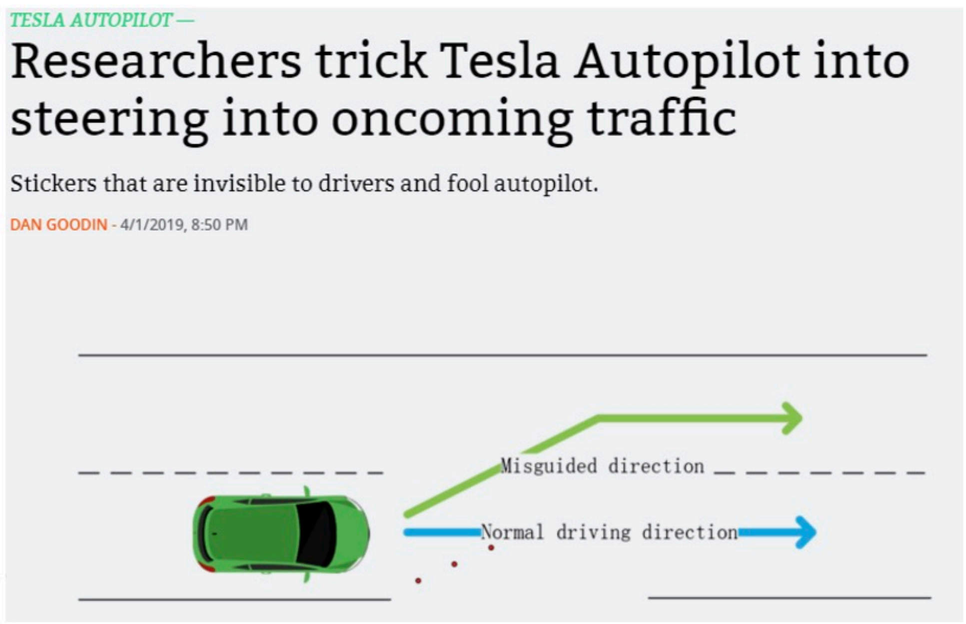
COMP6211I: Trustworthy Machine Learning

Test-time Integrity (attacks)

Minhao CHENG

Machine learning

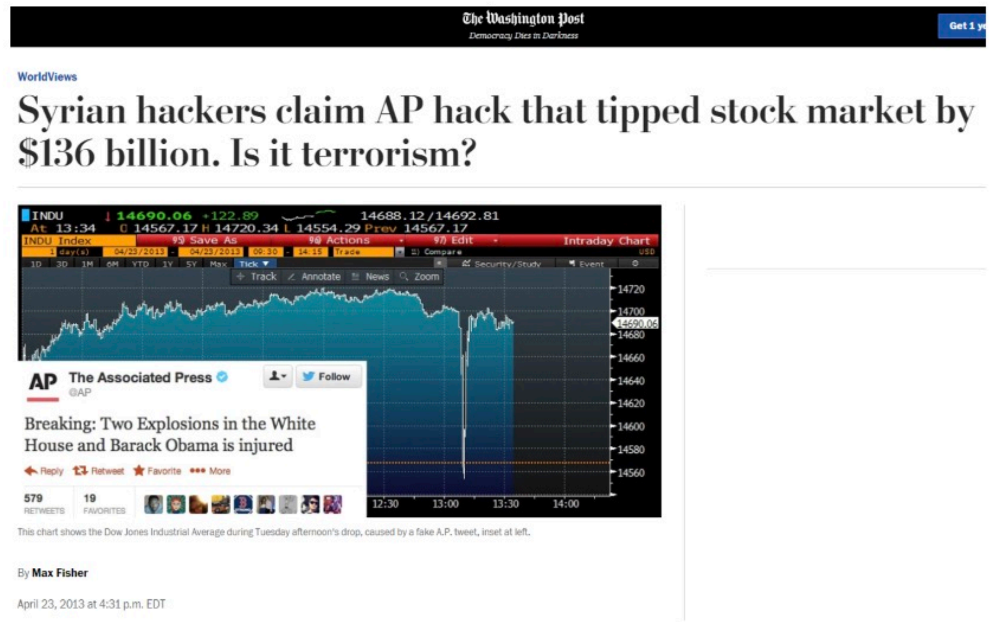
Beyond Accuracy


TESLA AUTOPILOT —

Researchers trick Tesla Autopilot into steering into oncoming traffic

Stickers that are invisible to drivers and fool autopilot.

DAN GOODIN - 4/1/2019, 8:50 PM


The Washington Post

Democracy Dies in Darkness

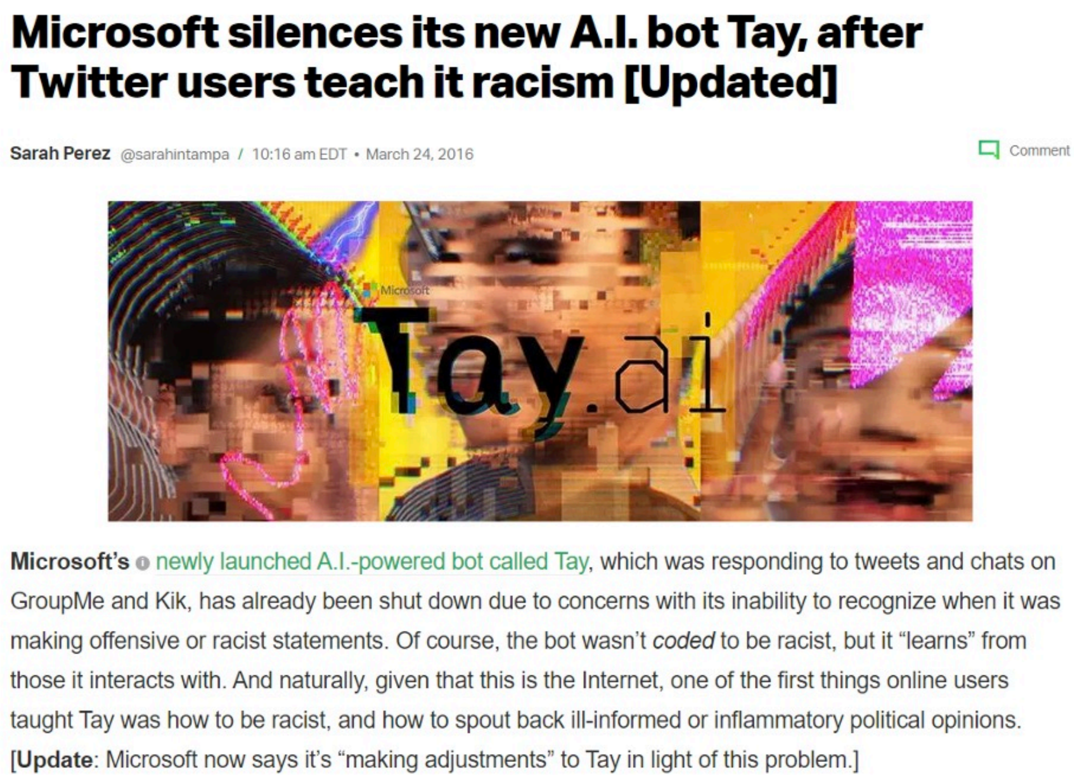
WorldViews

Syrian hackers claim AP hack that tipped stock market by $136 billion. Is it terrorism?

This chart shows the Dow Jones Industrial Average during Tuesday afternoon's drop, caused by a fake A.P. tweet, inset at left.

By Max Fisher

April 23, 2013 at 4:31 p.m. EDT


Microsoft silences its new A.I. bot Tay, after Twitter users teach it racism [Updated]

Sarah Perez @sarahintampa / 10:16 am EDT • March 24, 2016

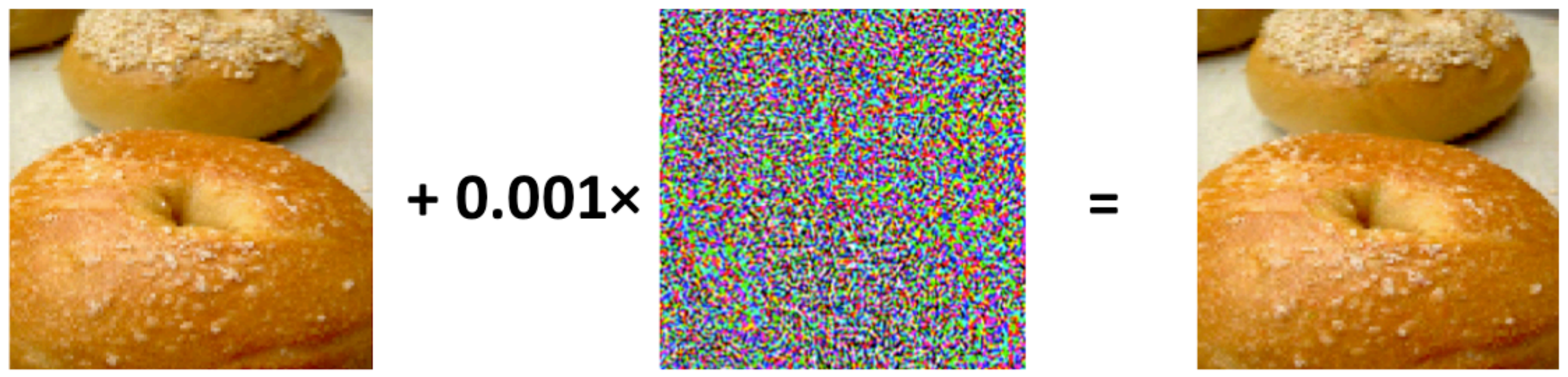
Microsoft's newly launched A.I.-powered bot called Tay, which was responding to tweets and chats on GroupMe and Kik, has already been shut down due to concerns with its inability to recognize when it was making offensive or racist statements. Of course, the bot wasn't *coded* to be racist, but it "learns" from those it interacts with. And naturally, given that this is the Internet, one of the first things online users taught Tay was how to be racist, and how to spout back ill-informed or inflammatory political opinions. [**Update**: Microsoft now says it's "making adjustments" to Tay in light of this problem.]

