Black-Box Adversarial Examples

Lecture 4

CS25800, Adv. ML

		Lecture Topic	Readings and Lecture Slides	Assignment
Tu	3/25	Kick off	Lecture 1	
Th	3/27	Deep Learning: Review	Lecture 2	
Fri	3/28	PyTorch Info Session	Info session materials	HW1 assigned
Tu	4/1	Evasion Attacks (White-Box)	Lecture 3 [Papers listed on lecture slides] [Optional: https://arxiv.org/pdf/1909.08072]	
Th	4/3	Evasion Attacks (Black-Box)	Lecture 4 [Optional: https://arxiv.org/pdf/1909.08072]	
Tu	4/8	Evasion Attack Detection & Prevention	Lecture 5 [Optional: https://arxiv.org/pdf/1909.08072]	
Th	4/10	Poison Attacks		
Tu	4/15	Backdoor Attacks		
Th	4/17	Poison/Backdoor Defenses		
Tu	4/22	Midterm Review		
Th	4/24	Exam 1		

Current Lecture Schedules

Canvas (under Syllabus)

Tu	4/29	Privacy Attacks (Membership Inference, Model Extraction)
Th	5/1	Watermarks and Radioactive Data
Tu	5/6	Beyond Images: Evasion Attacks on NLP/Audio/Video/Graphs
Th	5/8	Poisoning Attacks on Diffusion
Tu	5/13	Watermarks on LLMs
Th	5/15	Prompt Attacks on LLMs
Tu	5/20	Attacks/Defenses in the Physical World
Th	5/22	Exam 2

Lecture 3 Recap:

White-box Adversarial Examples

- Understand the methodology for adding adversarial perturbations $\boldsymbol{\eta}$ to an image x to cause misclassification on $(x + \boldsymbol{\eta})$
  - White-box AE attacks: When the attacker has full access to the classification model
  - Targeted attacks: $F(x+\eta; \mathbf{W}) = \mathbf{y}_{target} \neq F(x; \mathbf{W})$
  - Untargeted attacks: $F(x+\eta; \mathbf{W}) \neq F(x; \mathbf{W}) = y$
  - Three attack designs: FGSM, PGD, CW
    - Gradient-based heuristics: FGSM, PGD
    - Optimization-based: CW
- Physical-world adversarial examples
  - Non-noise "universal" adversarial patches/patterns

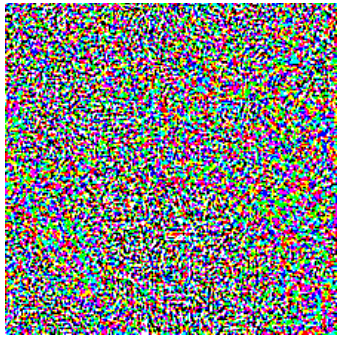
Definition: Adversarial Example (AE)

- Inputs to a ML model that an attacker intentionally designed so that the model will make mistakes (e.g. misclassification) on them

Original image

Classified as panda
57.7% confidence

Small adversarial noise
(**adversarial perturbation**)



Adversarial example

Classified as gibbon
99.3% confidence

Gibbon

Reading: Explaining and Harnessing Adversarial Examples, Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, ICLR 2015

White-box AE Attacks

- Goal: compute perturbation $\boldsymbol{\eta}$ on input x, so that (x+ $\boldsymbol{\eta}$) is misclassified, while $\boldsymbol{\eta}$ is bounded by some Lp constraint
  - Attacker knowledge: full access to the model, including its loss function, model weights
- Three key attacks:
  - FGSM
    - Untargeted attack: $\boldsymbol{\eta} = \boldsymbol{\epsilon} \cdot \boldsymbol{sign}\left(\boldsymbol{\nabla_x} \boldsymbol{J}(\mathbf{W}, \mathbf{x}, \mathbf{y})\right)$
    - Targeted attack: $\boldsymbol{\eta} = -\boldsymbol{\epsilon} \cdot \boldsymbol{sign}\left(\boldsymbol{\nabla_x} \boldsymbol{J}(\mathbf{W}, \mathbf{x}, \mathbf{y}_{\boldsymbol{target}})\right)$
  - PGD
    - Iterative FGSM
  - C&W
    - Frame as a constrained optimization

$$\text{minimize}_{\eta} \ ||x+\eta, x|| + \lambda \cdot f(x+\eta)$$

$x+\eta$ is a valid image

$$\text{minimize}_{\varpi} \ \left|\left|\frac{1}{2}(\tanh(\varpi)+1), x\right|\right| + \lambda \cdot f\left(\frac{1}{2}(\tanh(\varpi)+1)\right)$$

$$\text{where: } f(x') = \max\left(\max\{Z(x')_i : i \neq y_{target}\} - Z(x')_{y_{target}}, -\tau\right)$$

What is the point of learning/understanding these complex (optimization) algorithms when I can just call many pre-built functions and existing implementations ?

Because (optimization) algorithms and attacks are not turnkey solutions. They can fail or perform suboptimally. Using them as black boxes without understanding them is like driving a car without learning about changing flat tires.

Adaptation & Customization

- AI/ML Models are rapidly evolving
  - Advances in AI tech/research
  - Increasing complexity in AI systems and varying demands

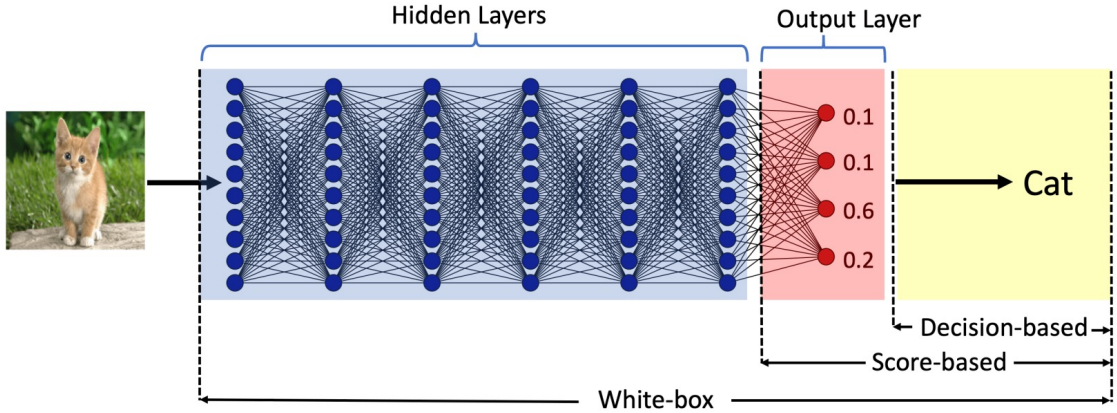
- Attacks and Defenses wrt these models are also evolving
  - ML practitioners strive to develop and deploy new, stronger defenses to counter known attacks
  - Attackers strive to adapt their strategies to exploit model/deployment vulnerabilities, leading to frequent emergence of new attack vectors.

