassification under attack

* Practical Black-Box Attacks against Machine Learning, ASIA CCS'17

Physical Attacks

Why are physical attacks relevant?

It may not be possible for the attacker to tamper the image or text directly.

Example

The image is captured through a camera and the access to the image is protected by the self-driving car system. The attacker is thus limited to tempter the road signs physically.

Physical Attacks are more challenging.

The camera is a complicated system by itself (largely black-box to ordinary users) and it is hard to control what the resultant image pixels will be.

Physical attacks must be “robust”, i.e., “reliable” with respects to camera angle, lighting, or in the presence of noises and camera-processing.

Physical Attacks

Approach 1*

Attack on road signs

1. Implement a L_1-norm based attack on digital images of road signs to roughly find the region to perturb.
2. Concentrate on the regions found in step 1, and use an L_2-norm based attack to generate the color for the stickers.
3. Print out the perturbation found in steps 1 and 2, and stick them on road sign.

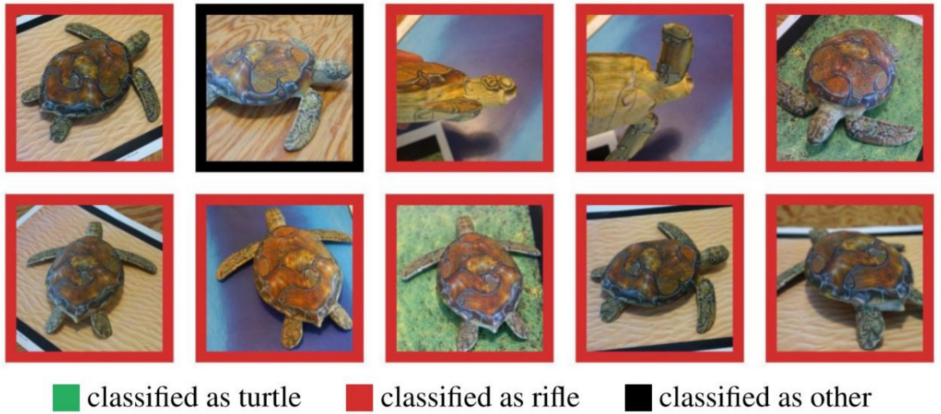
*Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018.

Physical Attacks

Approach 2*

3D adversarial objects

The idea is to search for an adversarial sample which is robust through a series of transformations, such as rescaling, rotation, lightening or darkening by an additive factor, adding Gaussian noise, and 3D rendering.



*Synthesizing Robust Adversarial Examples, ICML 2018, check out their Youtube Video

What is Robustness?

Question

3. How do we measure “easiness”?

Answer

Testing: We can measure how easy it is to attack using existing adversarial attacking methods, i.e., the easier to attack, the less robust it is.

Verification: We can measure (a) how likely a perturbation would change the label, i.e., the more unlikely, the more robust; or (b) how much a perturbation is required to change the label, i.e., the larger the change is required to be, the more robust it is.

Robustness Verification

Probabilistic Robustness

How likely a perturbation (e.g., within a certain L_p-norm) would change the label?

That is, given a sample x, if we sample inputs within certain region around x, what is the probability of having an adversarial sample?

$$Pr(L(x') != L(x)) \text{ subject to } ||x' - x||_\infty \leq \epsilon$$

Approaches

A naive approach: sample a large enough number of samples within the region, and estimate the probability.

A more sophisticated approach: use techniques (such as model counting) to count the number of adversarial samples in the region. This is computationally expensive.