Test-time integrity

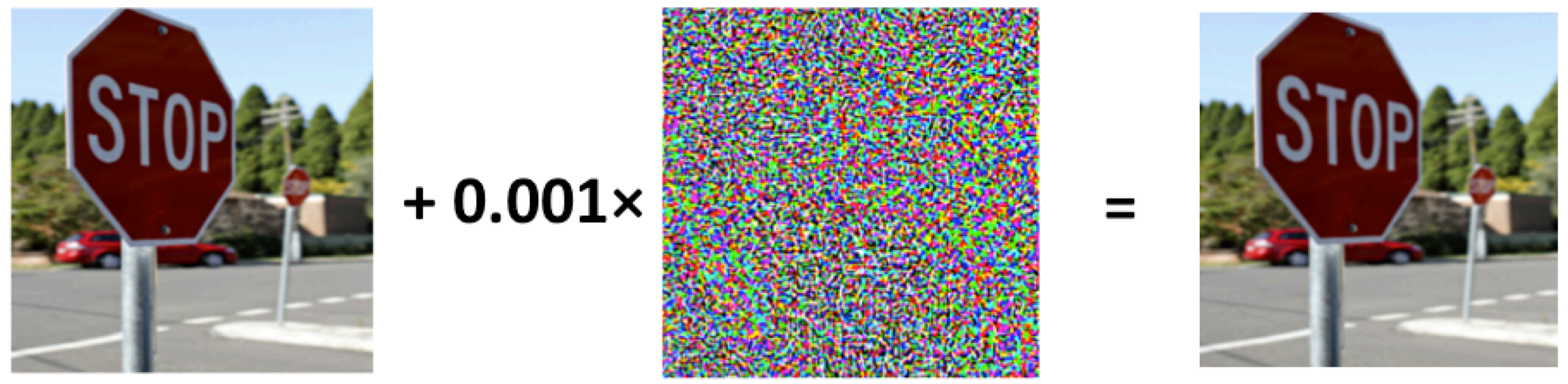
Adversarial examples

- An adversarial example can easily fool a deep network

- Robustness is critical in real systems



Bagle piano

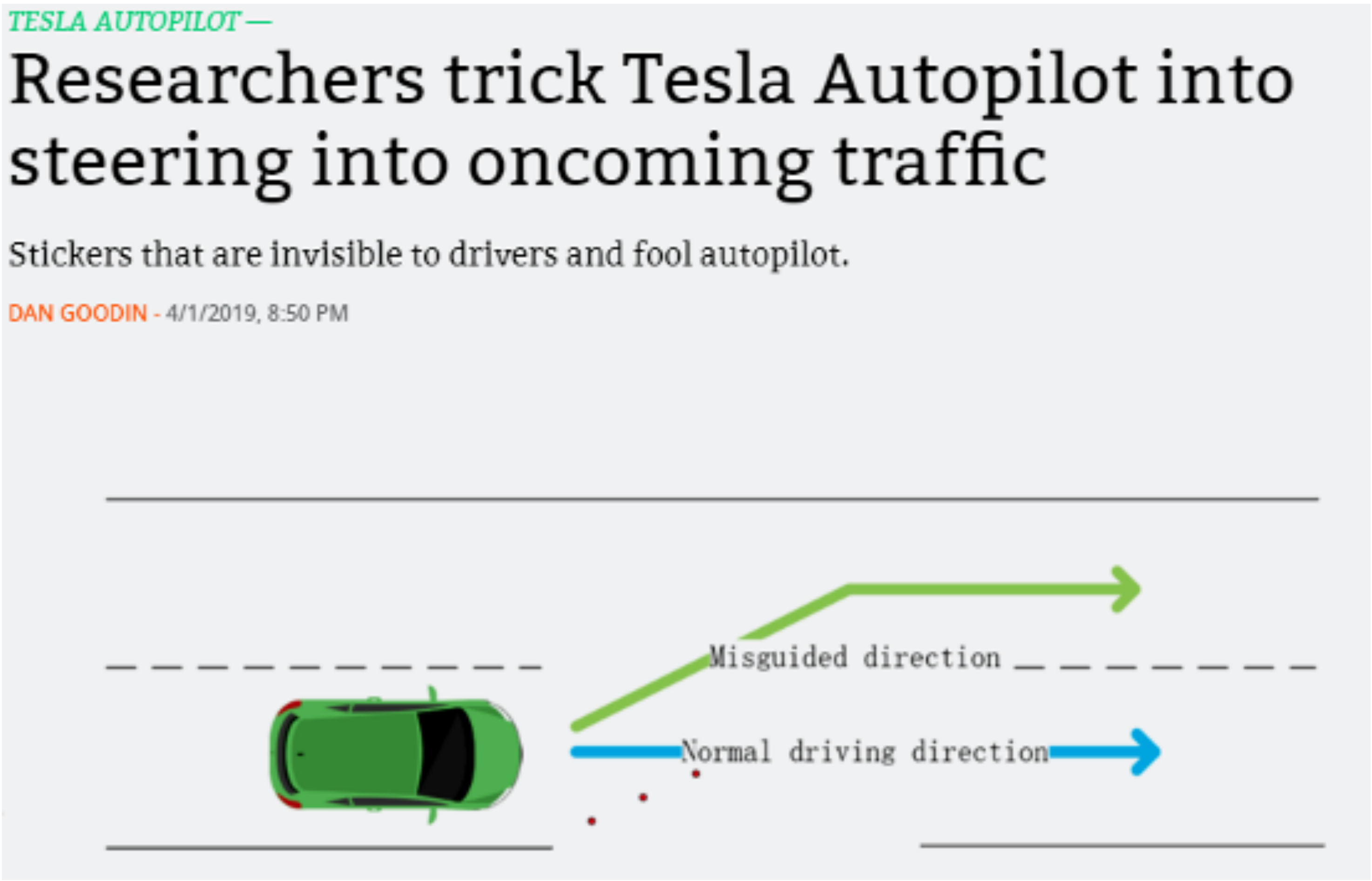


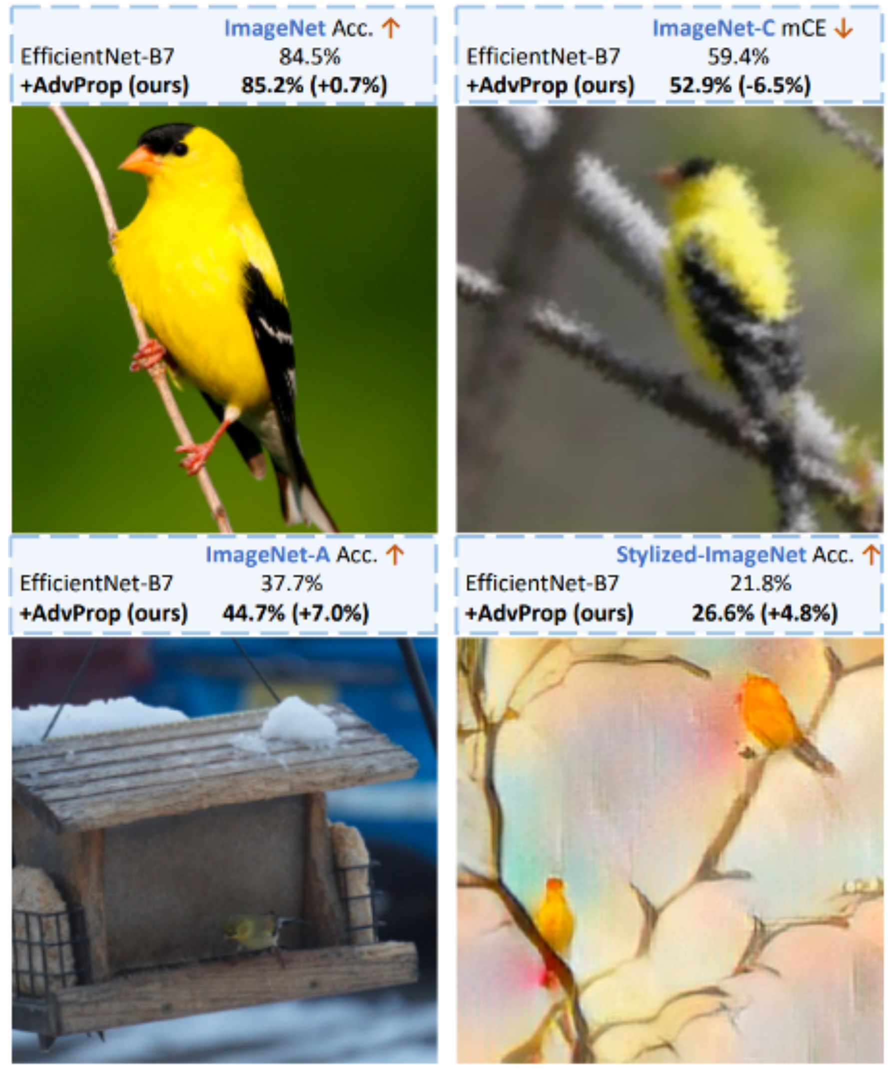
stop sign speed limit 40

Test-time integrity

Why matters

- Adversarial examples raises trustworthy and security concerns
- Critical in high-stake, safety-critical tasks
- Helps to understand the model and build a better one (SAM …)





Adversarial examples

Definition

- Given a K-way multi-class classification model $f : \mathbb{R}^d \to \{1,\ldots,K\}$ and an original example x_0, the goal is to generate an adversarial example x such that
  - x is close to x_0 and $\arg\max_i f_i(x) \neq \arg\max_i f_i(x_0)$
  - i.e., x has a different prediction with x_0 by model f.

Universal adversarial example

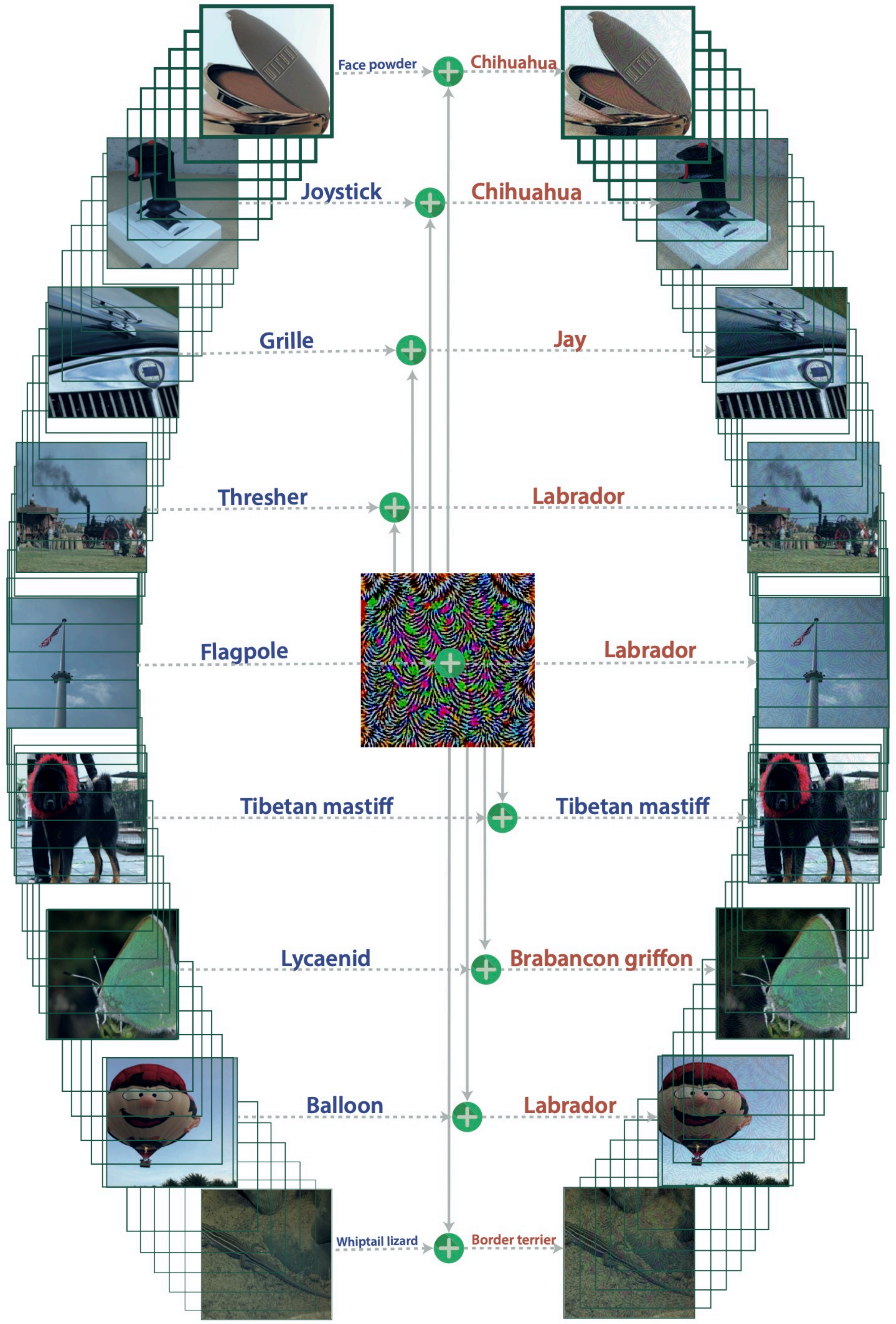
- A single perturbation that fools **almost all** tested samples

$$\hat{k}(x+v) \neq \hat{k}(x) \text{ for "most" } x \sim \mu.$$

- With two constraints

$$1.\ \|v\|_p \leq \xi,$$

$$2.\ \mathbb{P}_{x\sim\mu}\left(\hat{k}(x+v) \neq \hat{k}(x)\right) \geq 1-\delta.$$



Adversarial example

Attack as an optimization problem

- Craft adversarial example by solving
  - $\arg\min_{x} \|x - x_0\|^2 + c \cdot h(x)$
- $\|x - x_0\|^2$: the distortion