- Attacks and Defenses are often unique to a deployment scenario
  - Open-source models (white-box attacks)
  - Private, API only models (black-box attacks)

Today's focus

Today's Focus: Black-box AE Attacks

- No knowledge of model weights, loss functions, hyperparameters etc
- Attacker can only query the model with images and analyze the output
- Attack design depends on the query output
  - Score-based vs. decision-based



Img source: Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models, ICLR 2018

Two Types of Black-box AE Attacks

- **Transfer-based attacks**
  - Attacker finds a substitute model, uses it to create adversarial examples (i.e., white-box attacks on the substitute model), sends them to the target model
  - The substitute can be 1) a single surrogate model, or 2) an ensemble of models
  - **Goal: increase attack transferability**

- **Query-based attacks**
  - Attacker queries the target model and uses the query response to create adversarial examples
  - No need to train or collect substitute models
  - Two types of query responses:
    - Class probabilities (i.e., confidence scores per class) → **score-based attacks**
    - Class label only → **decision-based attacks; label-only attacks**
  - **Goal: minimize attack cost, i.e., # of queries**

Transfer-based Black-Box AE Attacks

Find a substitute model that mimics the target model's decision behavior

Build white-box adversarial examples on the substitute model

Hope they will also cause desired misclassification on the target model

Attack Transferability

General observation of ML models

- Consider two models trained to perform the same task
- A benign input will be classified similarly by the two models
- Adversarial examples that affect one model ***often*** affect another model, even if the two models have different architectures or were trained on different training sets

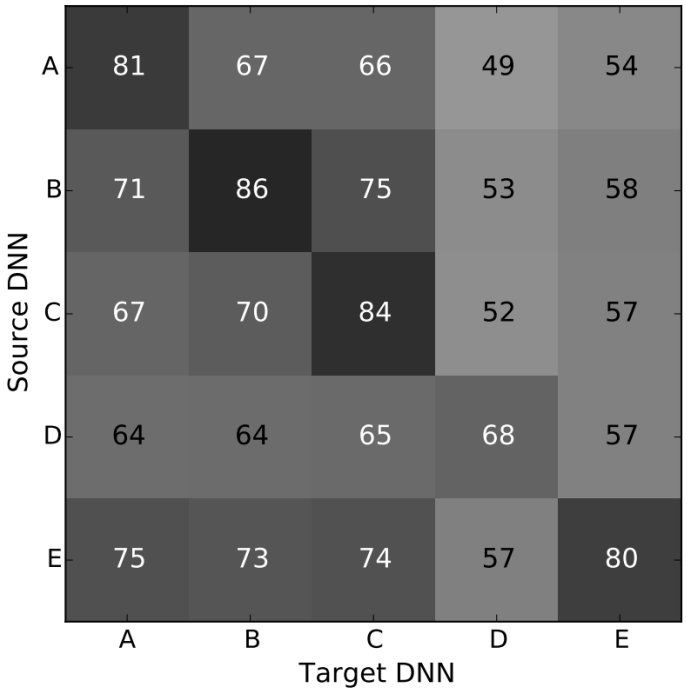
Attack strategy

- An attacker finds (or trains) a substitute model and crafts white-box adversarial examples against the substitute mode
- **Often** these adversarial examples will cause misclassification on the target model, i.e., they transfer to the target model
- Targeted AEs are much harder to transfer compared to untargeted AEs

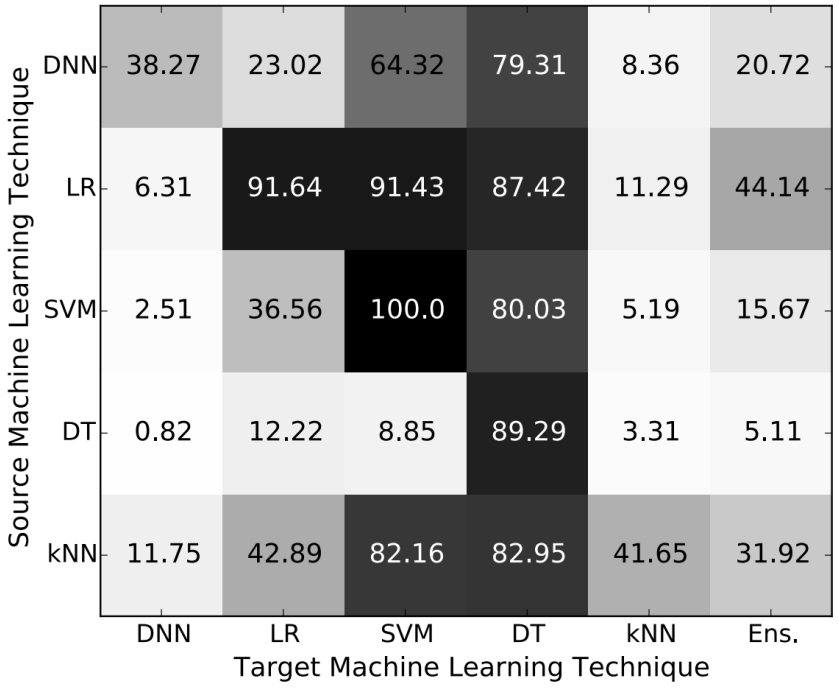
Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples, 2016

Empirical Evidence (attack transferability %, MNIST)

Same DNN Architecture, Disjoint Training Data



Different Model Architecture



Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples, 2016

Additional Observations

Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017

- Consider five models trained on ImageNet dataset (ResNet-50, ResNet-101, ResNet-152, GoogLeNet, VGG-16)
- Use optimization-based attacks (e.g., CW) and FGSM attacks

L2: $||\eta||$* 255

The target model to test the transferability of untargeted adversarial example (AE)

Substitute model used to build untargeted AE

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	22.83	0%	13%	18%	19%	11%
ResNet-101	23.81	19%	0%	21%	21%	12%
ResNet-50	22.86	23%	20%	0%	21%	18%
VGG-16	22.51	22%	17%	17%	0%	5%
GoogLeNet	22.58	39%	38%	34%	19%	0%

Panel A: Optimization-based approach

Substitute model used to build untargeted AE

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	23.45	4%	13%	13%	20%	12%
ResNet-101	23.49	19%	4%	11%	23%	13%
ResNet-50	23.49	25%	19%	5%	25%	14%
VGG-16	23.73	20%	16%	15%	1%	7%
GoogLeNet	23.45	25%	25%	17%	19%	1%

Panel B: Fast gradient approach

Cell value = % of AE built from substitute model failed to cause misclassification on the target model

= 1 – Attack transferability(%)

Observation 1: Transferability of untargeted attacks is good (60-95%) for both CW and FGSM

Observation 2: CW tends to have lower transferability than FGSM!

