



AI System Evaluation

Week 2: AI Robustness



Aug 23 - Week 1: 7-10	Introduction	
Aug 30 - Week 2: 7-10	AI Robustness	Exercise 1
Sep 06 - Week 3: 7-10	Improving AI Robustness	Exercise 2
Sep 13 - Week 4: 7-10	AI Backdoors	Exercise 3
Sep 20 - Week 5: 7-10	Mitigating AI Backdoors	Exercise 4; Project Proposal
Sep 27 - Week 6: 7-10	AI Fairness	Exercise 5
Oct 11 - Week 7: 7-10	Improving AI Fairness	Exercise 6
Oct 18 - Week 8: 7-10	AI Privacy	Exercise 7
Oct 25 - Week 9: 7-10	Improving AI Privacy	Exercise 8
Nov 01 - Week 10: 7-10	AI Interpretability	Project Due
Nov 08 - Week 11: 1-3	End-of-Term Exam	

Recall: Robustness



Outline

The question

Given a deep learning model, how do we systematically evaluate its robustness?

To answer the question, we will discuss:

What is robustness?

Under what conditions do we evaluate the robustness, e.g., against an all-knowing attacker or one with limited knowledge?

What are the existing methods for evaluating robustness?

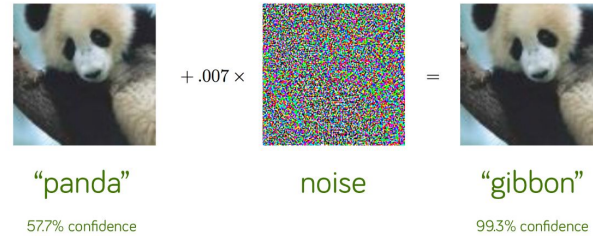
What is Robustness?

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

What is robustness?

Intuitive Definition

Given a neural network, 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 their 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?

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 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 word
- ...

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, closeness can be defined in terms of pixel differences, e.g., L_1 -norm, L_2 -norm, and L_∞ -norm.

Given two images (i.e., vectors of pixel values), L_1 -norm is the sum of the corresponding (absolute) pixel difference; L_2 -norm is the Euclidean distance; and L_∞ -norm is the maximum (absolute) pixel difference.

An input is considered “close” if the norm is within certain threshold.

Exercise 0

Given a 28×28 image from the MNIST dataset, what is the maximum number of images that are considered to be close 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.

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

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 text, closeness can be defined in terms of the number of character differences or word differences. The latter can be further defined based on Euclidean distance in the embedding space.

An input is considered “close” if the difference is within certain threshold.

Does it capture the notion of similar texts? If not, any better ideas?

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.

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 $\text{Loss}(\theta, x', t)$ 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).

Does it work with other kinds of norms?

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.

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

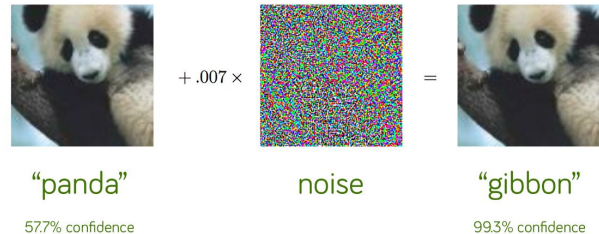
Is x' guaranteed to be within the norm?

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 1

The implementation in `week2/exercise1/fgsmUntargeted.py` (from [this repository](#)) is a un-targeted attack (i.e., an adversarial sample with a label different from the original label is generated) based on the FGSM method.

1. Study the code of `fgsmUntargeted.py` 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 `week2/exercise1/fgsmTargeted.py` accordingly (with one line) so that it is targeted.
4. Test your program with different eps.

PGD Attack

Approach: Projected Gradient Descent

PDG is a multi-step version of FGSM.

$$x_0 = x$$

$$x_{t+1} = \text{Clip}_{x,\epsilon}(x_t + \alpha * \text{sign}(\nabla_x L(\theta, x_t, y)))$$

where $\text{Clip}_{x,\epsilon}(\dots)$ 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 $\epsilon = 2$

$$\text{sign}(\nabla_x L(\theta, x_0, t)) = [-1, 1]$$

$$x_1 = \text{Clip}([123, 25] + [-1, 1]) = [122, 26].$$

$$\text{sign}(\nabla_x L(\theta, x_1, t)) = [-1, 1]$$

$$x_2 = \text{Clip}([122, 26] + [-1, 1]) = [121, 27].$$

$$\text{sign}(\nabla_x L(\theta, x_2, t)) = [1, 1]$$

$$x_3 = \text{Clip}([121, 27] + [1, 1]) = [122, 27].$$

...

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

C&W Attack

Background

In 2014, the paper reporting the problem of adversarial perturbation was published.

Multiple papers claimed to successfully defend FGSM/PGD attack in 2015/2016.

C&W was proposed to overcome almost all of those defenses in 2017.

Approach: Carlini & Wagner Attack

The authors address the same optimization problem of L-BFGS attack by instead solving:

$$\text{minimize } \|x - x'\|_2 + c \cdot f(x', t)$$

Function $f(x', t)$ is the maximal difference $L(i) - L(t)$ between any label i and the target label t .