Adversarial example

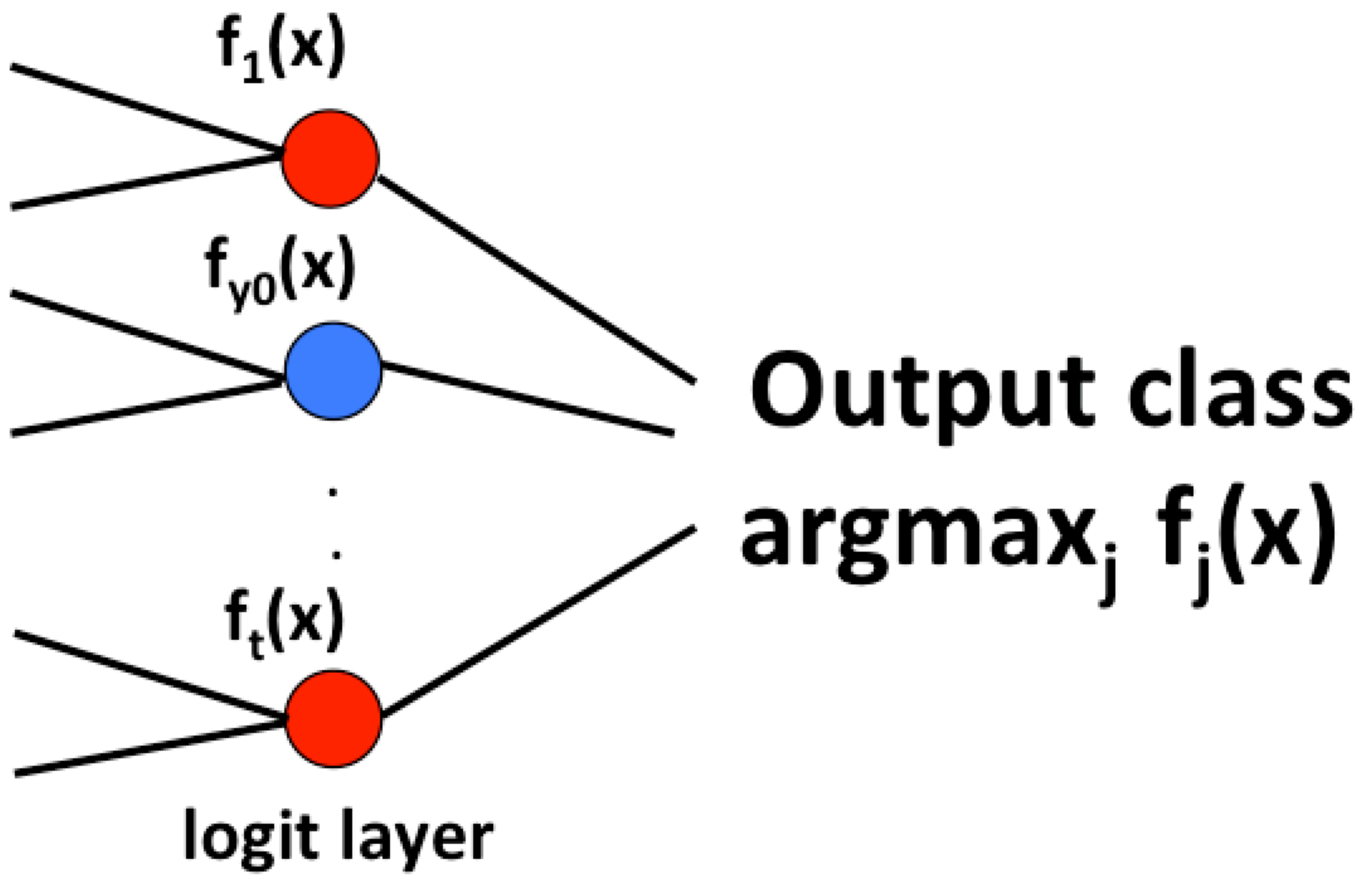
Attack as an optimization problem

- Craft adversarial example by solving
  - $\arg\min_{x} \; ||x - x_0||^2 + c \cdot h(x)$
- $||x - x_0||^2$: the distortion
- $h(x)$: loss to measure the successfulness of attack

Adversarial example

Attack as an optimization problem

- Craft adversarial example by solving
  - $\arg\min_{x} \|x - x_0\|^2 + c \cdot h(x)$
- $\|x - x_0\|^2$: the distortion
- $h(x)$: loss to measure the successfulness of attack
- Untargeted attack: success if $\arg\max_j f_j(x) \neq y_0$
  - $h(x) = \max\{f_{y_0}(x) - \max_{j \neq y_0} f_j(x), 0\}$



How to find adversarial examples

White-box vs black-box setting

- Attackers knows the model structure and weights (white-box)
- Can query the model to get probability output (soft-label)
- Can query the model to get label output (hard-label)
- No information about the model (universal)

Adversarial example

White-box setting

- $\arg\min_{x} \; \|x - x_0\|^2 + c \cdot h(x)$
- Model (network structure and weights) is revealed to attacker
  - $\Rightarrow$ gradient of $h(x)$ can be computed
  - $\Rightarrow$ attacker minimizes the objective by gradient descent

Adversarial example

White-box adversarial attack

- C&W attack [CW17]:
  - $h(x) = \max\{[Z_{y_0}(x) - \max_{j \neq y} Z_j(x)], -\kappa\}$
  - Where $Z(x)$ is the pre-softmax layer output

Adversarial example

White-box adversarial attack

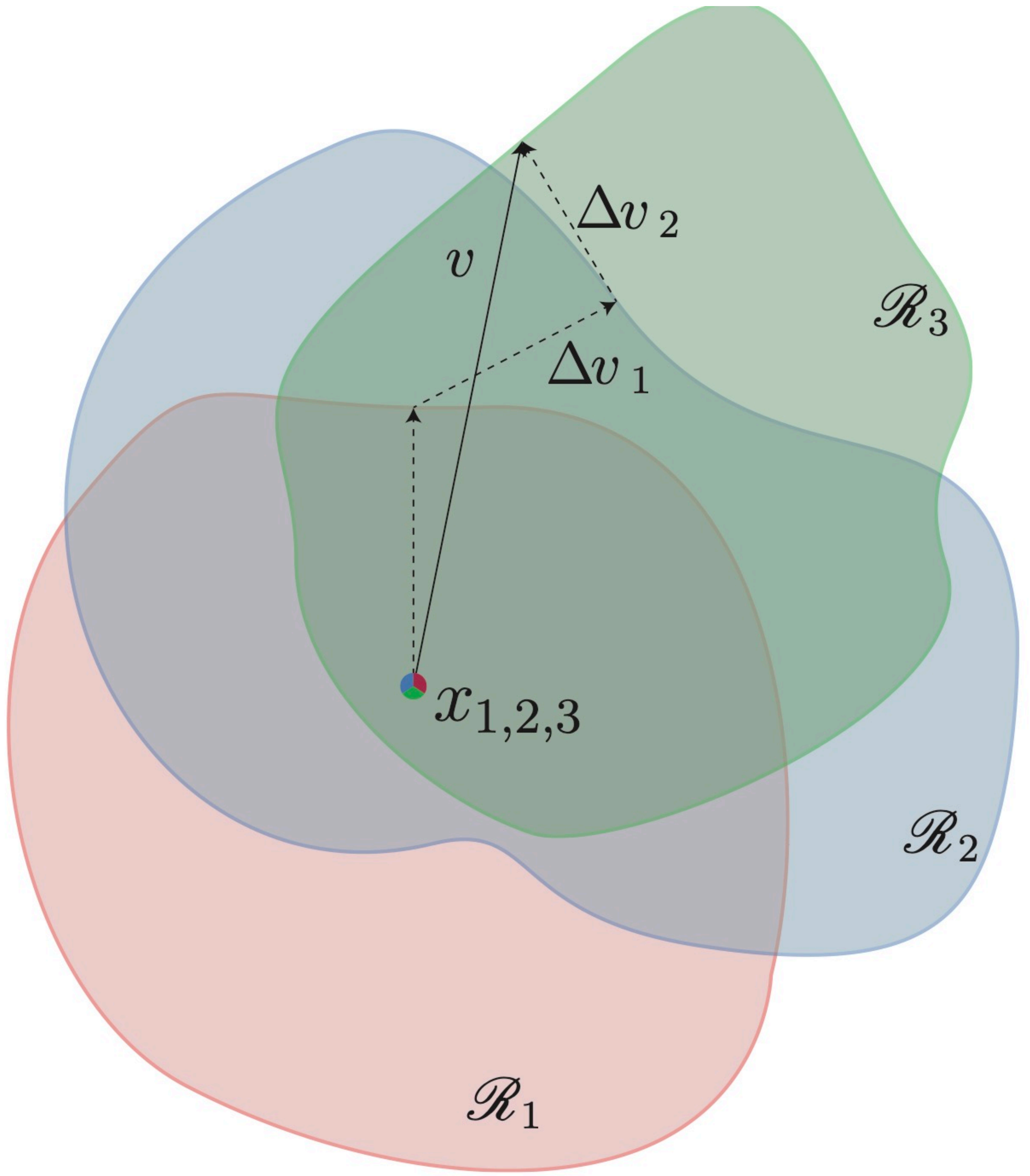
- If there is $||x - x_0||_\infty$ constraint, we could turn to solve by
- FGSM attack [GSS15]:
  - $x \leftarrow \text{proj}_{x+\mathcal{S}}(x_0 + \alpha\text{sign}(\nabla_{x_0}\ell(\theta, x, y)))$
- I FGSM attack [KGB17]
  - $x^{t+1} \leftarrow \text{proj}_{x+\mathcal{S}}(x^t + \alpha\text{sign}(\nabla_{x^t}\ell(\theta, x, y)))$

Extend to UAP

- Seek the extra perturbation by

  $$\Delta v_i \leftarrow \arg\min_r \|r\|_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i).$$

- Project to ℓ_p ball
  - $\mathcal{P}_{p,\xi}(v) = \arg\min_{v'} \|v - v'\|_2$ subject to $\|v'\|_p \leq \xi$.



Extend to UAP

- Seek the extra perturbation by

  $$\Delta v_i \leftarrow \arg\min_r \|r\|_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i).$$

- Project to ℓ_p ball
  - $\mathcal{P}_{p,\xi}(v) = \arg\min_{v'} \|v - v'\|_2$ subject to $\|v'\|_p \leq \xi$.

Algorithm 1 Computation of universal perturbations.

input: Data points X, classifier $\hat{k}$, desired ℓ_p norm of the perturbation ξ, desired accuracy on perturbed samples δ.
output: Universal perturbation vector v.
Initialize $v \leftarrow 0$.
while $\mathrm{Err}(X_v) \leq 1 - \delta$ **do**
    for each datapoint $x_i \in X$ **do**
        if $\hat{k}(x_i + v) = \hat{k}(x_i)$ **then**
            Compute the *minimal* perturbation that sends $x_i + v$ to the decision boundary:

$$\Delta v_i \leftarrow \arg\min_r \|r\|_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i).$$

            Update the perturbation:

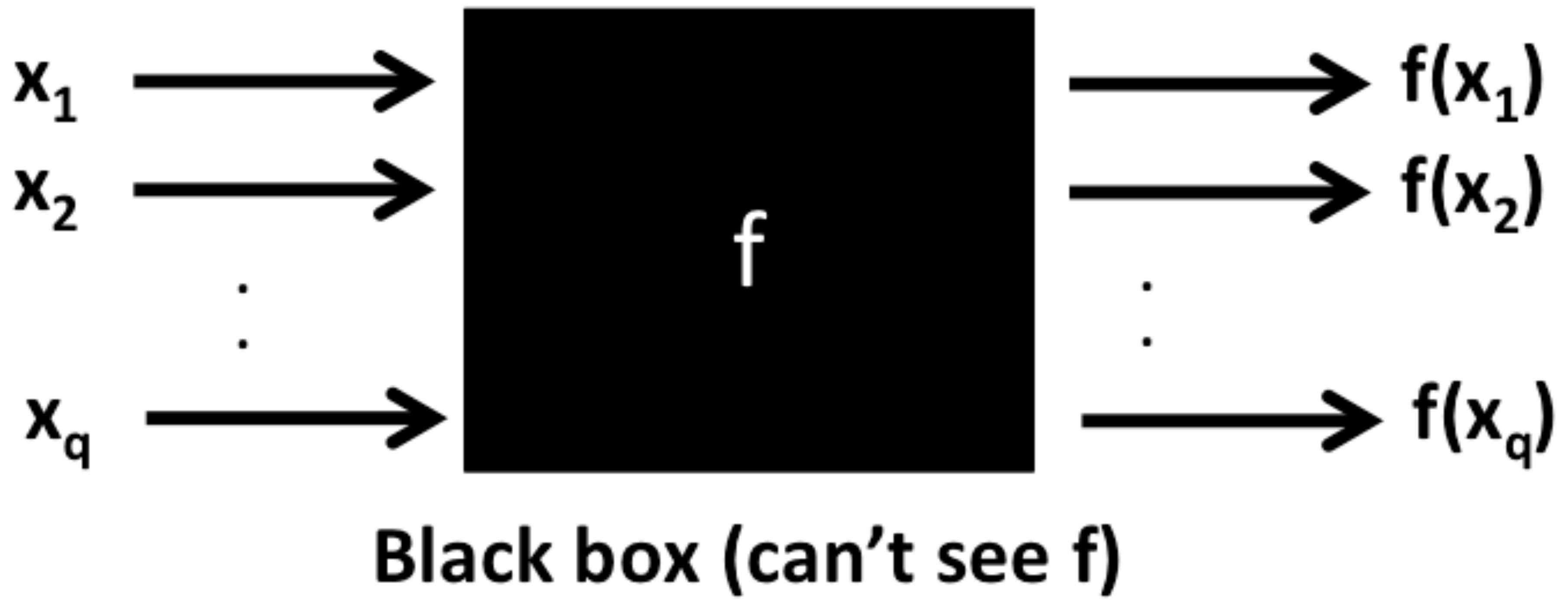
$$v \leftarrow \mathcal{P}_{p,\xi}(v + \Delta v_i).$$

        end if
    end for
end while

Adversarial example

Black-box Soft-label Setting

- Black-box Soft Label setting (practical setting):
  - Structure and weights of deep network are not revealed to attackers
  - Attacker can query the ML model and get the probability output