However, Targeted Attacks Rarely Transfer!

- A targeted attack from model A transfers to model B if and only if the produced class labels **match** each other

L2: $||\eta||* 255$

The target model to test the transferability of targeted adversarial example (AE)

Substitute model used to build targeted AE

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
ResNet-152	23.13	100%	2%	1%	1%	1%
ResNet-101	23.16	3%	100%	3%	2%	1%
ResNet-50	23.06	4%	2%	100%	1%	1%
VGG-16	23.59	2%	1%	2%	100%	1%
GoogLeNet	22.87	1%	1%	0%	1%	100%

Observation: Transferability of targeted attacks is nearly 0 !

Cell value = % of targeted AE built from sub. model succeed to cause the desired misclassification on the target model

= **Attack transferability(%)**

(note: different from prev slide)

Untargeted attacks easily transfer but targeted attacks rarely transfer

So far, Substitute Model = another DNN model for the same task

What if we choose a different substitute model ???

Two Kinds of Substitute Models

- Collect an **ensemble** of models w/ white-box access (e.g., open-source models that naturally grant white-box access, or train a set of models locally
  - **Maximize attack transferability:** Why an ensemble of models?
  - Paper Reading: Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017
- Train a new, local surrogate model by querying the target model **F**
  - **Model extraction**: Generate inputs synthetically and use them to query the target model to obtain their labels; train a local model using these labeled data.
  - Build white-box Adv. Examples on the local model
  - Paper Reading: Practical Black-Box Attacks against Machine Learning, Asia CCS 2017

(1) Substitute = An Ensemble of Models

- Create the ensemble: Train multiple surrogate models locally or gather multiple models (for the same task) with white-box access to each
- Assumption: If an adversarial example succeeds on all the models in the ensemble, it will (more) likely succeed on the target model

- How to find such adversarial example?
- Define the right optimization objective: replace single model loss $\boldsymbol{J}(\mathbf{W}, \mathbf{x}, \mathbf{y})$ with **weighted sum of loss across multiple models** :

$$\sum_k \alpha_k \boldsymbol{J}(\mathbf{W_k}, \mathbf{x}, \mathbf{y}) \text{ where } \sum_k \alpha_k = 1$$

Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017

Using an Ensemble of Models Boosts Transferability of Targeted Attacks

L2: $||\eta||$* 255

The target model to test the transferability of targeted adversarial example (AE)

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	30.68	38%	76%	70%	97%	76%
-ResNet-101	30.76	75%	43%	69%	98%	73%
-ResNet-50	30.26	84%	81%	46%	99%	77%
-VGG-16	31.13	74%	78%	68%	24%	63%
-GoogLeNet	29.70	90%	87%	83%	99%	11%

Ensemble of 4 models, marked by the model excluded from the ensemble

Cell value: % of targeted AEs built from model ensemble that succeed on the target model = **attack transferability(%)**

Delving into Transferable Adversarial Examples and Black-box Attacks, ICLR 2017

(2) Substitute= Training a Surrogate Model Locally

- **Leverage model extraction attacks**:
  - Generate inputs synthetically and use them to query the target model to obtain their labels;
  - Train a local model using these labeled data;
  - Build white-box adv. examples on the local model

- **Choices of training hyperparameters matter!**
  - Low complexity surrogate classifiers generally provide stabler gradients better aligned with those of the target model;
  - Should not overfit the surrogate model to its training data

Theoretical Analysis of Attack Transferability

- Will not cover in this course
- But you can refer to this paper: "*Why Do Adversarial Attacks Transfer? Explaining Transferability of Evasion and Poisoning Attacks," USENIX Security 2019*

- Key Insight: decreasing complexity of the surrogate model by properly adjusting the hyperparameters of its learning algorithm provides adversarial examples that transfer better to a range of models

Pros and Cons of Transfer-based Attacks

Pros

- Efficient if the attacker needs to create adversarial examples for many input x
- Model Ensemble: No need to query the target model → efficient
- Query-based Surrogate: a successful model extraction → powerful white-box adversarial attacks

Cons

- No guarantee of success
  - DNN classification's decision boundaries are extremely complex to model/analyze
- High training cost for building surrogate models
- Transferability of targeted (or highly optimized) attacks is generally lower than untargeted (or less complex) attacks

		Lecture Topic	Readings and Lecture Slides	Assignment
Tu	3/25	Kick off	Lecture 1	
Th	3/27	Deep Learning: Review	Lecture 2	
Fri	3/28	PyTorch Info Session	Info session materials	HW1 assigned
Tu	4/1	Evasion Attacks (White-Box)	Lecture 3 [Papers listed on lecture slides] [Optional: https://arxiv.org/pdf/1909.08072]	
Th	4/3	Evasion Attacks (Black-Box)	Lecture 4 [Optional: https://arxiv.org/pdf/1909.08072]	
Tu	4/8	Evasion Attack Detection & Prevention	Lecture 5 [Optional: https://arxiv.org/pdf/1909.08072]	
Th	4/10	Poison Attacks		
Tu	4/15	Backdoor Attacks		
Th	4/17	Poison/Backdoor Defenses		
Tu	4/22	Midterm Review		
Th	4/24	Exam 1		

Current Lecture Schedules

Canvas (under Syllabus)