*** $L(i)$ is the prediction confidence of label i .

C&W Attack

Approach

The authors address the same optimization problem of L-BFGS attack by instead solving (based on ADAM optimization):

$$\text{minimize } \|x - x'\|_2 + c \cdot f(x', t)$$

Function $f(x',t)$ is the maximal difference $L(i)-L(t)$ between any label i and the target label t .

*** $L(i)$ is the prediction confidence of label i .

Example

Assume $L(x)=\text{bird}$ and t is dog.

Initially

$$L(x,\text{bird})=0.5, L(x,\text{cat})=0.3, L(x,\text{dog})=0.2$$
$$f(x,t) = 0.3$$

After one round of optimization

$$L(x',\text{bird}) = 0.4, L(x',\text{cat}) = 0.25, L(x',\text{dog})=0.35$$
$$f(x',t) = 0.05$$

Exercise 2

Execute `week2/exercise2/fgsm.py`, `pdg.py`, and `cw.py` (which attacks the 5 MNIST images in the folder `week2/exercise2/toattack/`).

Observe the difference in the attacking time, and the generated adversarial samples.

Summarize whether the attack is successful in the following table. Each entry should be of the form $n/5$ where n is the number of times the attack is successful.

eps	FGSM	PGD	C&W
0.01			
0.02			
0.05			
0.1			
0.2			

Text Adversarial Attacks

Approach

Given a text t , find t' such that t' and t are similar and $L(t') \neq L(t)$.

Application

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

Question

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

Any idea?

Text Adversarial Attacks

Approach 1*

Character-level perturbation (which simulates natural typos)

- swap (e.g., noise => nosie),
- random (e.g., noise => onise),
- and typo (e.g., noise => noize).

*Adversarial Examples for Natural Language Classification Problems, ICLR 2018.

Example: Toxicity classification

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

Are such attacks natural? Can you suggest a way to detect such attacks?

Text Adversarial Attacks

Approach 2*

Select and replace a word with synonyms (i.e., words that are close according to the embedding distance and are not out of context according to a Google language model), and maximize the probability of a target label.

*Adversarial Examples for Natural Language Classification Problems, ICLR 2018.

Can you suggest a way to detect such attacks?

Classifier: LSTM. **Original label:** 97% Negative. **New label:** 0% Negative.

Text: ~~this~~ **that** ~~place~~ **location** is far from the the best pho experience i 've ever had (that is almost a bad pun) . it 's really not bad , but there are much better vietnamese restaurants in vegas . the pho broth is n't on the same level as pho so 1 or lemongrass cafe . for some reason , they were out of bean ~~sprouts~~ **sprout** and while i do n't love them , i 've become accustomed to having them in my pho . finally , i was a little disappointed that they do n't serve tripe in any of their pho variations . overall , although i did enjoy the soup , i probably wo n't return . i need to try the ~~jenni~~ **jenny** pho place just down the street . if that does n't work out , i 'll just have to make the extra drive to chinatown .



Text Adversarial Attacks

Approach 3*

Use BERT to replace or insert words that fit better with the overall context.

*BAE: BERT-based Adversarial Examples for Text Classification, EMNLP 2020.

Example

ORIGINAL	The government made a quick decision
BAE - R 	The MASK made a quick decision judge , doctor , captain
BAE - I 	The MASK government made a quick decision state , british , federal
	The government MASK made a quick decision officials , then , immediately

Are such attacks natural? Are such attacks easy to detect?

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 Google Cloud Vision or Amazon Rekognition?

Try this yourself

<https://cloud.google.com/vision>

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 temper 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.

Exercise 3

Conduct the following experiment to understand the difficulty of physical attacks.

1. Apply `week2/exercise2/cw.py` to the 5 images in the folder `week2/exercise2/toattack/` to generate 5 adversarial images.
2. Take a photo of the 5 images from the screen;
3. Process the 5 photos so that it has the same format of the MNIST dataset;
4. Evaluate whether the 5 photos are adversarial samples.

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.



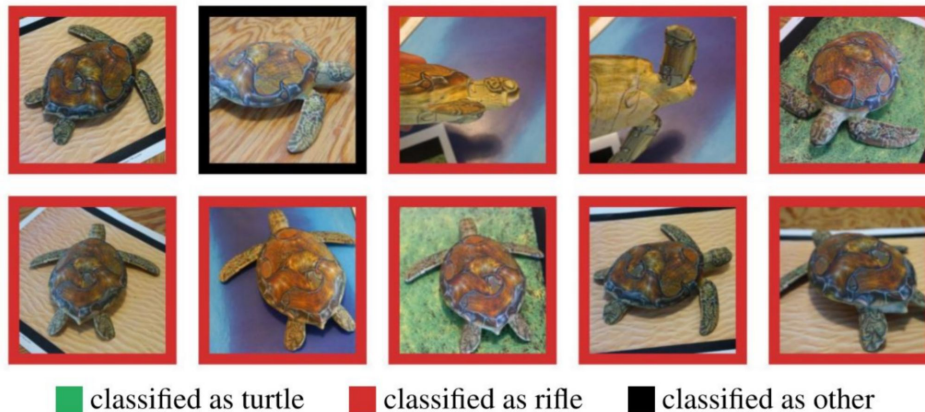
*Robust Physical-World Attacks on Deep Learning Visual Classification, CVPR 2018.