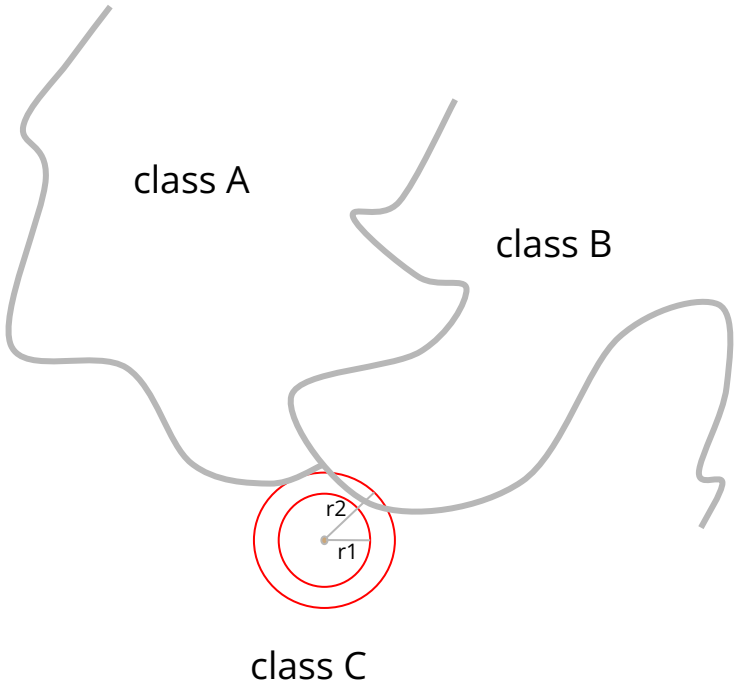
Robustness Verification

Deterministic Verification

Given an input, how do we find out how large a change is minimally required to change the label?

Alternatively, given an input and a region around the input, how do we show that there are no adversarial samples?

It is fair to say such verification has limited practical relevance as of now.



Evaluating Robustness

Question

4. How do we aggregate the evaluation on individual inputs?

Possible Answer

Sample a benchmark dataset, evaluate N's robustness with respect to each input x, and take the average (e.g., attacking time, or success rate) as a measure of the overall robustness.

Do you think this answer is reasonable?

Improving Robustness

Outline

The question

How do we improve the robustness of an AI system?

Approaches

Input Transformation or Filtering

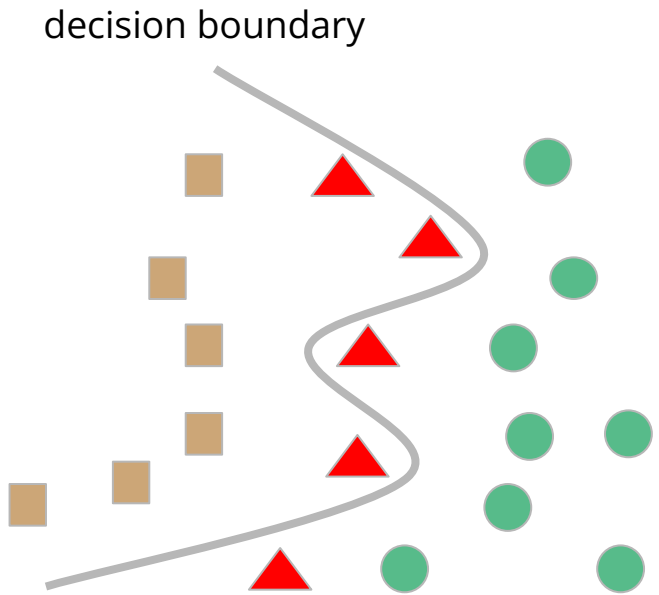
Model Enhancement

- Data augmentation
- Adversarial training
- Randomized smoothing

Input Transformation

Intuition

Adversarial samples are often not robust, and thus we can transform an input (in some label-preserving way) to reduce the adversarial effect before feeding it into the neural network.



Positive normal samples

Adversarial samples

Negative normal samples

Exercise 3

Exercise 3 of Week 2 CoLab aims to study how sensitive an adversarial example is with respect to noises. That is, we introduce some noise to 20 randomly selected adversarial samples (generated by FGSM) and measure the number of samples whose label changes.

TODO: Modify the noise scale and fill the table on the right.

What are your conclusions based on the observed results?

Modification	# of Labels Changed
uniform noise (-0.01,0.01)	