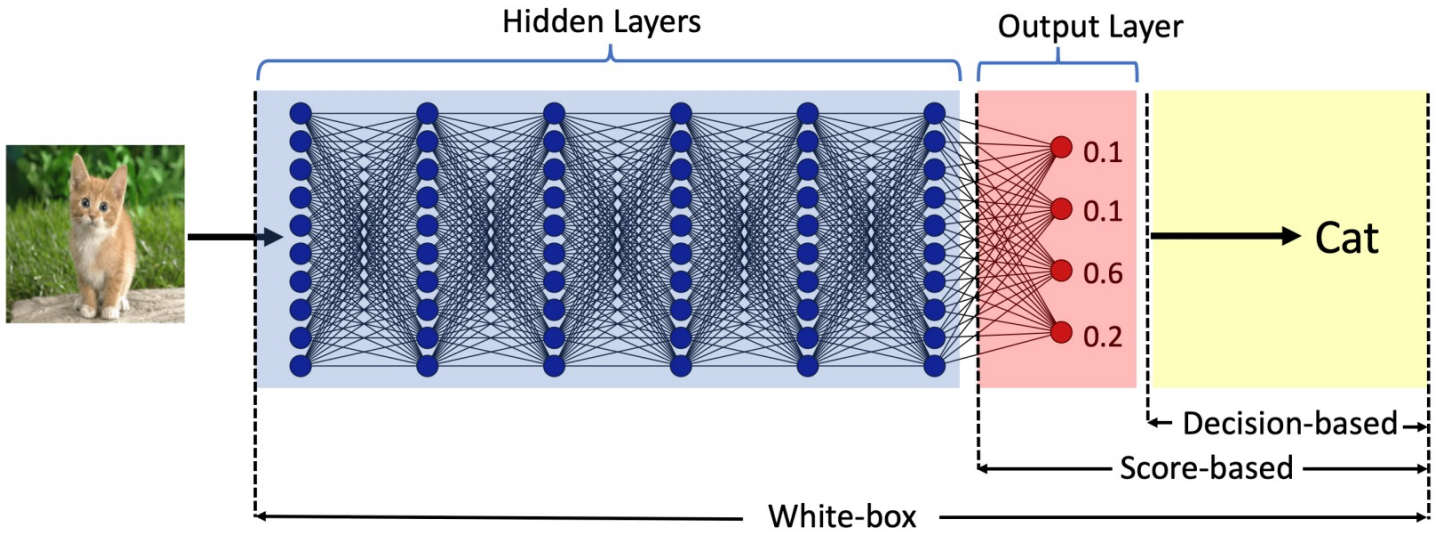
Tu	4/29	Privacy Attacks (Membership Inference, Model Extraction)
Th	5/1	Watermarks and Radioactive Data
Tu	5/6	Beyond Images: Evasion Attacks on NLP/Audio/Video/Graphs
Th	5/8	Poisoning Attacks on Diffusion
Tu	5/13	Watermarks on LLMs
Th	5/15	Prompt Attacks on LLMs
Tu	5/20	Attacks/Defenses in the Physical World
Th	5/22	Exam 2

Query-based Black-box AE Attacks

No need to train/use substitute models;
build adv. examples by querying the target model

Attack Design Depends on Query Response Format!!

- Score-based attack: query returns confidence for all (or top-k) classes
- Decision-based attack: query returns only a single label (or top-k label), no confidence value



Basis for Query-based Black-box AE Attacks

- Recall: what model information is being used by white-box AE attacks?

  - **Untargeted FGSM/PGD:** $\boldsymbol{\eta = \epsilon \cdot sign\left(\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y})\right)}$
  - **Targeted FGSM/PGD:** $\boldsymbol{\eta = -\epsilon \cdot sign\left(\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y}_{target})\right)}$
  - **Targeted C&W:** $f(x+\boldsymbol{\eta}) = \max\left(\max\{Z(x+\boldsymbol{\eta})_i : i \neq y_{target}\} - Z(x+\boldsymbol{\eta})_{y_{target}}, -\tau\right)$

Gradient! → Can attacker estimate gradients by querying the model?

A Closer Look at $\nabla_x J(\mathbf{W}, \mathbf{x}, \mathbf{y})$

- Untargeted FGSM/PGD: $\eta = \epsilon \cdot sign\left(\nabla_x J(\mathrm{W}, \mathrm{x}, \mathrm{y})\right)$
- Targeted FGSM/PGD: $\eta = -\epsilon \cdot sign\left(\nabla_x J(\mathrm{W}, \mathrm{x}, \mathrm{y}_{target})\right)$

Cross-entropy loss for a given x (a cat image) and its label y ("cat")

$$J(\mathrm{W}, x, y) = -\log P(y|x)$$

$P(y|x)$: confidence value (i.e., softmax prob.) of predicting x as class y

$$sign\left(\nabla_x J(W, x, y)\right) = sign\left(-\frac{\nabla_x P(y|x)}{P(y|x)}\right) = sign\left(-\nabla_x P(y|x)\right)$$

Only need to compute $\boldsymbol{\nabla_x P(y|x)}$ or $\boldsymbol{\nabla_x P(y_{target}|x)}$

Method 1:
Gradient Estimation Attacks with Limited Queries

Black-box Adversarial Attacks with Limited Queries and Information, ICML 2018

NES (NATURAL EVOLUTIONARY STRATEGIES)

- Estimate the gradient of a function F over input θ, using a finite-differences estimate on a random Gaussian basis:

$$\nabla \mathbb{E}[F(\theta)] \approx \frac{1}{\sigma n} \sum_{i=1}^{n} \delta_i F(\theta + \sigma \delta_i)$$

Consider: ***F*** **(x) = P(y|x)**
→ Query the model with x, return P(y|x) as the softmax probability of x being class y

Algorithm 1 NES Gradient Estimate

Input: Classifier $P(y|x)$ for class y, image x
Output: Estimate of $\nabla P(y|x)$
Parameters: Search variance σ, number of samples n, image dimensionality N

$g \leftarrow \mathbf{0}_n$
for $i = 1$ **to** n **do**
  $u_i \leftarrow \mathcal{N}(\mathbf{0}_N, \boldsymbol{I}_{N \cdot N})$
  $g \leftarrow g + P(y|x + \sigma \cdot u_i) \cdot u_i$
  $g \leftarrow g - P(y|x - \sigma \cdot u_i) \cdot u_i$
end for
return $\frac{1}{2n\sigma} g$

Black-box Adversarial Attacks with Limited Queries and Information, ICML 2018

Three NES Attacks with Different Threat Models

- Query-limited Attack
  - Query response is class probabilities, i.e., $\mathrm{P}(\mathrm{y}|\mathrm{x})$ for each of top-k classes
    - "Cat" 0.7, "dog" 0.1, "monkey" 0.1, "snake" 0.1
- Partial Information Attack
  - For targeted attacks where $\boldsymbol{y_{target}}$ ("donut") does not show up in the original image's top-k class list, i.e., $\boldsymbol{y_{target}}$ is very "far" from $\boldsymbol{y}$
- Label-only Attack
  - Query response is just top-k classes without any probability $\mathrm{P}(\mathrm{y}|\mathrm{x})$

NES Query-Limited Attack

- A score-based attack:
  - Query response is class probabilities, i.e., $\mathrm{P(y|x)}$ for top-k classes

- Following the PGD attack by estimating gradient
  - Use NES based queries to estimate $\boldsymbol{\nabla_x P\,(y|x)}$ or $\boldsymbol{\nabla_x P\,(y_{target}|x)}$
    - Untargeted FGSM/PGD: $\eta = \epsilon \cdot sign\left(\boldsymbol{\nabla_x P\,(y|x)}\right)$
    - Targeted FGSM/PGD: $\eta = -\epsilon \cdot sign\left(\boldsymbol{\nabla_x P\,(y_{target}|x)}\right)$
  - **Update x and repeat**