- Cannot compute gradient ∇_x

Adversarial attack

Soft-label Black-box Adversarial attack

- Soft-label Black-box: query to get the **probability output**
- Key problem: how to estimate gradient?
- Gradient-based [CZS17,IEAL18]:
- $\nabla_x = \dfrac{h(x + \beta u) - h(x)}{\beta} \cdot u$
- Genetic algorithm [ASC19]

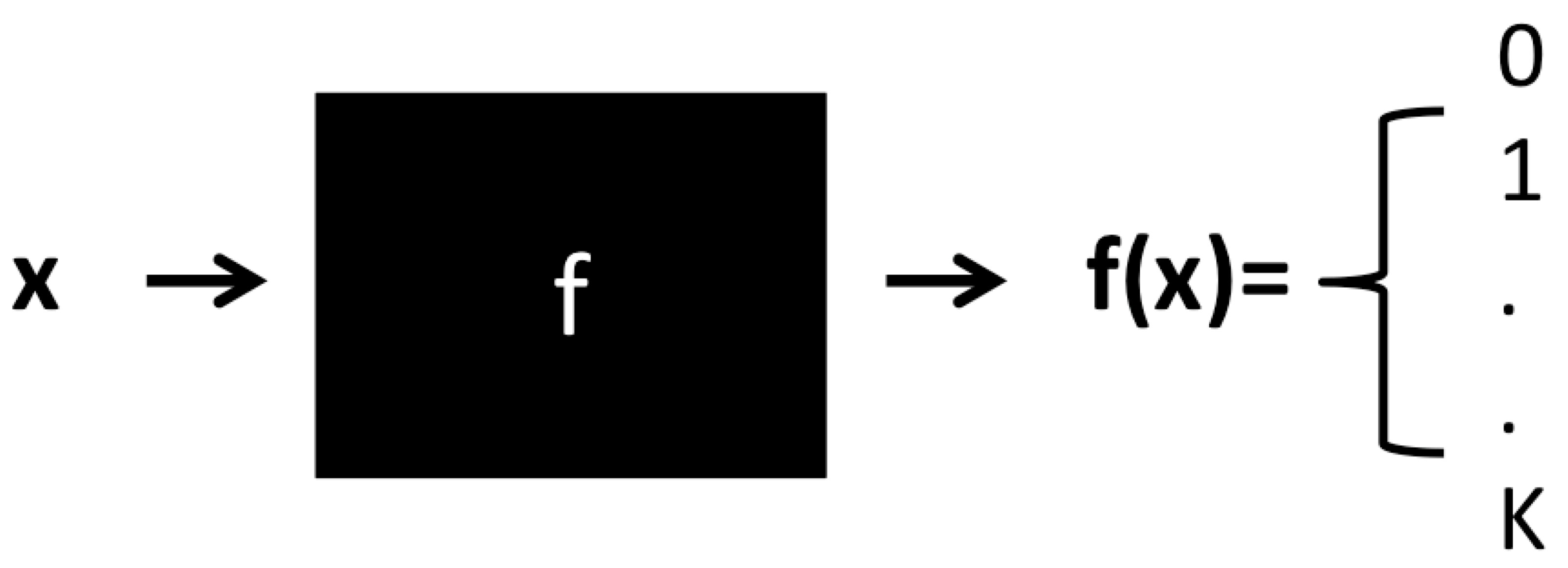
Soft-label Black-box Adversarial attack

- Transfer based:
  - Train a substitute model to mimic the black-box model
  - Attack the substitute model by white-box attack

Adversarial attack

Hard-label Black-box Attack

- Model is not known to the attacker
- Attacker can make query and observe hard-label multi-class output

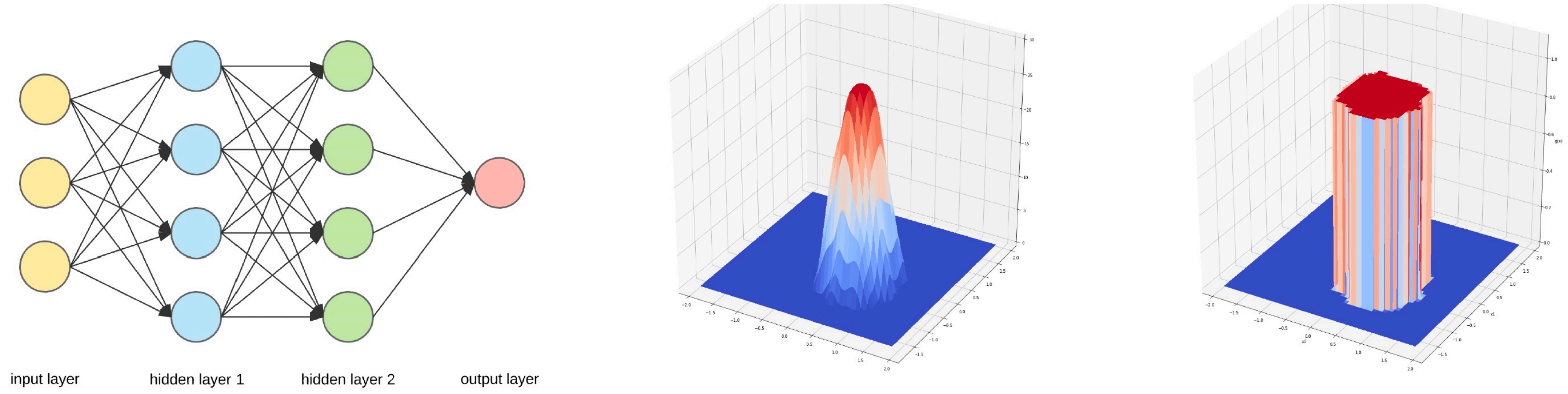


- (K: number of classes)
- More practical setting for attacker
- Discrete and complex models (e.g quantization, projection, detection)
- Framework friendly

Hard-label black-box attack

The difficulty

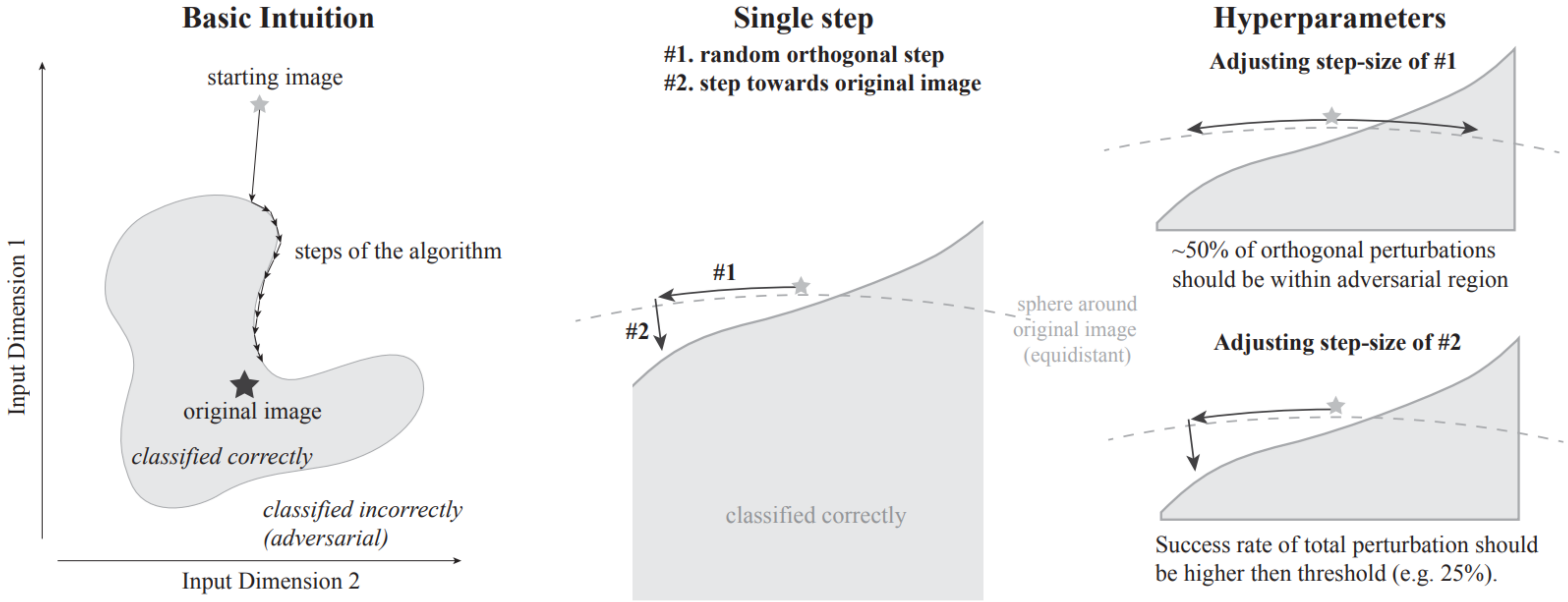
- Hard-label attack on a simple 3-layer neural network yields a discontinuous optimization problem



(a) neural network $f(x)$ (b) $h(Z(x))$ (c) $h(f(x))$

Hard-label black-box attack

Boundary attack: based on random walk



Hard-label black-box attack

Boundary attack: based on random walk

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

Boundary attack

What P to use?

1. The perturbed sample lies within the input domain,
$$\tilde{o}_i^{k-1} + \eta_i^k \in [0, 255]. \tag{1}$$
2. The perturbation has a relative size of δ,
$$\left\|\boldsymbol{\eta}^k\right\|_2 = \delta \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}}). \tag{2}$$
3. The perturbation reduces the distance of the perturbed image towards the original input by a relative amount ϵ,
$$d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}}) - d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}} + \boldsymbol{\eta}^k) = \epsilon \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}}). \tag{3}$$

Hard-label black-box attack

What P to use?

1. The perturbed sample lies within the input domain,
$$\tilde{o}_i^{k-1} + \eta_i^k \in [0, 255]. \tag{1}$$
2. The perturbation has a relative size of δ,
$$\left\| \boldsymbol{\eta}^k \right\|_2 = \delta \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}}). \tag{2}$$
3. The perturbation reduces the distance of the perturbed image towards the original input by a relative amount ϵ,
$$d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}}) - d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}} + \boldsymbol{\eta}^k) = \epsilon \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{\mathbf{k-1}}). \tag{3}$$

Hotskipjump attack

Formalization

- Turn it into optimization

$$\min_{x'} d(x', x^\star) \quad \text{such that} \quad \phi_{x^\star}(x') = 1.$$

- Where $\phi_{x^\star}(x') := \operatorname{sign}\left(S_{x^\star}(x')\right) \; = \begin{cases} 1 & \text{if } S_{x^\star}(x') > 0, \\ -1 & \text{otherwise.} \end{cases}$

$$S_{x^\star}(x') := \begin{cases} \max\limits_{c \neq c^\star} F_c(x') - F_{c^\star}(x') & \text{(Untargeted)} \\ F_{c^\dagger}(x') - \max\limits_{c \neq c^\dagger} F_c(x') & \text{(Targeted)} \end{cases}$$

Hotskipjump attack

Formalization

- Turn it into optimization

$$\min_{x'} d(x', x^\star) \quad \text{such that} \quad \phi_{x^\star}(x') = 1.$$

- Where $\phi_{x^\star}(x') := \operatorname{sign}\left(S_{x^\star}(x')\right) \;=\; \begin{cases} 1 & \text{if } S_{x^\star}(x') > 0, \\ -1 & \text{otherwise.} \end{cases}$

$$S_{x^\star}(x') := \begin{cases} \max\limits_{c \neq c^\star} F_c(x') - F_{c^\star}(x') & \text{(Untargeted)} \\ F_{c^\dagger}(x') - \max\limits_{c \neq c^\dagger} F_c(x') & \text{(Targeted)} \end{cases}$$

Hotskipjump attack

Solve the optimization

- In the hard-label setting, we only have $\phi_{x^\star}(x) = \mathrm{sign}(S_{x^\star}(x))$.
- Given $x_t \in \mathrm{bd}(S_{x^\star})$, approximate the gradient by $\nabla S_{x^\star}(x_t)$

$$\widetilde{\nabla S}(x_t, \delta) := \frac{1}{B} \sum_{b=1}^{B} \phi_{x^\star}(x_t + \delta u_b) u_b,$$

- Where $\{u_b\}_{b=1}^{B}$ are i.i.d. draws from the uniform distribution
- How to get to x_t?