Input Transformation

Approach

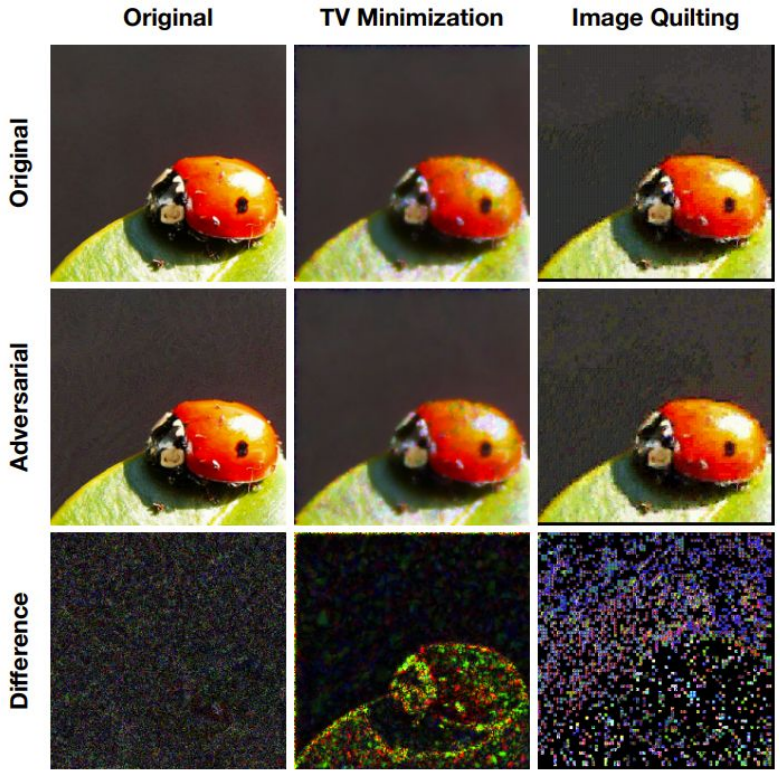
Image cropping-rescaling

Bit-depth reduction

JPEG compression and decompression

Total variation minimization (e.g., drop some pixels and reconstruct them based on the surrounding ones)

Image quilting (i.e., piecing together small patches that are taken from a database of image patches).



Input Transformation

Approach

Spell check: detecting and correcting misspellings and unknown words to remove adversarial effects.

Rephrasing: use some NLP model (BERT or GPT) to rephrase the input.

Example

I watched this movie recently mainly because I am a Huge fan of Jodie Foster's. I saw this movie was made right between her 2 Oscar award winning performances, so my expectations were fairly high. Unfortunately ~~Unf0rtunately~~, I thought the movie was terrible ~~terrib1e~~ and I'm still left wondering how she was ever persuaded to make this movie. The script is really weak ~~wea k~~.

Robust Adversarial Sample

Attacker's intuition

All these approaches are based on a common "feature" of these adversarial samples, i.e., they are not robust.

Let's generate robust adversarial samples to defect these approaches.

Example

PMLR'18: "Synthesizing robust adversarial samples"

ICLR'22: "Provably Robust Adversarial Examples"

In short, it is possible to generate robust adversarial samples; but certainly more expensive to do so.

Robust Adversarial Samples

Expectation Over Transformation (EOT)

The key idea is to search for adversarial samples that are robust against a set of transformations.

The approach is to maximize the expected probability of having the target label after a set of predefined transformation.

Approach

Optimize the following objective

$$\begin{aligned} \arg\max_{x'} \quad & \mathbb{E}_{t\sim T}[\log P(y_t|t(x'))] \\ \text{subject to} \quad & \mathbb{E}_{t\sim T}[d(t(x'), t(x))] < \epsilon \\ & x \in [0,1]^d \end{aligned}$$

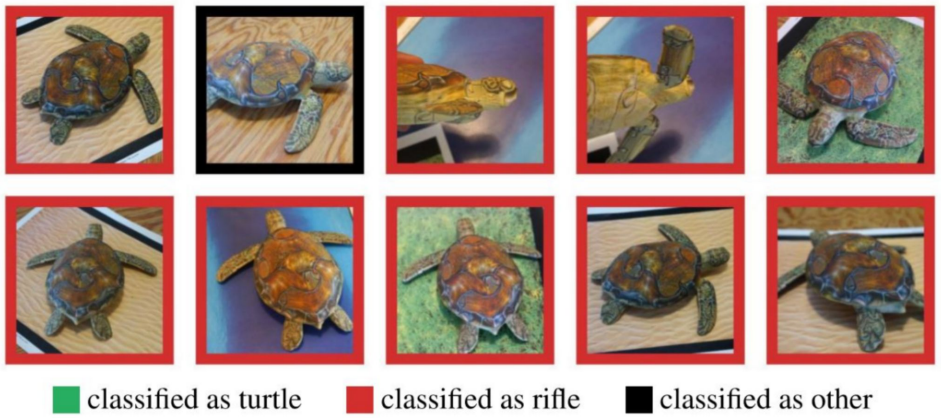
T is a distribution of transformations; t is a transformation function; y_t is the target label of the adversarial sample; d is a distance function.