Black-box Adversarial Attacks with Limited Queries and Information, ICML 2018

NES Partial Information Attack

Challenge: Can't estimate $\nabla_x P\left(y_{target}|x\right)$ when y_{target} does not even appear in top-k classes reported by the model query result

Solution: find x_0 that makes y_{target} into top-k

- Begin the search by setting x_0 as a benign image of y_{target}
- CLIP x_0 to the allowed perturbation of x, and make y_{target} appear in the top-k classes
- Iteratively update to reach a successful AE

Algorithm 2 Partial Information Attack

Input: Initial image x, Target class y_{adv}, Classifier $P(y|x) : \mathbb{R}^n \times \mathcal{Y} \rightarrow [0,1]^k$ (access to probabilities for y **in top** k), image x
Output: Adversarial image x_{adv} with $||x_{adv} - x||_\infty \leq \epsilon$
Parameters: Perturbation bound ϵ_{adv}, starting perturbation ϵ_0, NES Parameters (σ, N, n), epsilon decay δ_ϵ, maximum learning rate η_{max}, minimum learning rate η_{min}

$\epsilon \leftarrow \epsilon_0$
$x_{adv} \leftarrow$ image of target class y_{adv}
$x_{adv} \leftarrow \text{CLIP}(x_{adv}, x - \epsilon, x + \epsilon)$
while $\epsilon > \epsilon_{adv}$ or $\max_y P(y|x) \neq y_{adv}$ **do**
  $g \leftarrow \text{NESESTGRAD}(P(y_{adv}|x_{adv}))$
  $\eta \leftarrow \eta_{max}$
  $\hat{x}_{adv} \leftarrow x_{adv} - \eta g$
  while not $y_{adv} \in \text{TOP-K}(P(\cdot|\hat{x}_{adv}))$ **do**
    if $\eta < \eta_{min}$ **then**
      $\epsilon \leftarrow \epsilon + \delta_\epsilon$
      $\delta_\epsilon \leftarrow \delta_\epsilon / 2$
      $\hat{x}_{adv} \leftarrow x_{adv}$
      break
    end if
    $\eta \leftarrow \frac{\eta}{2}$
    $\hat{x}_{adv} \leftarrow \text{CLIP}(x_{adv} - \eta g, x - \epsilon, x + \epsilon)$
  end while
  $x_{adv} \leftarrow \hat{x}_{adv}$
  $\epsilon \leftarrow \epsilon - \delta_\epsilon$
end while
return x_{adv}

Black-box Adversarial Attacks with Limited Queries and Information, ICML 2018

NES Label-Only Attack

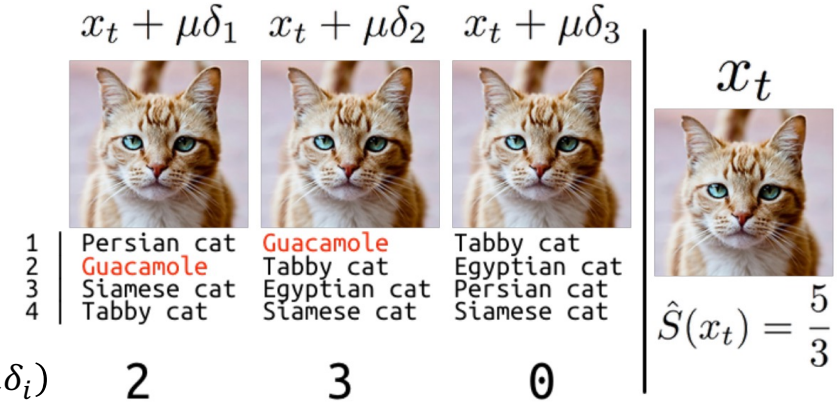
- A decision-based attack:
  - Query response is just the top-k class list, w/o probability value
  - Label only, does not return P(y|x)

  - First estimate P(y|x) by adding gaussian noise to x and observing the ranking of y in the top-*k* decision list (*k*>1)

$\hat{S}(y|x)=\frac{1}{n}\sum_{i=1}^{n}(k-rank(y|x+\mu\delta_i))$

  - Compute gradient on $\hat{S}(y|x)$

rather than P(y|x)



Attack Evaluation

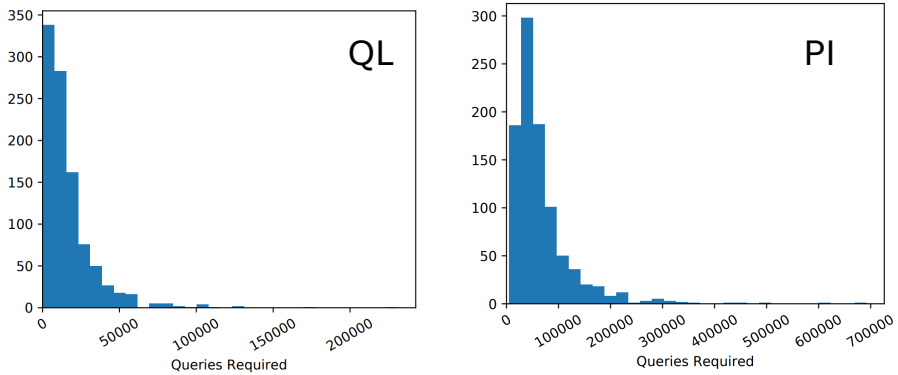
- InceptionV3 model trained on ImageNet
- randomly chosen target class
- L_∞ <0.05 (perturbation budget)

Threat model	Success rate	Median queries
QL	99.2%	11,550
PI	93.6%	49,624
LO	90%	2.7×10^6

QL: Query Limited
PI: Partial Information (k=1)
LO: Label Only (k=1)

General	
σ for NES	0.001
n, size of each NES population	50
ϵ, l_∞ distance to the original image	0.05
η, learning rate	0.01
Partial-Information Attack	
ϵ_0, initial distance from source image	0.5
δ_ϵ, rate at which to decay ϵ	0.001
Label-Only Attack	
m, number of samples for proxy score	50
μ, ℓ_∞ radius of sampling ball	0.001

Table 2. Hyperparameters used for evaluation



Queries Required for Successful AE

Today: Azure Face Identification Query Cost : $0.8 per 1000 queries
→ Median cost of ($9.24 , $39, $2160) to create an AE per image for the three attacks

Successful Adv. Examples (L_∞ <0.05)

Benign Image
& classification result


Lionfish


Tiger Beetle


Magpie


Rhodesian Ridgeback

Adv. Image
& classification result


Eggnog


Green Mamba


Great Pyrenees


Shopping Basket

Pros and Cons of NES (gradient estimation attacks)

Pros

- Operates directly on target model
- Gradient estimation turns a black-box attack into a White-box attack
- Query cost << Model training cost