Hotskipjump attack

Solve the optimization

- Approach the boundary via binary search

$$\tilde{x}_t := x_t + \xi_t v_t(x_t, \delta_t), \text{ such that}$$
$$v_t(x_t, \delta_t) = \begin{cases} \widehat{\nabla S}(x_t, \delta_t) / \|\widehat{\nabla S}(x_t, \delta_t)\|_2, & \text{if } p = 2, \\ \text{sign}(\widehat{\nabla S}(x_t, \delta_t)), & \text{if } p = \infty, \end{cases}$$

- Correct with variance reduction

$$\widehat{\nabla S}(x_t, \delta) := \frac{1}{B-1} \sum_{b=1}^{B} (\phi_{x^\star}(x_t + \delta u_b) - \overline{\phi_{x^\star}}) u_b \qquad \overline{\phi_{x^\star}} := \frac{1}{B} \sum_{b=1}^{B} \phi_{x^\star}(x_t + \delta u_b),$$

Hotskipjump attack
Overview

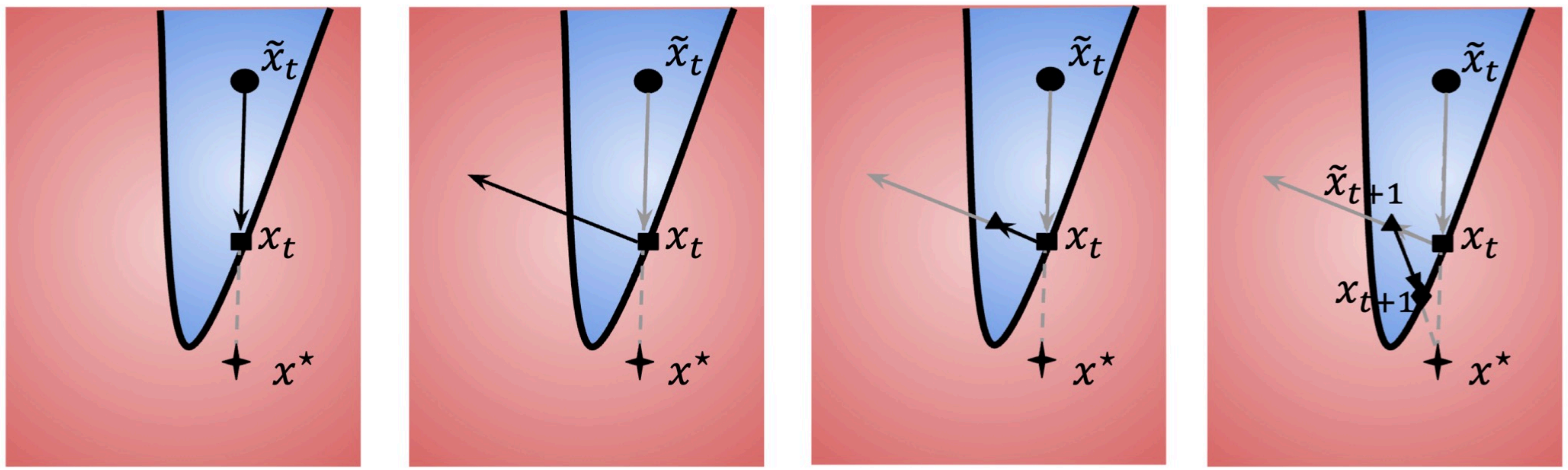
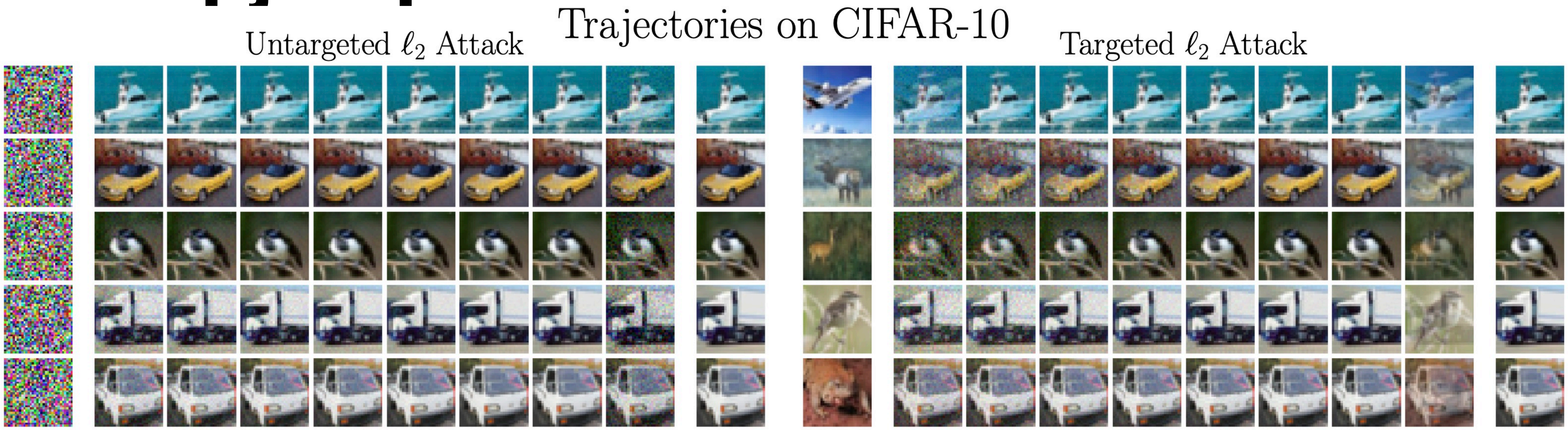
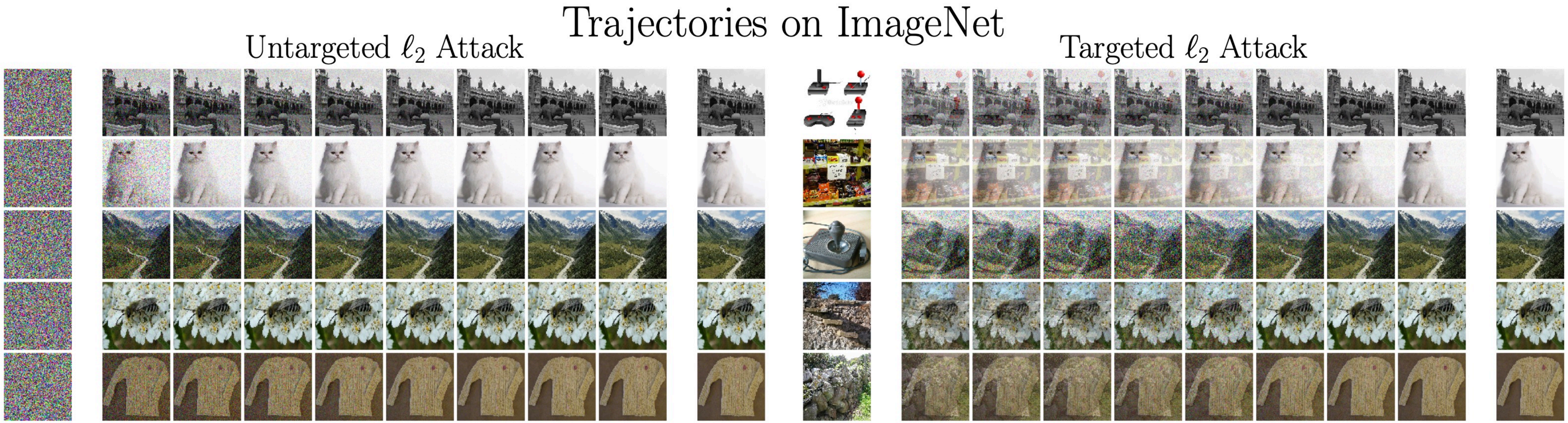


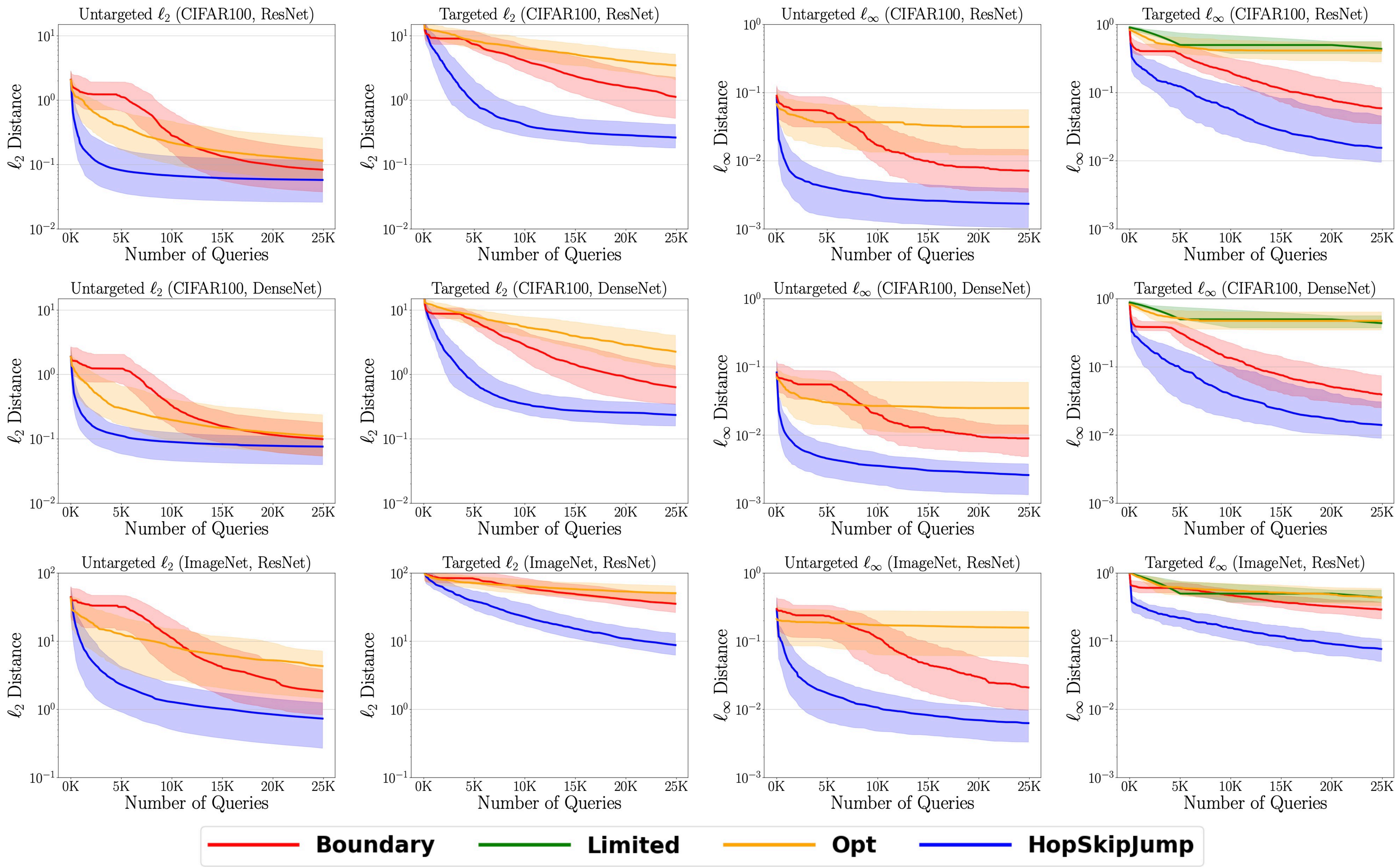
Figure 2: Intuitive explanation of HopSkipJumpAttack. (a) Perform a binary search to find the boundary, and then update $\tilde{x}_t \rightarrow x_t$. (b) Estimate the gradient at the boundary point x_t. (c) Geometric progression and then update $x_t \rightarrow \tilde{x}_{t+1}$. (d) Perform a binary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