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.



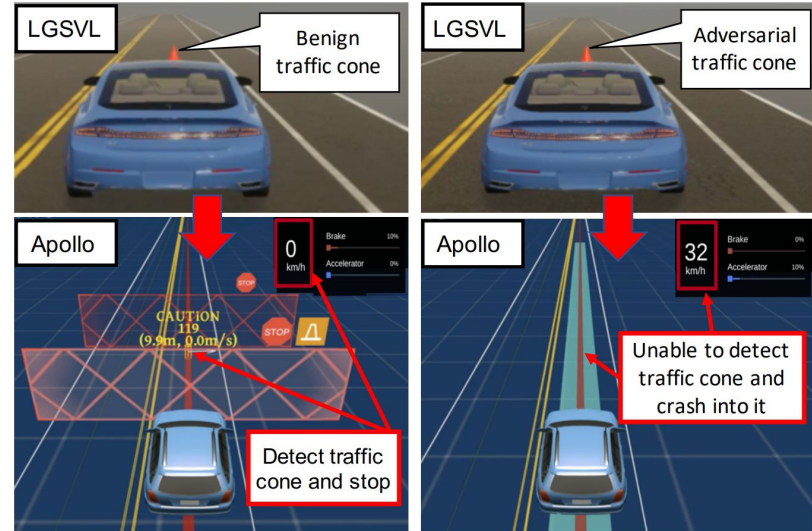
*Synthesizing Robust Adversarial Examples, ICML 2018, check out their [Youtube Video](#)

Physical Attacks

Approach 3*

The idea is to 3D-print certain objects such that they are invisible to both LIDAR + Camera.

*Invisible for both Camera and LiDAR: Security of Multi-Sensor Fusion based Perception in Autonomous Driving Under Physical-World Attacks, S&P 2021



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.

Robustness Verification

Probabilistic Robustness

How likely a perturbation (e.g., with 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?

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

Approaches

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

A more sophisticated approach: use model counting techniques to count the number of adversarial samples.

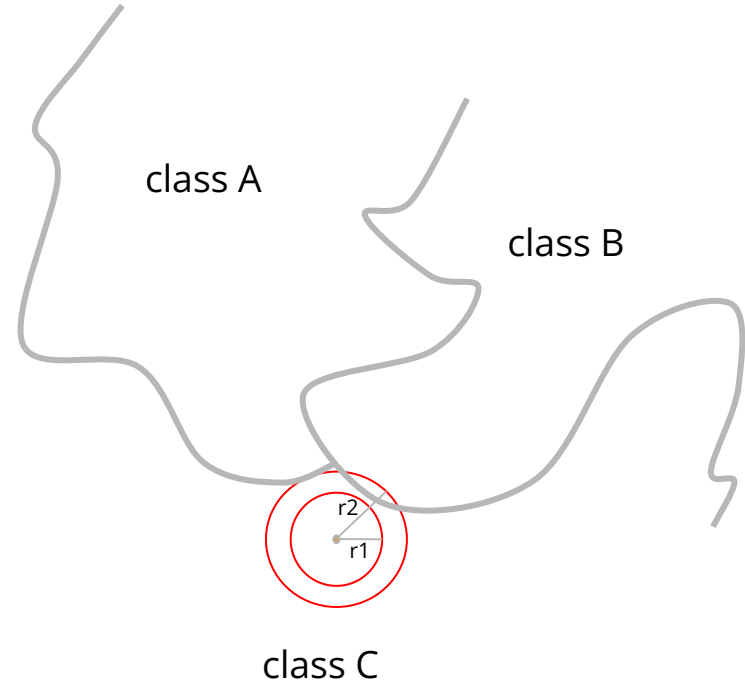
Even more sophisticated approach: See, e.g., “PROVEN: Verifying Robustness of Neural Networks with a Probabilistic Approach”, ICML 2019.

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 constant on the amount of “change”, how do we show that there is no adversarial attack?



Neural Network Verification

The problem of verifying robustness

Given an input x , show that $L(x+\Delta) = L$ for all Δ (e.g., any pixel different satisfying a given L_1 -norm, L_2 -norm, and L_∞ -norm threshold or any label-preserving transformation such as rotation up to **certain degree**).

Can we exhaustively try all samples within the norm?

Example

Assume a toy image with two pixels $[53, 125]$.

Assume that the attacker is only allowed to change the image with the L_∞ -norm of 1.

If we show that

$$\begin{aligned}L([53, 124]) &= L([54, 124]) = L([52, 124]) = \\L([53, 125]) &= L([54, 125]) = L([52, 125]) = \\L([53, 126]) &= L([54, 126]) = L([52, 126]),\end{aligned}$$

we show that there is no adversarial sample within a L_∞ -norm of 1 given input $[53, 125]$.

Neural Network Verification

The problem of verifying robustness

Given an input x , show that $L(x+\Delta) = L$ for all Δ (e.g., any pixel different satisfying a given L_1 -norm, L_2 -norm, and L_∞ -norm threshold or any label-preserving transformation such as rotation up to **certain degree**).

Approaches

Many approaches and tools have been developed.

- Reluplex
- Sherlock
- MIPVerify
- ReLUVal
- FastLin
- AI², DeepPoly, RefinePoly, PRIMA
- ...