Cons

- Query cost (especially under label-only) is high

Method 2:
Boundary Attack and HopSkipJump Attack

starts from a large adversarial perturbation $\boldsymbol{\eta}$ that makes (x+ $\boldsymbol{\eta}$) misclassified, iteratively reduces perturbation $\boldsymbol{\eta}$ while ensuring (x+ $\boldsymbol{\eta}$) remains misclassified

Papers:
- Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models, ICLR 2018
- HopSkipJumpAttack: A Query-Efficient Decision-Based Attack, IEEE S&P 2020

Boundary Attack: Overview

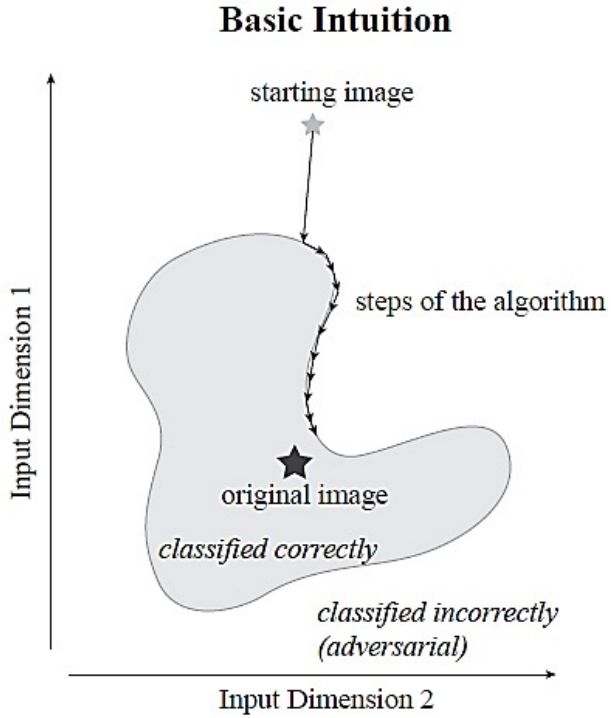
- Decision-based attacks: only need top-1 class label returned by the model query
  - Same threat model of NES Label-only attack with k=1
  - Parallel work

  - Fits real-world model deployment
  - Supports both targeted and untargeted attacks
  - Caveat: needs (many) more queries

Intuition (for untargeted attack)

- Given an original image (x) and its class (y) predicted by the target model
- Start from an image x_{adv} (e.g., random noise) whose model classification is NOT y
- Iteratively perturb x_{adv} to reduce the difference between x and x_{adv}, while keeping x_{adv} NOT classified as y

Walk along the boundary between the adversarial and the non-adversarial region, but stay in the adversarial region



Perform rejection sampling with a suitable proposal distribution P to find progressively smaller adversarial perturbations that produce desired misclassification

Data: original image $\mathbf{o}$, adversarial criterion $c(.)$, decision of model $d(.)$
Result: adversarial example $\tilde{\boldsymbol{o}}$ such that the distance $d(\boldsymbol{o}, \tilde{\boldsymbol{o}}) = \|\boldsymbol{o} - \tilde{\boldsymbol{o}}\|_2^2$ is minimized

initialization: $k = 0$, $\tilde{\boldsymbol{o}}^0 \sim \mathcal{U}(0, 1)$ s.t. $\tilde{\boldsymbol{o}}^0$ is adversarial;
while $k <$ *maximum number of steps* **do**
  draw random perturbation from proposal distribution $\boldsymbol{\eta}_k \sim \mathcal{P}(\tilde{\boldsymbol{o}}^{k-1})$;
  if $\tilde{\boldsymbol{o}}^{k-1} + \boldsymbol{\eta}_k$ *is adversarial* **then**
    set $\tilde{\boldsymbol{o}}^k = \tilde{\boldsymbol{o}}^{k-1} + \boldsymbol{\eta}_k$;
  else
    set $\tilde{\boldsymbol{o}}^k = \tilde{\boldsymbol{o}}^{k-1}$;
  end
  $k = k + 1$
end

Algorithm 1: Minimal version of the Boundary Attack.

HopSkipJump Attack (HSJA)

- Also called Boundary Attack ++
  - Boundary attack takes baby steps
  - Combine Boundary Attack with Gradient Estimation to reduce # of queries
  - Decide each step size carefully!

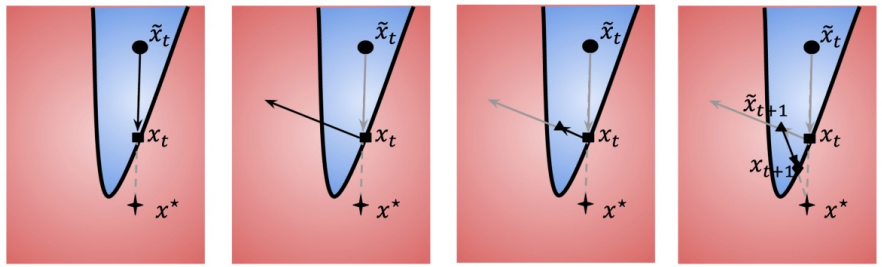
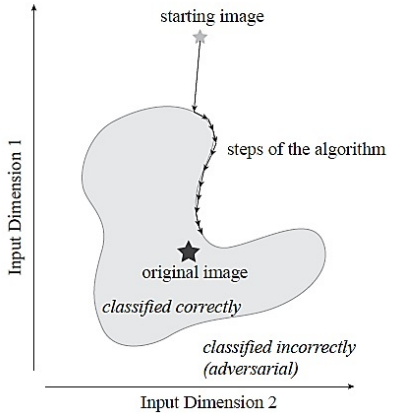
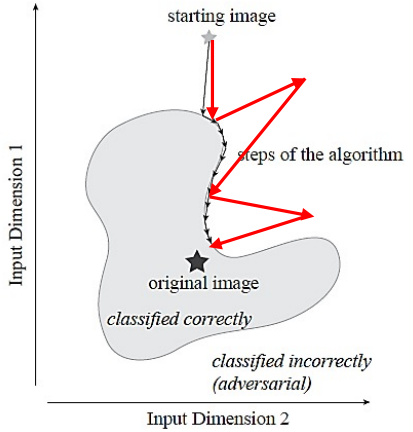


Figure 2: Intuitive explanation of HopSkipJumpAttack. (a) Perform a binary search to find the boundary, and then update $\tilde{x}_t \rightarrow x_t$. (b) Estimate the gradient at the boundary point x_t. (c) Geometric progression and then update $x_t \rightarrow \tilde{x}_{t+1}$. (d) Perform a binary search, and then update $\tilde{x}_{t+1} \rightarrow x_{t+1}$.