Hotskipjump attack





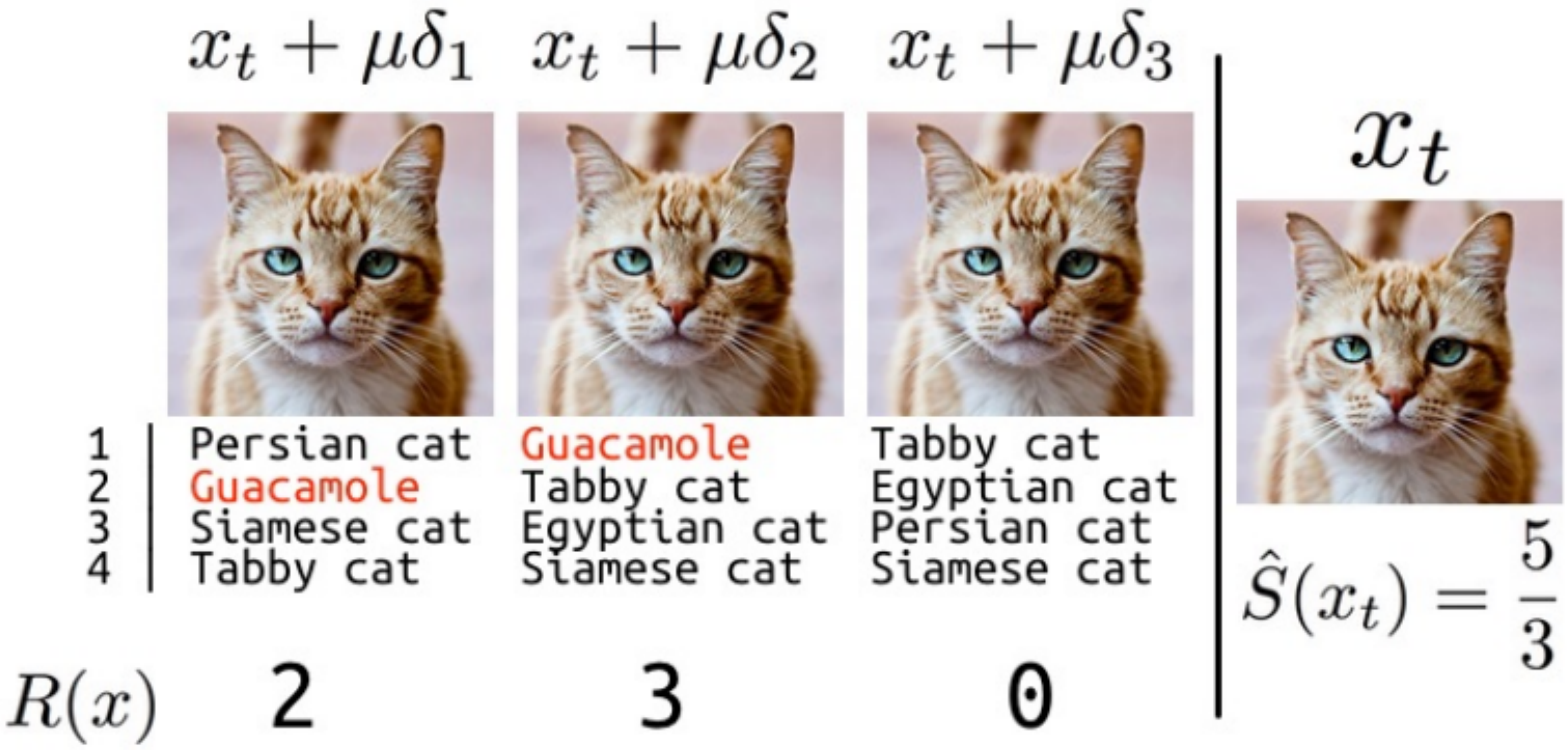
Hotskipjump attack



Hard-label black-box attack

Limited attack

- Limited Attack: Monte Carlo method to get the probability output



Hard-label black-box attack

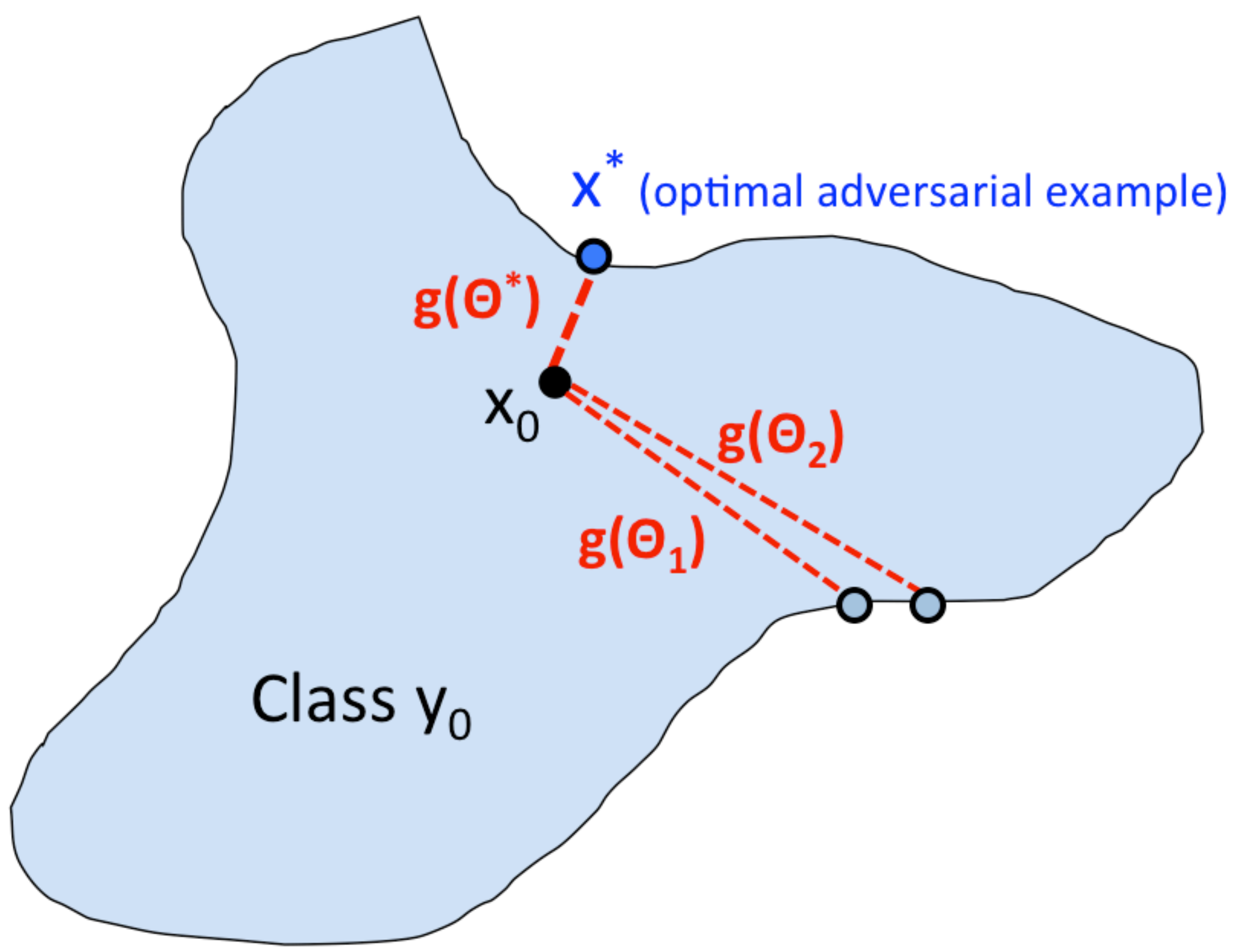
OPT-attack

- We reformulate the attack optimization problem (untargeted attack):

$$\theta^* = \arg\min_{\theta} \quad g(\theta)$$

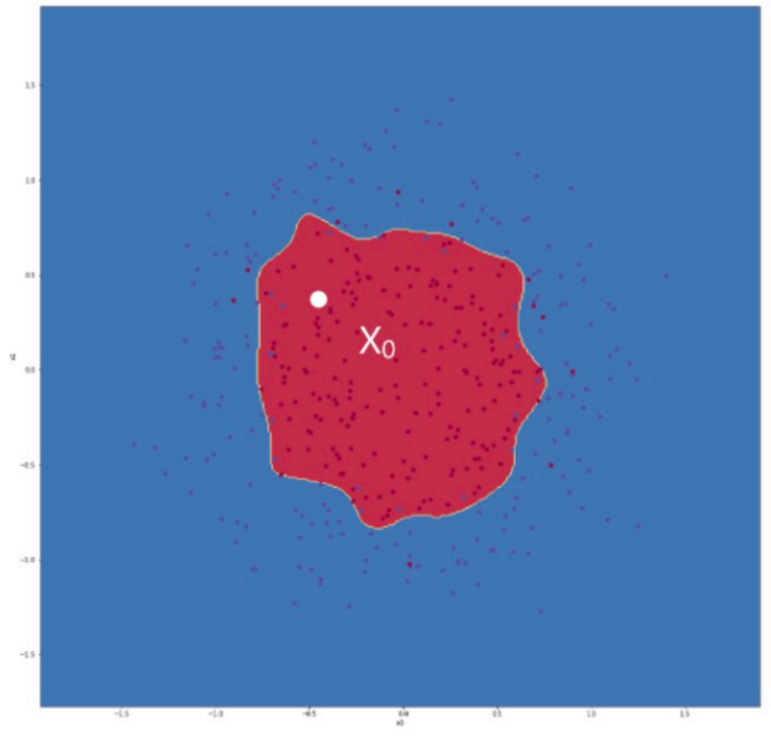
- where $g(\theta) = \text{argmin}_{\lambda>0}\left(f(x_0 + \lambda\frac{\theta}{||\theta||}) \neq y_0\right)$

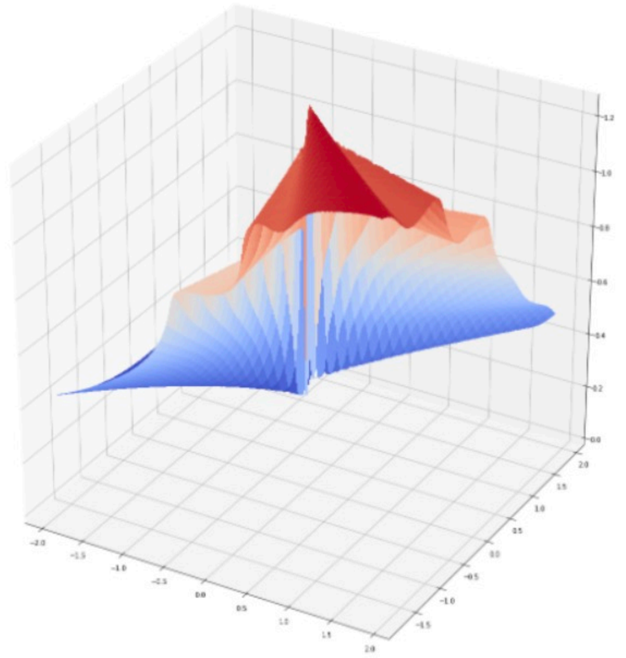
- θ: the direction of adversarial example