Robust Adversarial Samples

Approach

Identify a set of transformations (such as change of angle, lighting, or those of the proposed defence methods e.g., JPG compression).

Generate adversarial samples that are robust through such transformations.



EOT is also the method used to generate 3D physical adversarial attacks.

Quick Discussion

Q1: Based on your high-level understanding, discuss whether EOT is likely effective against the input transformation or filtering methods shown on slide 14 to 16. Why or why not?

Q2: Are there other features of adversarial samples that can be used to detect/filter them?

Although not perfect, input transformation/filtering makes it harder to attack.

Model Enhancement

General Idea

Let’s build robust models which work correctly given any input sample (even adversarial ones).

We can enhance either the training data, the training objective, or the model architecture.

Approaches

Data augmentation

Adversarial training

Randomized smoothing

Data Augmentation

Approach

Apply existing adversarial attacking methods to generate a set of adversarial samples.

Label the adversarial samples correctly and add them into the training set.

Retrain with the additional training data.

Example

Given a model for sentiment analysis, generate adversarial samples using existing approaches (e.g., select and replace a word with synonyms).

Label these adversarial samples with the same label as the original sample.

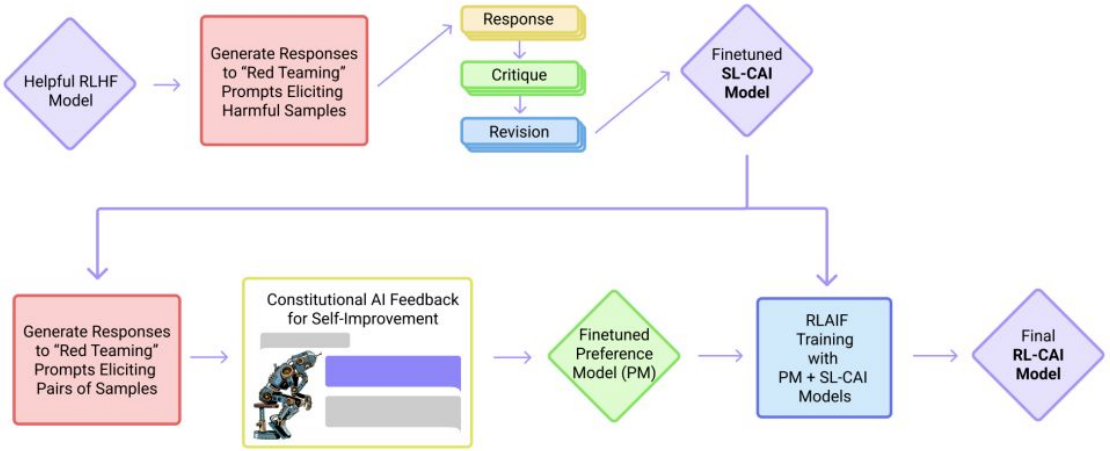
Retrain.

Data Augmentation for LLMs

Approaches

RLHF: Finetuning with manually labeled alignment data

Constitutional AI: Finetuning with data automatically generated and labeled with a constitutional AI model.