Boundary Attack



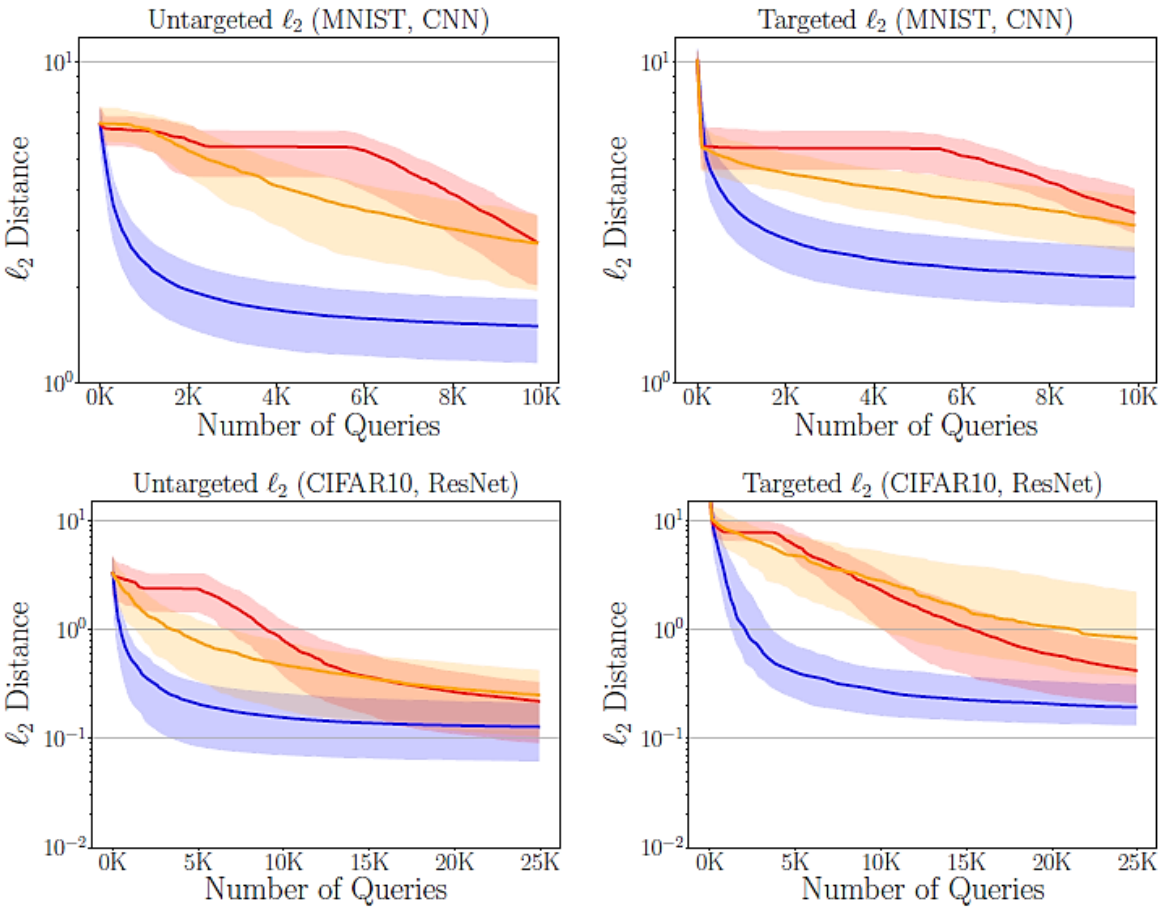
HopSkipJump Attack



HopSkipJumpAttack: A Query-Efficient Decision-Based Attack, IEEE S&P 2020

HSJA vs. Boundary Attack

- MNIST and CIFAR10
- HopSkipJump (blue) quickly lowers L_2 perturbation while maintaining misclassification, using fewer queries