Abstract Interpretation

The general abstract interpretation algorithm

1. Choose an abstraction domain.
2. Replace initial variable values using their abstract values.
3. Abstract each statement in the program using its abstract interpretation.
4. Analyze the program based on the abstracted program.

What if I abstract $y\%100$ as $[0,100]$?

Example

If we choose the abstraction domain to be the interval of the variables, e.g., variable k 's value is in the range of $[1,10]$.

```
func foo (int x, int y) { //x, y in [int.Min, int.Max]
    x = (x+1)%100; // x in [0,99]
    z=y%100 + x; // y%100 in [0,99], z in [0,198]
    assert(z<=200);
}
```

Ergo: `assert(z<=200)` is always satisfied.

Abstract Interpretation

Choosing abstraction domains

There are many abstraction domains.
Some are more precise than others.

More precise domains are powerful but more expensive to apply; less precise domains are less powerful but more efficient to apply.

Example

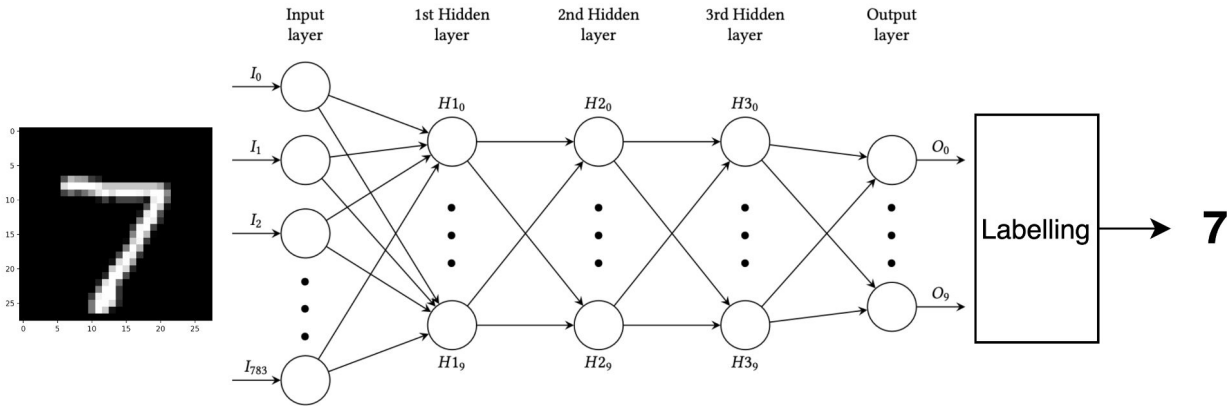
If we choose the abstraction domain to be whether a variable is non-negative, e.g., $k \geq 0$.

```
func foo (int x, int y) { //We have no idea.  
    x = (x+1)%100; // x>=0  
    z=y%100 + x; // y%100 >=0, z >= 0  
    assert(z<=200);  
}
```

We cannot conclude $z \leq 200$ is always satisfied.

DeepPoly

DeepPoly is one abstract interpretation method (and tool) for verifying neural networks. It focuses on feedforward neural networks, and CNN.



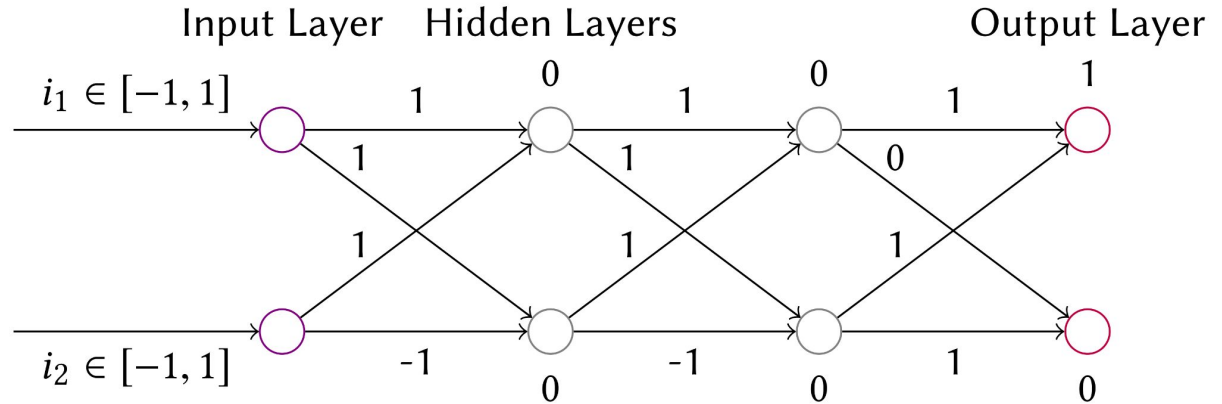
DeepPoly

Example

Assume an input $[0,0]$.

Assume the L_∞ -norm bound is 1, i.e., the two input i_1 and i_2 can be any real value before -1 and 1.

Show that the output is always 1.