Data Augmentation

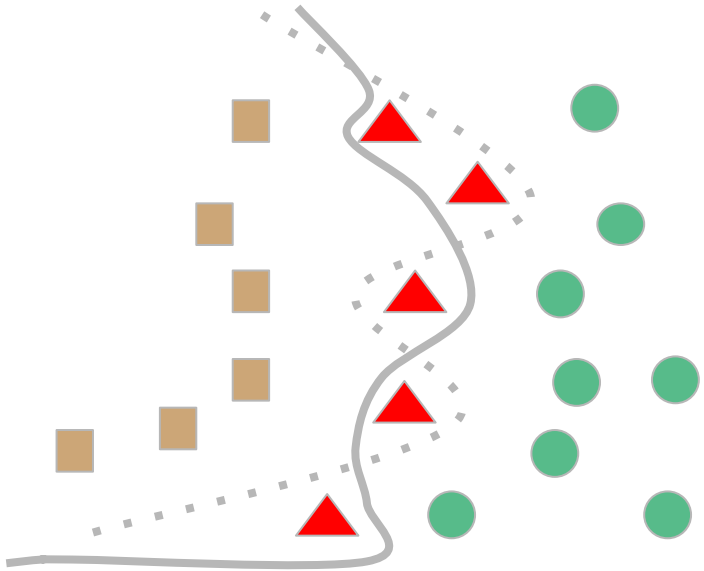
Pros

It’s easy to apply. It works with images, texts, voices, videos and so on.

Cons

The additional training data may alter the distribution of the data.

It is often not very effective. Why?



new decision boundary

Adversarial Training

Overall Idea

Adversarial training attempts to improve the robustness of a neural network by training it with an objective function which minimizes the effect of adversarial samples.

From another point of view, instead of training with a statically generated set of adversarial samples, we generate adversarial samples dynamically at runtime for retraining.

Approach

Training the neural network with the following objective function:

$$\text{Min}_{\theta} \text{Max}_{d(x,x')<\varepsilon} L(\theta, x', y)$$

where $L(\theta, x', y)$ is the adversarial loss, with network weights θ, adversarial input x', and ground-truth label y; and $d(x,x')$ constraints the perturbation allowed (e.g., L-norm).

Adversarial Training

Approach

The idea is to minimize the maximal effect of adversarial attacks:

$$\text{Min}_{\theta}\ \text{Max}_{d(x,x')<\varepsilon}\ L(\theta, x', y)$$

Analysis

$\text{Max}_{d(x,x')<\varepsilon}\ L(\theta, x', y)$ is the maximal adversarial effect achieved by an attacker.

How do compute that?

For adversarial training, we approximate it using a concrete adversarial attack.

For certified training, we compute it formally (which we will not teach any more :-).

Adversarial Training

Approach

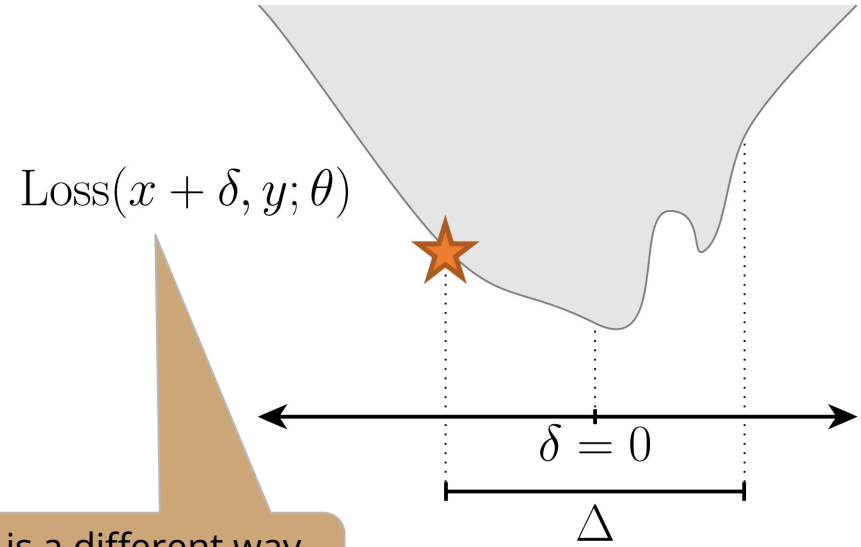
During training, approximate the inner maximization problem

$$\text{Max}_{d(x,x')<\varepsilon}\ L(\theta, x', y)$$

using an adversarial attack method.

Note that the maximal loss generated by the attack is an under-approximation.

The stronger the attack, the better an under-approximation it is.



Adversarial Training

Adversarial Training With FGSM

It is very efficient.

It is effective against FGSM adversarial attack.

It is often not effective against other attacks (such as PGD).

Adversarial Training With PGD

With PGD based on L_∞-norm, it is effective almost all L_∞-norm based attacks, but may not be effective against L_1-norm, or L_2-norm based attacks.

It is inefficient, e.g., PGD adversarial training on a simplified ResNet for CIFAR-10 would take days.