OPT-attack

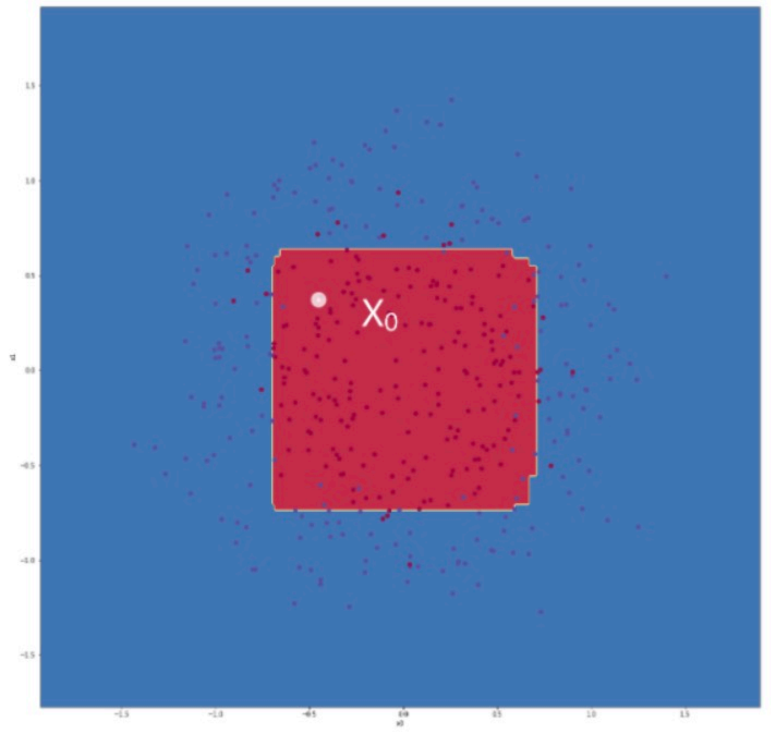
Examples

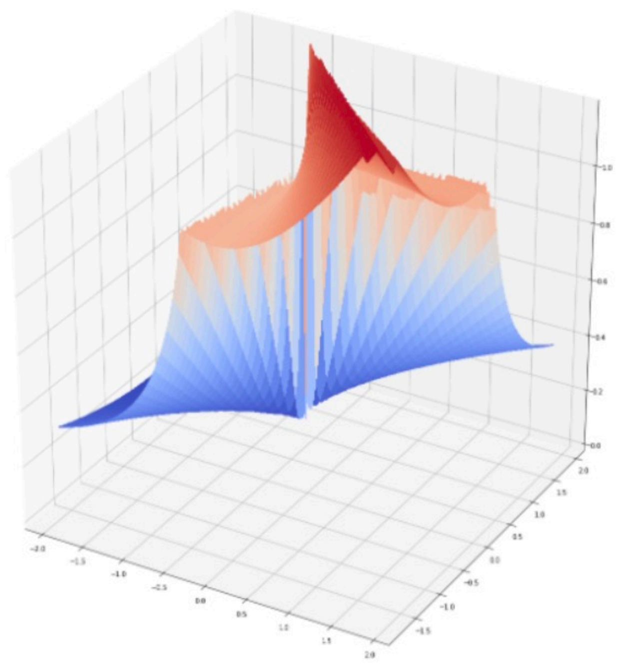




Neural network decision function

$g(\boldsymbol{\theta})$





Boosting Tree decision function

$g(\boldsymbol{\theta})$

OPT-attack

Two things unaddressed

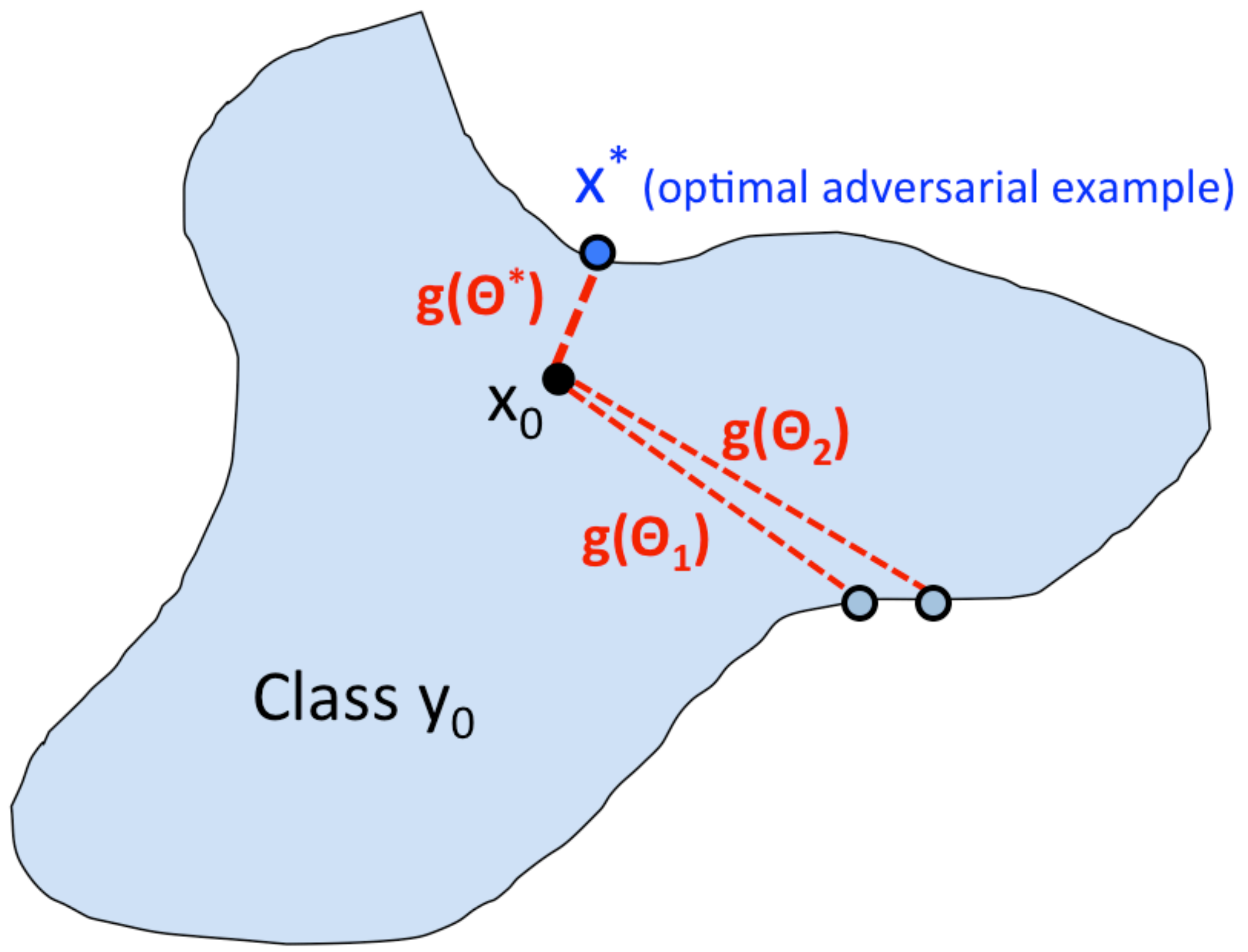
$$\theta^* = \arg\min_{\theta} \quad g(\theta)$$

- where $g(\theta) = \text{argmin}_{\lambda>0}\left(f(x_0 + \lambda\frac{\theta}{||\theta||}) \neq y_0\right)$
- How to estimate $g(\theta)$
- How to find θ^*

OPT-attack

Computing Function Value

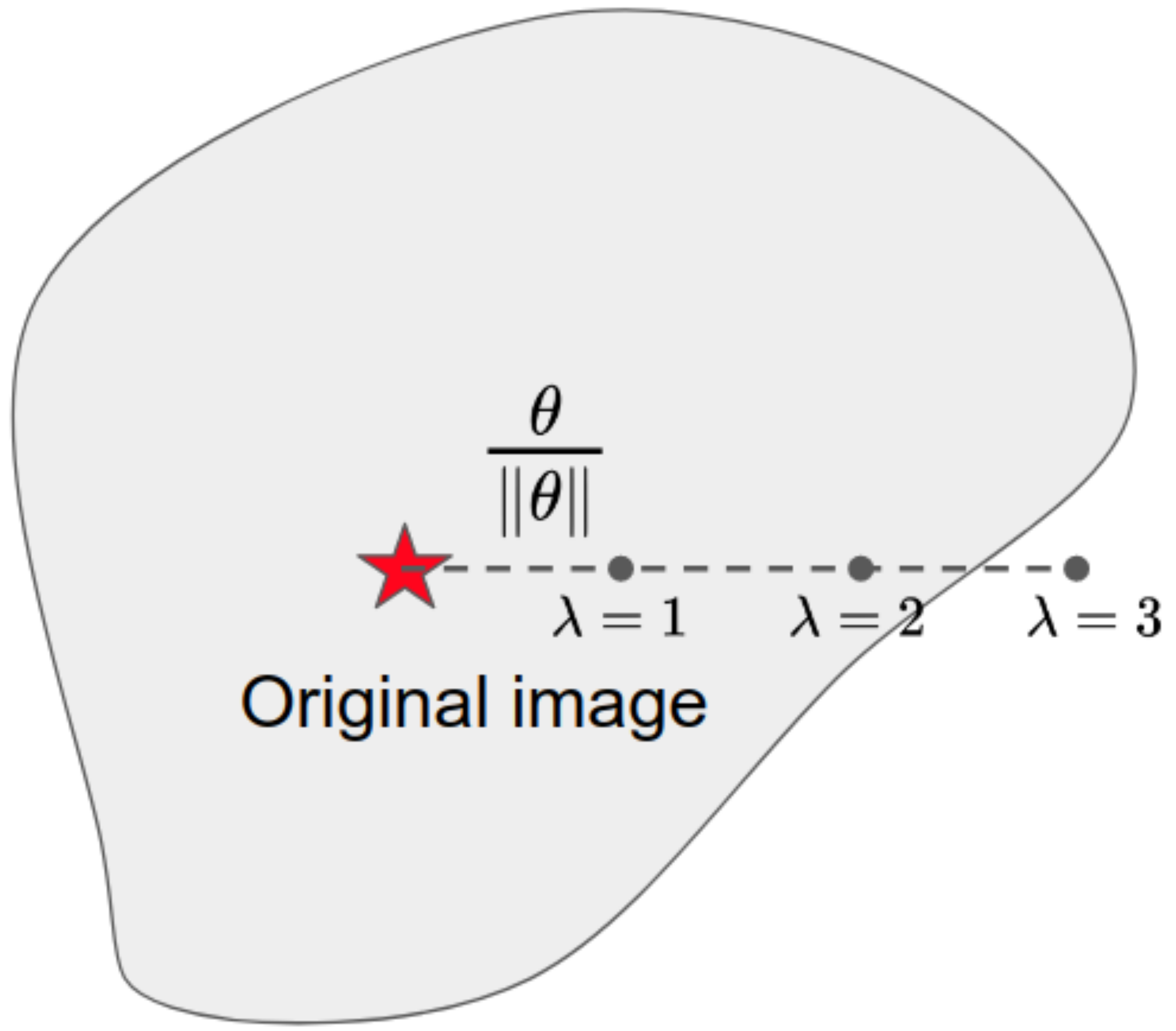
- Can't compute the gradient of g
- However, we can compute the function value of g using queries of $f(\cdot)$
- Implemented using fine-grained search + binary search



OPT-attack

Estimation of $g(\theta)$

- Fine-grained search
- Binary search
  - Prediction unchanged enlarge g
  - Prediction changed shrink g



How to optimize $g(\theta)$

- The gradient of g is available by
- $\nabla g(\theta) \approx \frac{g(\theta + \beta u) - g(\theta)}{\beta} \cdot u$
- One u is too noisy, better to use multiple u (~ 20)
- Zeroth order optimization for minimizing $g(\theta)$

Algorithm

Algorithm 1 OPT attack (ICLR '19)

1: **Input:** Hard-label model f, original image x_0, initial $\boldsymbol{\theta}_0$.
2: **for** $t = 0, 1, 2, \ldots, T$ **do**
3: Randomly choose $\boldsymbol{u}$ from a zero-mean Gaussian distribution
4: Evaluate $g(\boldsymbol{\theta}_t)$ and $g(\boldsymbol{\theta}_t + \beta \boldsymbol{u})$
5: Compute $\hat{\boldsymbol{g}} = \frac{g(\boldsymbol{\theta}_t + \beta \boldsymbol{u}) - g(\boldsymbol{\theta}_t)}{\beta} \cdot \boldsymbol{u}$
6: Update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \hat{\boldsymbol{g}}$
7: **return** $x_0 + g(\boldsymbol{\theta}_T)\boldsymbol{\theta}_T$

Algorithm

Algorithm 2 OPT attack (ICLR '19)