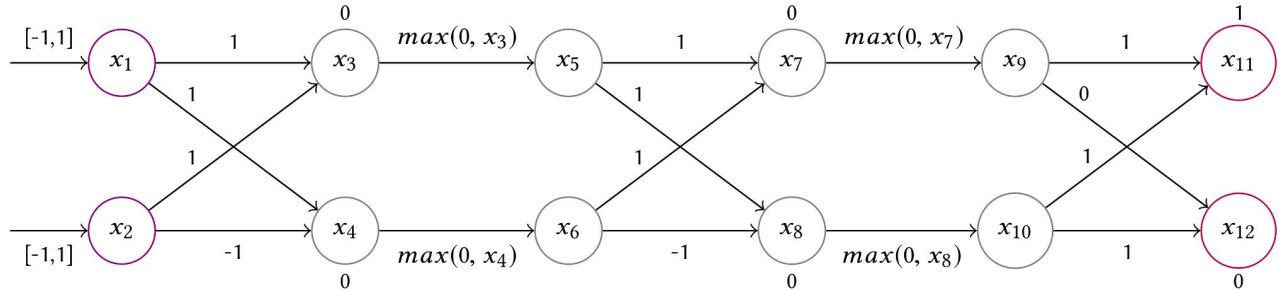
A simple neural network with 2-hidden layers, assuming the activation function is ReLU.

DeepPoly: Example

DeepPoly abstracts each neuron with a tuple.

$$\langle x_1 \geq -1, \quad \langle x_3 \geq x_1 + x_2, \\ x_1 \leq 1, \quad x_3 \leq x_1 + x_2, \\ l_1 = -1, \quad l_3 = -2, \\ u_1 = 1 \rangle \quad u_3 = 2 \rangle$$

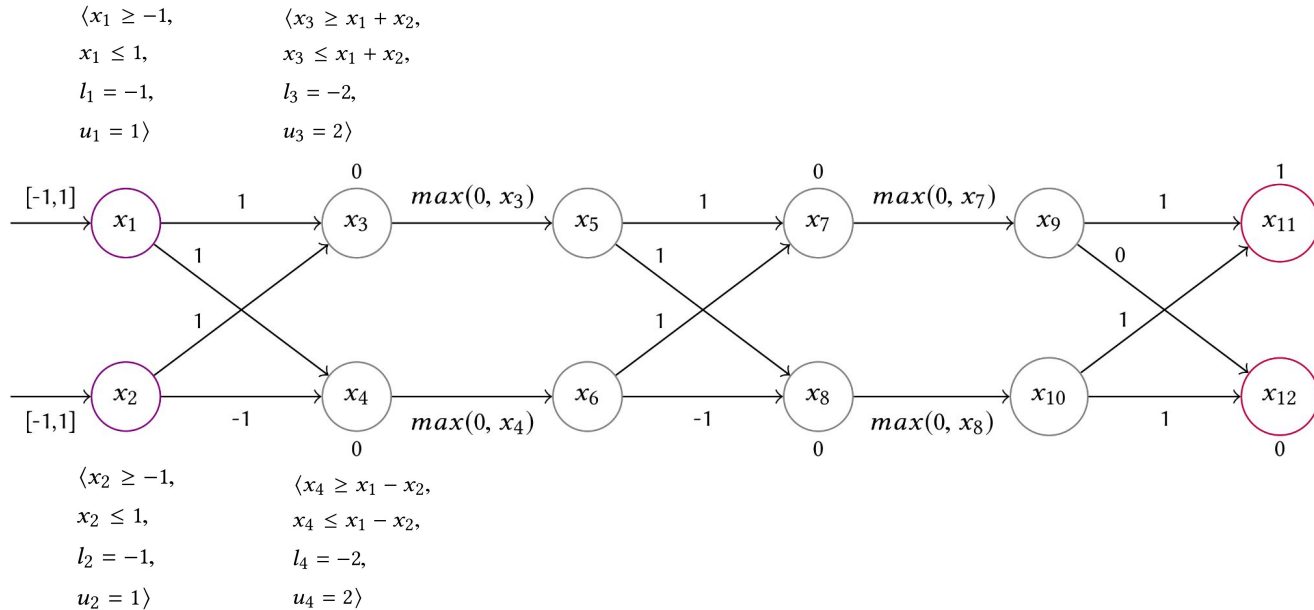
<
 symbolic lower bound,
 symbolic upper bound,
 concrete lower bound,
 concrete upper bound
 >



$$\langle x_2 \geq -1, \quad \langle x_4 \geq x_1 - x_2, \\ x_2 \leq 1, \quad x_4 \leq x_1 - x_2, \\ l_2 = -1, \quad l_4 = -2, \\ u_2 = 1 \rangle \quad u_4 = 2 \rangle$$

DeepPoly: Example

ReLU is not linear. How do we obtain the tuple for x_5 and x_6 ?

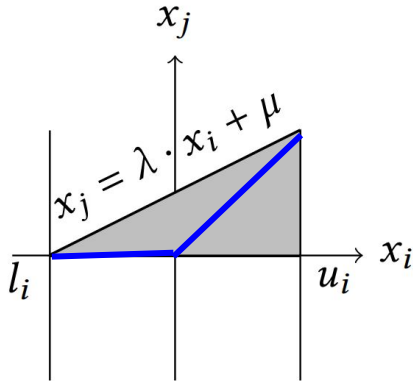


Abstract Interpretation using Linear Approximation

High-level idea

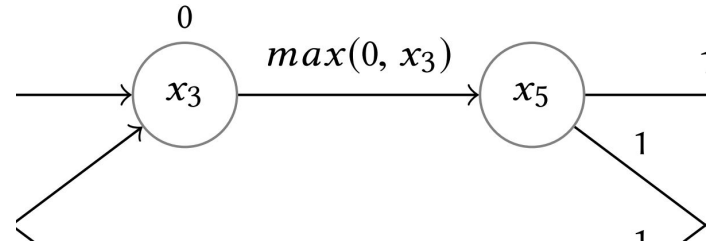
Soundly approximate the activation function using a linear constraint.