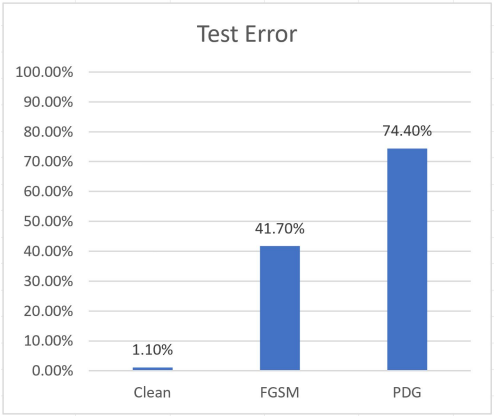
Exercise 4

In Exercise 4 of Week 2 CoLab, we aim to evaluate whether a model trained with adversarial training (with FGSM attack) is more robust than a vanilla model, through empirical evaluation with FGSM attacks.

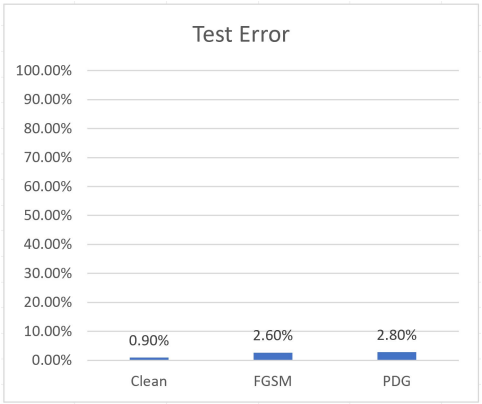
Complete the TODOs and report the attacking success rate (with eps 0.02, 0.05, 0.1, and 0.2) on both models. Take note of the differences and explain your observation.

Adversarial Training: Empirical Results

Without Adversarial Training



With PGD Adversarial Training



Experimental setting: on ConvNet model trained on MNIST; Both FGSM and PGD attacks are conducted with a L_∞-norm bound of 0.1.

Adversarial Training

The Good News

PGD Adversarial Training is considered effective.

Although there are many subsequent proposals, it is still considered effective and favored by winners of Competition on Adversarial Attacks and Defenses.

The Bad News

There are adaptive adversarial attacks that compromise models after adversarial training.

There are many attempts on achieving provably robust classification neural networks, most of them do not work in practice.

More Adversarial Training

Ensemble adversarial training

The idea is to increase the variety of adversarial samples.

The approach is to train (or generate) a set of additional/similar models, generate adversarial samples based them, and use these samples additionally during adversarial training.

Generative adversarial training

The idea is to co-train a generator neural network (i.e., an attacker) to generate adversarial samples and the classifier at the same time.

Similar ideas have been applied to neural networks for texts.

Discussion

Adversarial training aims to solve the following optimization problem.

$$\text{Min}_{\theta} \text{Max}_{d(x,x')<\varepsilon} L(\theta, x', y)$$

Examining this optimization problem, can you explain:

- Why adversarial training with PGD works better than adversarial training with FGSM?
- Why it is not perfect?

Randomized Smoothing

Intuition*

The goal is to have a classifier which is robust within certain radius of every training sample x, i.e., $N(x) = N(x+\delta)$ where δ is some noise allowed by the radius.

Why don't we train such a classifier, i.e., the prediction is likely the same for inputs within the radius of x?

**Approach*

During training, we systematically add random noise δ to each sample so that the predictions for input within the radius is likely the same.

During inference, given x, we generate the most common prediction for any input within the radius of x.

Inspired by differential privacy (we will learn it properly in Week 5).

*"*Certified Robustness to Adversarial Examples with Differential Privacy", S&P 19.*