Trajectories on ImageNet

Untargeted ℓ_2 Attack

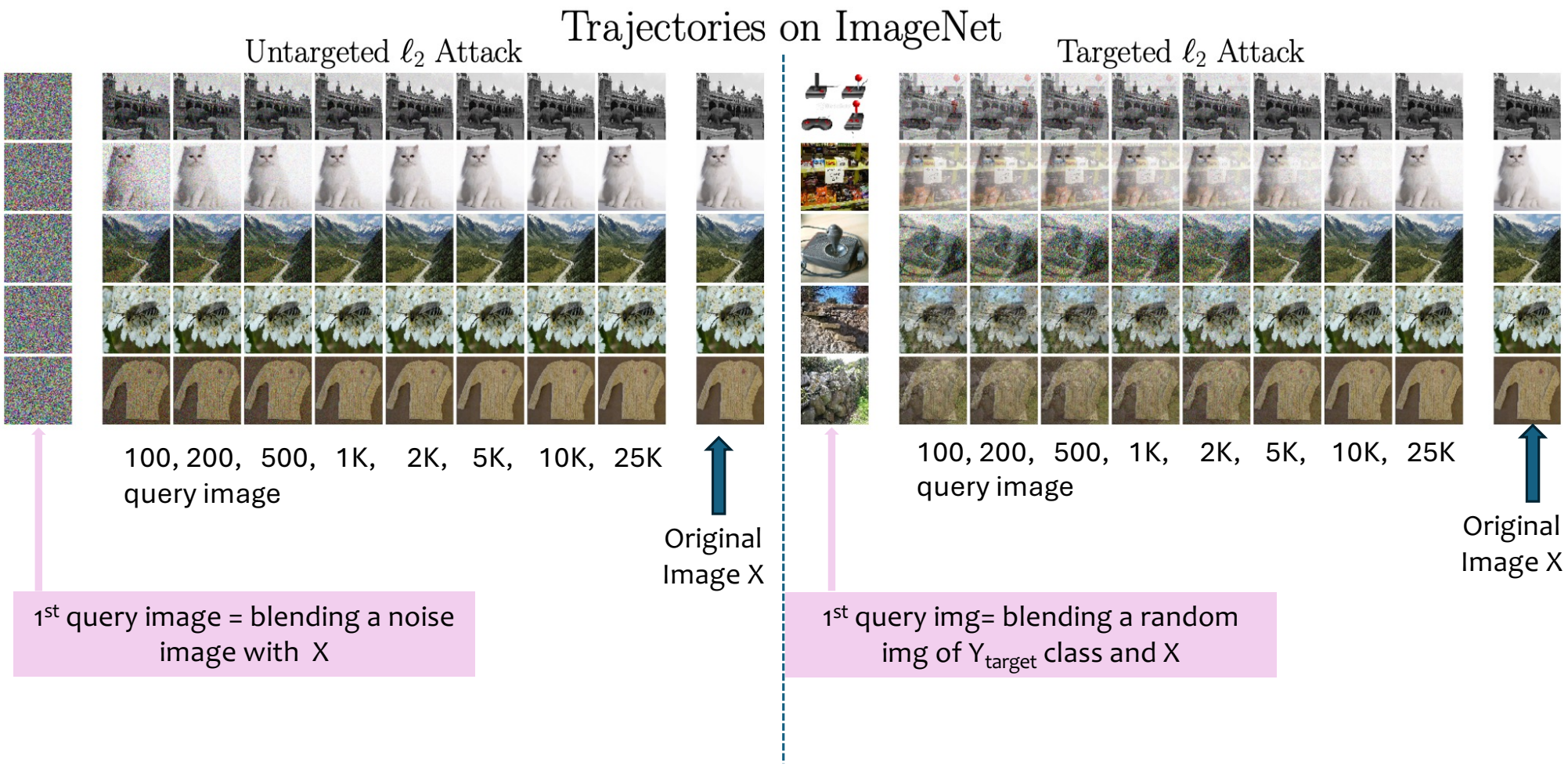
100, 200, 500, 1K, 2K, 5K, 10K, 25K

query image

Original Image X

1st query image = blending a noise image with X

Targeted ℓ_2 Attack


Trajectories on ImageNet
Untargeted ℓ_2 Attack
Targeted ℓ_2 Attack
100, 200, 500, 1K, 2K, 5K, 10K, 25K
query image
Original Image X
100, 200, 500, 1K, 2K, 5K, 10K, 25K
query image
Original Image X
1st query image = blending a noise image with X
1st query img= blending a random img of Y_{target} class and X

Pros and Cons of Boundary/HSJA

Pros

- Operates directly on target model
- Fits most deployed models: they only return the top-1 class label, not the confidence score
- HSJA largely reduces query cost

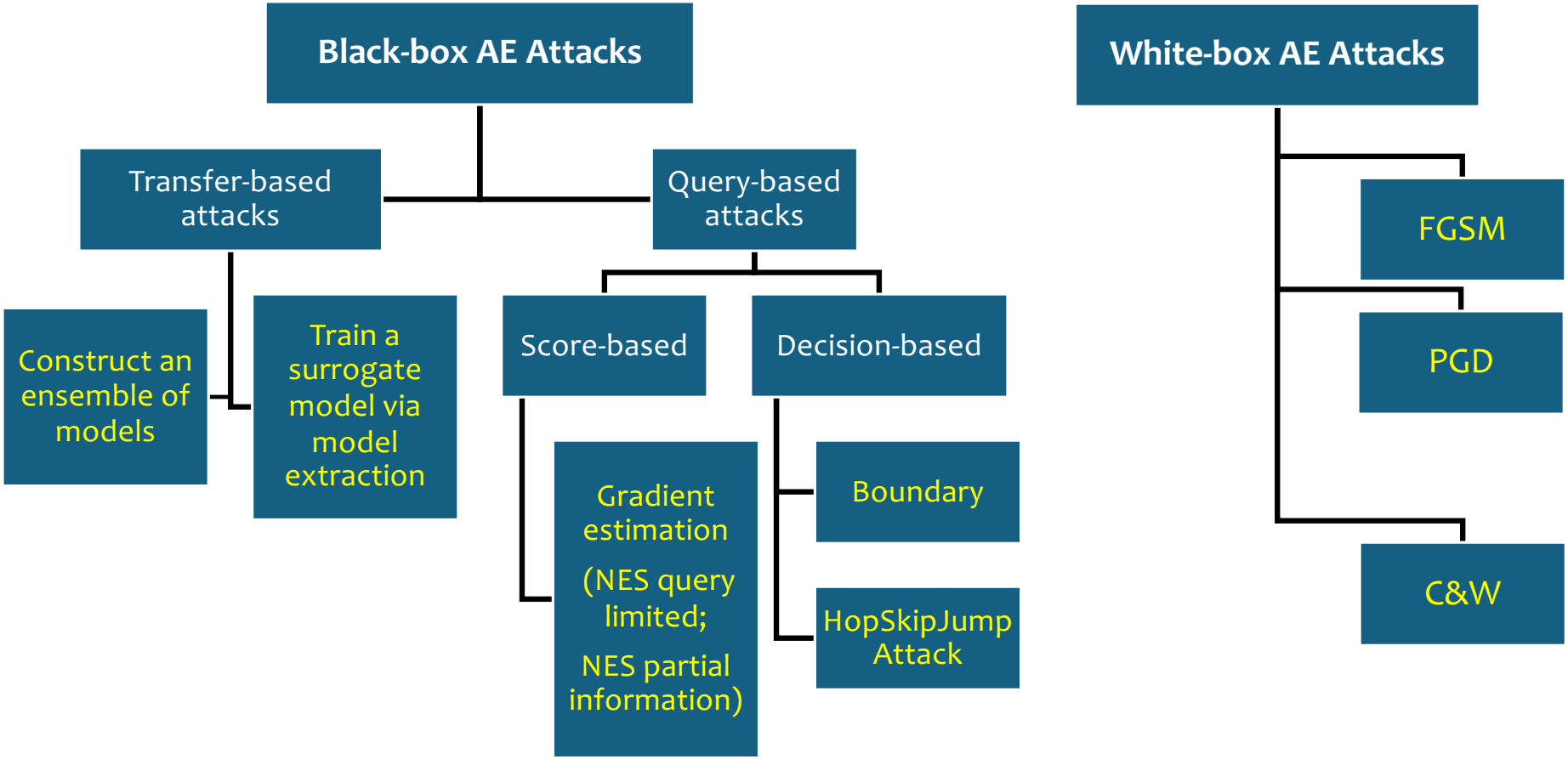
Cons

- high query cost
- 10K-20K queries per AE
- $8-$16 per AE

Summary: Two Groups of Black-box AE Attacks

- **Transfer-based attacks**
  - Attacker finds a substitute model, uses it to create adversarial examples, and sends them to the target model
  - The substitute can be 1) a single surrogate model, or 2) an ensemble of models
  - **Goal: increase attack transferability**

- **Query-based attacks**
  - Attacker queries the target model and uses the query response to create adversarial examples
  - No need to train or collect substitute models
  - Two types of query responses:
    - Class probabilities (i.e., confidence scores per class) → **score-based attacks**
    - Class label only → **decision-based attacks; label-only attacks**
  - **Goal: minimize attack cost, i.e., # of queries**


Black-box AE Attacks
Transfer-based attacks
Query-based attacks
Construct an ensemble of models
Train a surrogate model via model extraction
Score-based
Decision-based
Gradient estimation
(NES query limited;
NES partial information)
Boundary
HopSkipJump Attack
White-box AE Attacks
FGSM
PGD
C&W