Example: $\text{ReLU}(x) = \max(0, x)$



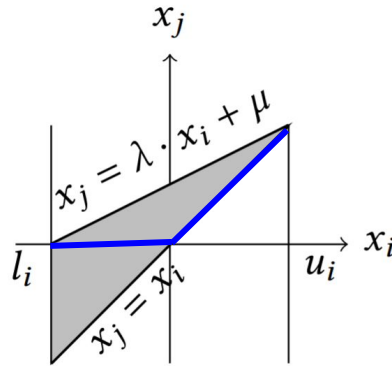
Concrete Example:

$$\begin{aligned} \langle x_3 \geq x_1 + x_2, & & \langle x_5 \geq 0, \\ x_3 \leq x_1 + x_2, & & x_5 \leq 0.5 \cdot x_3 + 1, \\ l_3 = -2, & & l_5 = 0, \\ u_3 = 2 \rangle & & u_5 = 2 \rangle \end{aligned}$$



Exercise 4

There are often different ways of approximating nonlinear functions using linear constraints. For instance, the following shows an alternative way of approximating ReLU, i.e., $x_j \leq \lambda \cdot x_i + \mu$ and $x_j \geq x_i$.



What is the resultant approximation of x_5 and x_6 using this approximation?

Exercise 4

For x_5 ,

<

$x_5 \geq$?????????,

$x_5 \leq$?????????,

$l_5 =$?????????,

$u_5 =$?????????

>

For x_6 ,

<

$x_6 \geq$?????????,

$x_6 \leq$?????????,

$l_6 =$?????????,

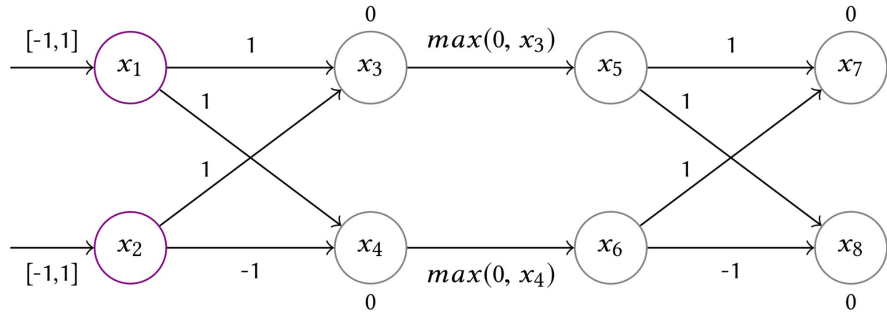
$u_6 =$?????????

>

Looking at the results, which of the two abstractions is better?

DeepPoly: Example

$$\begin{array}{lll}
 \langle x_1 \geq -1, & \langle x_3 \geq x_1 + x_2, & \langle x_5 \geq 0, \\
 x_1 \leq 1, & x_3 \leq x_1 + x_2, & x_5 \leq 0.5 \cdot x_3 + 1, \\
 l_1 = -1, & l_3 = -2, & l_5 = 0, \\
 u_1 = 1 \rangle & u_3 = 2 \rangle & u_5 = 2 \rangle
 \end{array}$$



$$\begin{array}{lll}
 \langle x_2 \geq -1, & \langle x_4 \geq x_1 - x_2, & \langle x_6 \geq 0, \\
 x_2 \leq 1, & x_4 \leq x_1 - x_2, & x_6 \leq 0.5 \cdot x_4 + 1, \\
 l_2 = -1, & l_4 = -2, & l_6 = 0, \\
 u_2 = 1 \rangle & u_4 = 2 \rangle & u_6 = 2 \rangle
 \end{array}$$

Moving on

We have $x_7 \geq x_5 + x_6$, $x_7 \leq x_5 + x_6$.

What is u_7 ?

Choice 1: $u_7 = 4$ since $x_5 \leq 2$ and $x_6 \leq 2$.