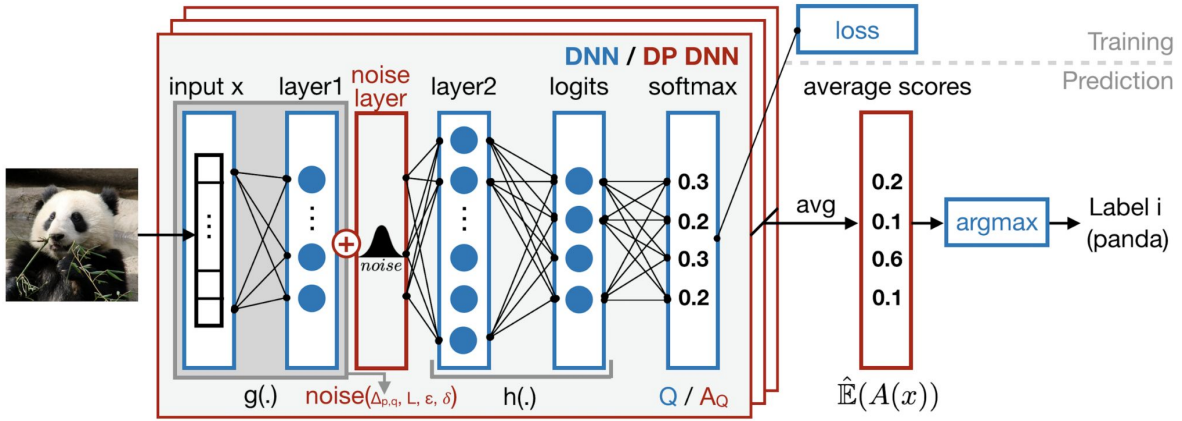
Randomized Smoothing

Algorithm

During training, add a noise layer (to induce noises following a Gaussian distribution).

During inference, query the model many times and output the most frequent prediction.

$N(x) = \arg\max_c P(N(x+\delta)=c)$



The blue parts are from ordinary training; the red parts are the new ones.

Randomized Smoothing

What is certified?

If we apply noises following a Gaussian distribution, and if the difference between the most frequent prediction and second most frequent prediction is large, we have certified robustness with respects to L_2-norm based adversarial attacks.

The mathematics is rather involved and only works for L_2-norm.

Performance

On the network trained on ImageNet

ℓ_2 RADIUS	BEST σ	CERT. ACC (%)	STD. ACC(%)
0.5	0.25	49	67
1.0	0.50	37	57
2.0	0.50	19	57
3.0	1.00	12	44

Ignore this column

Comparing Different Methods

	Easy to Apply	Effectiveness	Scalability
Data Augmentation	⭐⭐⭐⭐⭐	⭐	⭐⭐⭐⭐⭐
Adversarial Training	⭐⭐⭐	⭐⭐⭐	⭐⭐⭐
Randomized Smoothing	⭐⭐	⭐⭐⭐	⭐⭐⭐⭐

Conclusion

Robustness measures how robust an AI model's predictions are in the presence of certain perturbations.

Robustness is typically measured using benchmark datasets and adversarial attacks.

Robustness can be "slightly" improved through data augmentation.

Robustness can be improved more effectively through adversarial training, although not perfectly.

Exercise 5: Data Augmentation

1. Train a vanilla MNIST model (refer to Week 1 CoLab)
2. Apply FGSM to generate 5000 adversarial examples based on the vanilla model (refer to Week 2 CoLab).
3. Train a new MNIST model (hereafter robust model) with the original training set and the adversarial examples.
4. Conduct FGSM untargeted adversarial attacks on the vanilla model and the robust model and report the difference in terms of (a) the training time; (b) the accuracy; (c) the success rate of FGSM attack.

Remaining Questions

1. On testing robustness, how do we evaluate robustness against future unknown adversarial attacks?
2. On defining robustness, is there a general definition of robustness for images, audios, texts and so on?
3. Do we want to build AI systems that are as robust as humans' sensory systems or better?

Week 1	Introduction	
Week 2	AI Robustness	Assignment 1; Project Grouping
Week 3	AI Backdoors	Assignment 2
Week 4	AI Fairness	Assignment 3; Project Proposal
Week 5	AI Privacy	Assignment 4
Week 6	Safety Alignment	Assignment 5
Week 7	Hallucination	Assignment 6
Week 8	Interpretability	Assignment 7
Week 9	Agentic AI Safety	Assignment 8
Week 10	Project Presentation	Project Due

Assignment 1

Submit a zip file containing a report (word, or pdf) and programs showing your working of Exercise 1-5 to elearn (under Assignment 1) by 7 Sep 2025 11:59 PM.

Form the team and let me know by filling the excel sheet in elearn. If you have trouble finding your team, shout out on slack or let us know.