Choice 2: As $x_7 \leq x_5 + x_6$

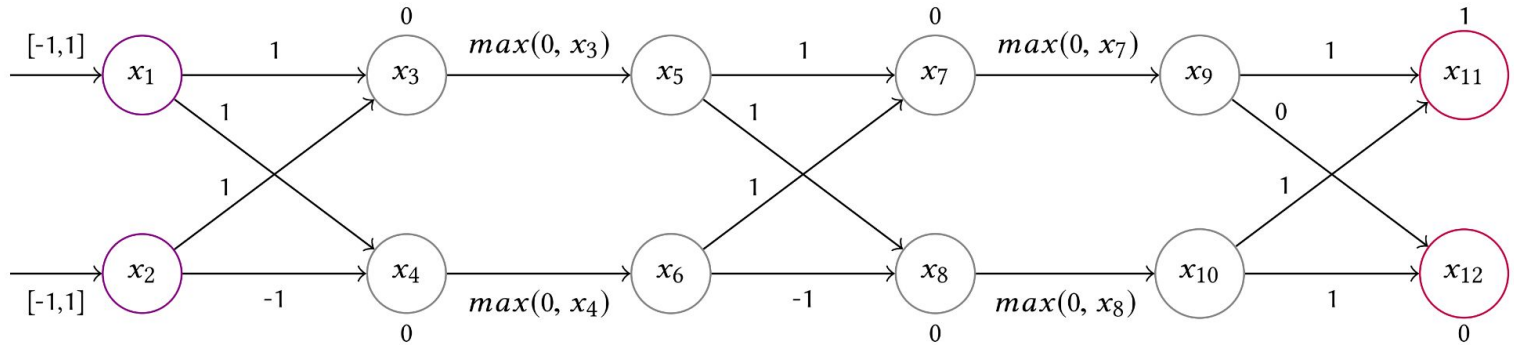
$$\begin{aligned}
 &\leq (0.5x_3 + 1) + (0.5x_4 + 1) \\
 &\leq 0.5(x_1 + x_2) + 1 + 0.5(x_1 - x_2) + 1 \\
 &= x_1 + 2 \\
 &\leq 3
 \end{aligned}$$

12

Thus, $u_7 = 3$.

DeepPoly: Example

$\langle x_1 \geq -1,$	$\langle x_3 \geq x_1 + x_2,$	$\langle x_5 \geq 0,$	$\langle x_7 \geq x_5 + x_6,$	$\langle x_9 \geq x_7,$	$\langle x_{11} \geq x_9 + x_{10} + 1,$
$x_1 \leq 1,$	$x_3 \leq x_1 + x_2,$	$x_5 \leq 0.5 \cdot x_3 + 1,$	$x_7 \leq x_5 + x_6,$	$x_9 \leq x_7,$	$x_{11} \leq x_9 + x_{10} + 1,$
$l_1 = -1,$	$l_3 = -2,$	$l_5 = 0,$	$l_7 = 0,$	$l_9 = 0,$	$l_{11} = 1,$
$u_1 = 1 \rangle$	$u_3 = 2 \rangle$	$u_5 = 2 \rangle$	$u_7 = 3 \rangle$	$u_9 = 3 \rangle$	$u_{11} = 5.5 \rangle$



$\langle x_2 \geq -1,$	$\langle x_4 \geq x_1 - x_2,$	$\langle x_6 \geq 0,$	$\langle x_8 \geq x_5 - x_6,$	$\langle x_{10} \geq 0,$	$\langle x_{12} \geq x_{10},$
$x_2 \leq 1,$	$x_4 \leq x_1 - x_2,$	$x_6 \leq 0.5 \cdot x_4 + 1,$	$x_8 \leq x_5 - x_6,$	$x_{10} \leq 0.5 \cdot x_8 + 1,$	$x_{12} \leq x_{10},$
$l_2 = -1,$	$l_4 = -2,$	$l_6 = 0,$	$l_8 = -2,$	$l_{10} = 0,$	$l_{12} = 0,$
$u_2 = 1 \rangle$	$u_4 = 2 \rangle$	$u_6 = 2 \rangle$	$u_8 = 2 \rangle$	$u_{10} = 2 \rangle$	$u_{12} = 2 \rangle$

DeepPoly: Example

Finally

$$x_{11} - x_{12} \geq x_9 + x_{10} + 1 - x_{10}$$

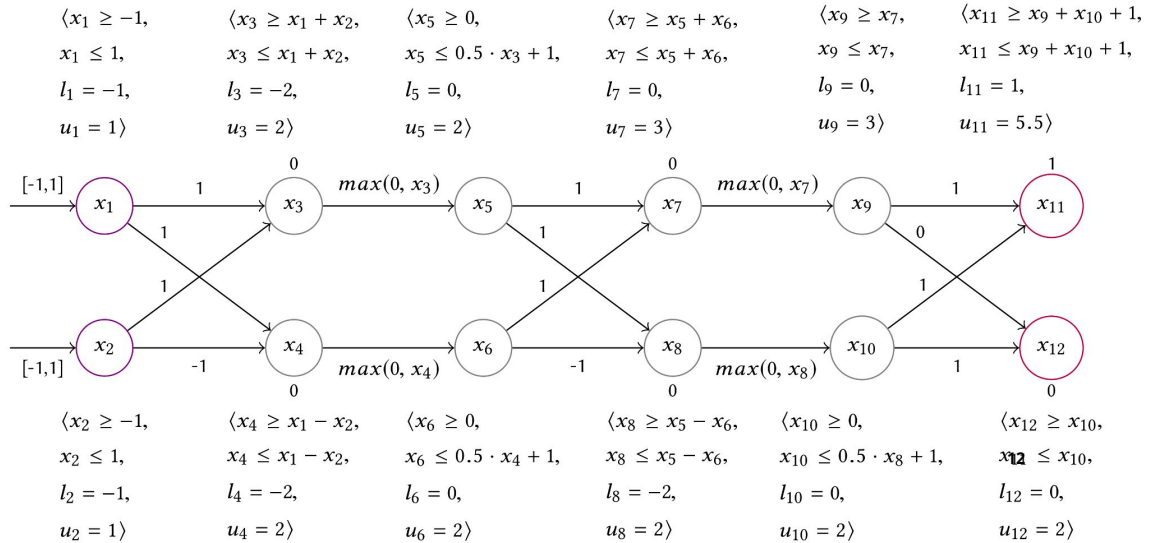
$$= x_9 + 1$$

$$\geq 1$$

$$> 0$$

Thus, the label is always 1.

It may not always be successful!



DeepPoly: Overall Approach

Approach

Abstract each neuron using linear constraints.

Conjunct all constraints (all of which are linear constraints).

Solve the constraint using a powerful linear constraint solver (such as Gruobi).

Example

$$(x_1 \geq -1 \ \&\& \ x_1 \leq 1 \ \&\& \ x_2 \geq -1 \ \&\& \ x_2 \leq 1 \ \&\&$$

$$x_3 \geq x_1 + x_2 \ \&\& \ x_3 \leq x_1 + x_2 \ \&\&$$

...

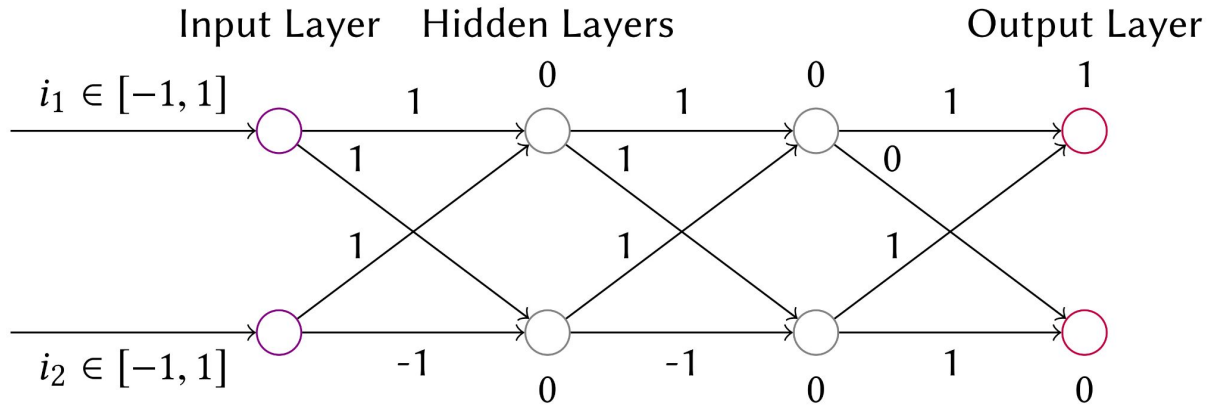
$$x_{11} \geq x_9 + x_{10} + 1 \ \&\& \ x_{11} \leq x_9 + x_{10} + 1$$

$$x_{12} \geq x_{10} \ \&\& \ x_{12} \leq x_{10}$$

$$\Rightarrow x_{11} > x_{10}$$

Exercise 5

Abstract all the neurons with the abstraction used in Exercise 4 and check whether you can prove that the output is always 1.



Evaluating Robustness

Testing

Given a neural network N and an input x , apply adversarial attacks to evaluate how easy it is to attack x .

N 's robustness with respect to x is measured based on the attacking time, or success rate.

Verification

Given a neural network N and an input x , gradually increase the attacking budget (e.g., the L_∞ -norm bound) until adversarial attack is possible.

N 's robustness with respect to x is measured based on the attacking budget.

How do we aggregate the results on many inputs and obtain the overall robustness evaluation result then?

Evaluating Robustness

Question

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

Do you think these answers are reasonable?

Testing: Possible Answer

Sample a set of inputs, 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.

Verification: Possible Answer

Sample a set of inputs, evaluate N's robustness with respect to each input x , and take the minimal (attacking budget) as a measure of the overall robustness.

Conclusion

There are many adversarial attacking methods.

PGD is typically considered to a powerful method for generating adversarial samples.

We can evaluate robustness through either testing (a.k.a. attacking) or formal verification. The latter is limited to small (thousands of neurons) neural networks of certain kinds.

Remaining Questions

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

Exercise 1

Submit a zip file containing a report (word, or pdf) and a program showing your working of Exercise 0-5 to elearn (under Assignments and Exercise 1) by Sep 5, 2022 11:59 PM.

Aug 23 - Week 1: 7-10	Introduction	
Aug 30 - Week 2: 7-10	AI Robustness	Exercise 1
Sep 06 - Week 3: 7-10	Improving AI Robustness	Exercise 2
Sep 13 - Week 4: 7-10	AI Backdoors	Exercise 3
Sep 20 - Week 5: 7-10	Mitigating AI Backdoors	Exercise 4; Project Proposal
Sep 27 - Week 6: 7-10	AI Fairness	Exercise 5
Oct 11 - Week 7: 7-10	Improving AI Fairness	Exercise 6
Oct 18 - Week 8: 7-10	AI Privacy	Exercise 7
Oct 25 - Week 9: 7-10	Improving AI Privacy	Exercise 8
Nov 01 - Week 10: 7-10	AI Interpretability	Project Due
Nov 08 - Week 11: 1-3	End-of-Term Exam	