Input: Hard-label model f, original image x_0, initial $\boldsymbol{\theta}_0$.
for $t = 0, 1, 2, \ldots, T$ **do**
  Randomly choose $\boldsymbol{u}_t$ from a zero-mean Gaussian distribution
  Evaluate $g(\boldsymbol{\theta}_t)$ and $g(\boldsymbol{\theta}_t + \beta \boldsymbol{u})$
  Compute $\hat{\mathbf{g}} = \frac{g(\boldsymbol{\theta}_t + \beta \boldsymbol{u}) - g(\boldsymbol{\theta}_t)}{\beta} \cdot \boldsymbol{u}$
  Update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \hat{\mathbf{g}}$
return $x_0 + g(\boldsymbol{\theta}_T)\boldsymbol{\theta}_T$

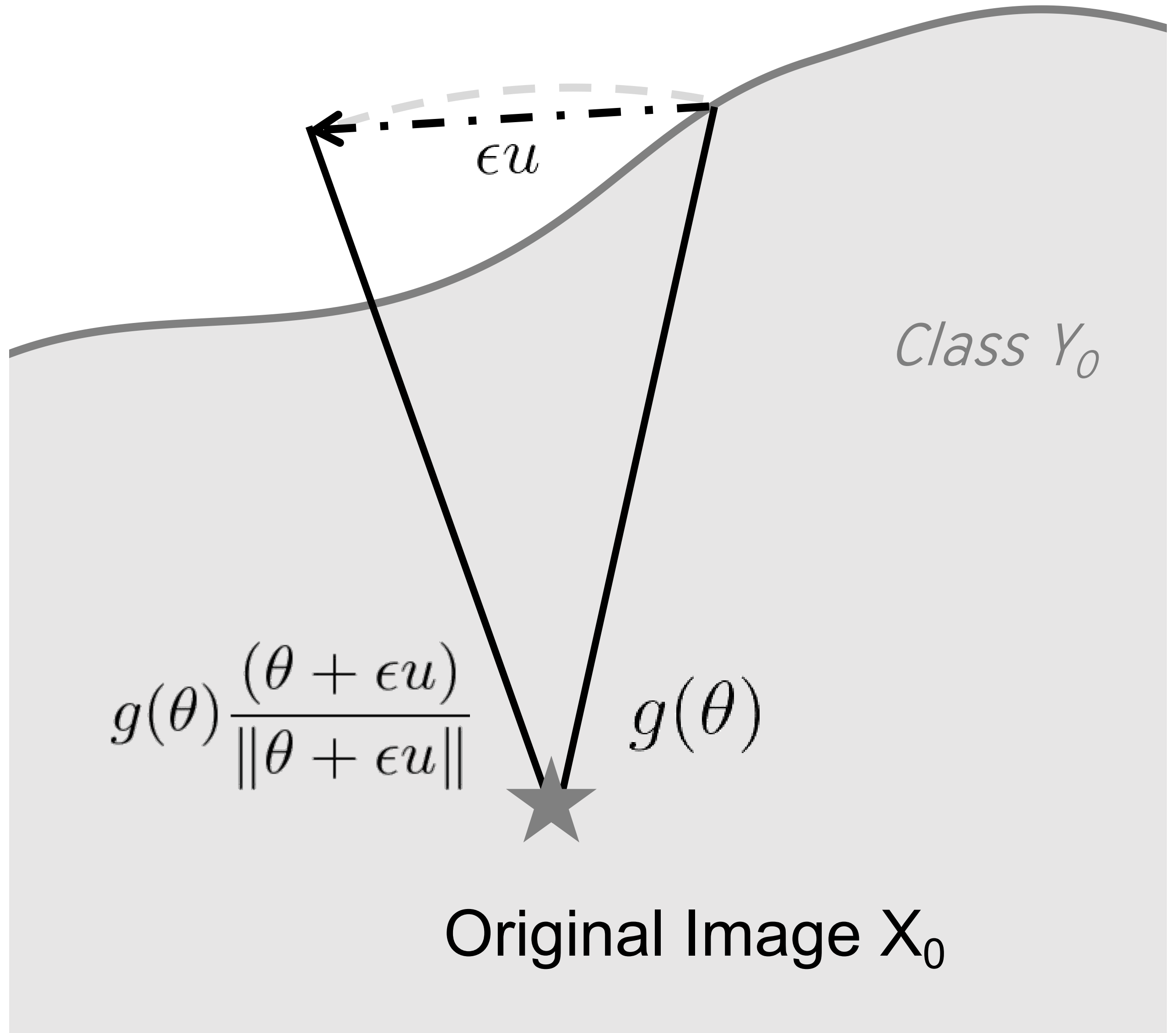
- $g(\theta_t)$ and $g(\theta_t + \beta u)$ in the gradient estimation takes most of queries, how to further reduce it?

Sign is enough!

- Binary search to estimate $g(\theta)$ in the gradient estimation takes most of queries.
- Gradient sign is powerful ! (FGSM)
- How to get the gradient sign efficiently ?

Single query oracle

- $$\text{sign}(g(\theta+\epsilon u)-g(\theta)) = \begin{cases} +1, & f(x_0+g(\theta)\frac{(\theta+\epsilon u)}{\|\theta+\epsilon u\|})=y_0, \\ -1, & \text{Otherwise.} \end{cases}$$



Sign-OPT attack

Algorithm 3 Sign-OPT attack (ICLR '20)

Input: Hard-label model f, original image x_0, initial $\boldsymbol{\theta}_0$
for $t = 1, 2, \ldots, T$ **do**
  Randomly sample $u_1, \ldots, u_Q$ from a Gaussian or Uniform distribution
  Evaluate $g(\boldsymbol{\theta}_t)$
  $\hat{\mathbf{g}} = \cancel{\frac{g(\boldsymbol{\theta}_t+\beta \boldsymbol{u})-g(\boldsymbol{\theta}_t)}{\beta}} \cdot \boldsymbol{u} \Rightarrow \text{sign}(\frac{g(\boldsymbol{\theta}_t+\beta \boldsymbol{u})-g(\boldsymbol{\theta}_t)}{\beta}) \cdot \boldsymbol{u}$
  Update $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta \hat{\mathbf{g}}$
  Evaluate $g(\boldsymbol{\theta}_t)$ using the same search algorithm

Results

Qualitative evaluation

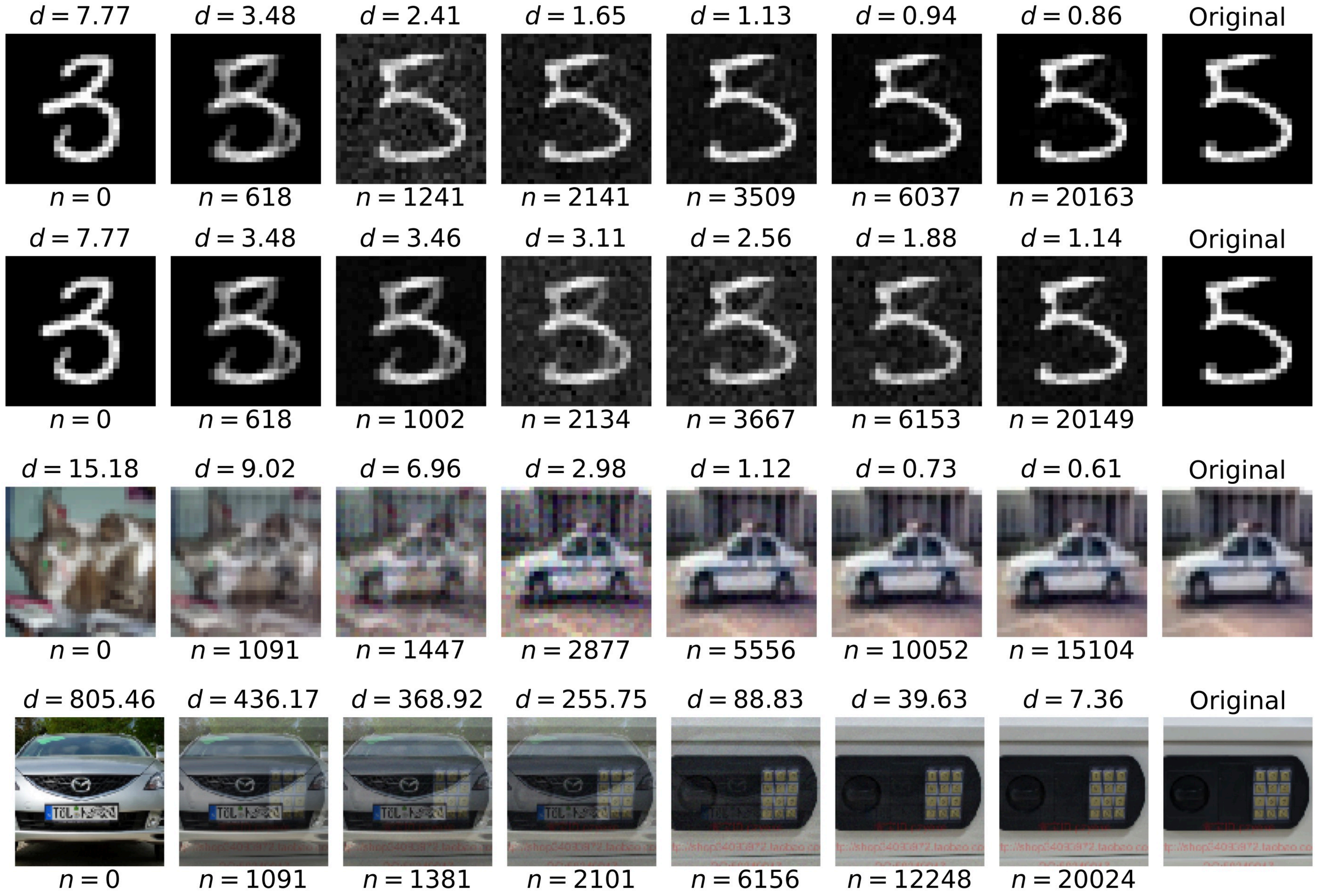


Figure 2: Example of Sign-OPT targeted attack. L_2 distortions and queries used are shown above and below the images. First two rows: Example comparison of Sign-OPT attack and OPT attack. Third and fourth rows: Examples of Sign-OPT attack on CIFAR-10 and ImageNet

Results

Quantitive evaluation

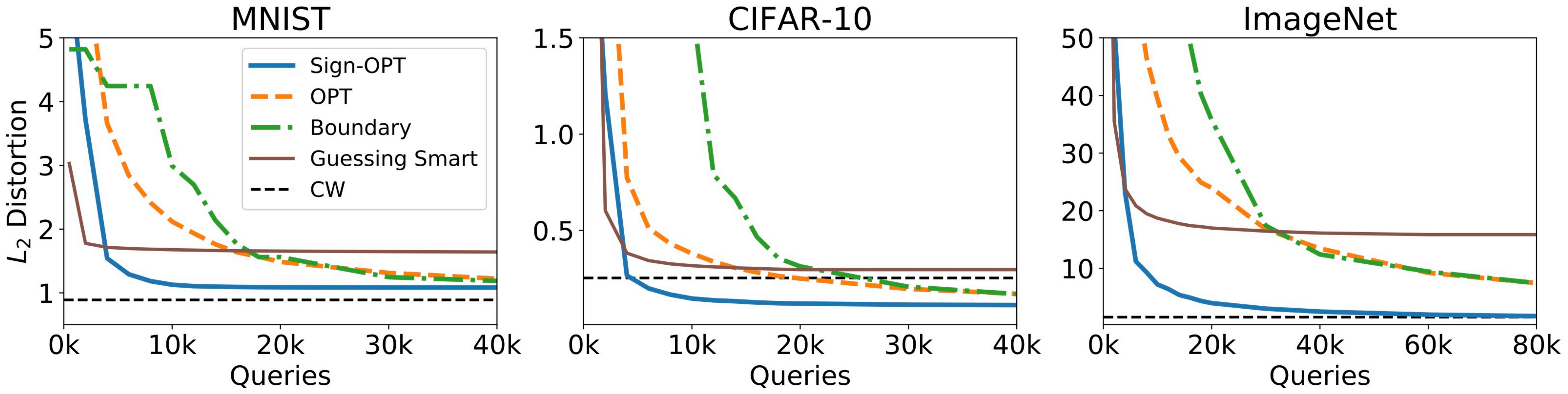
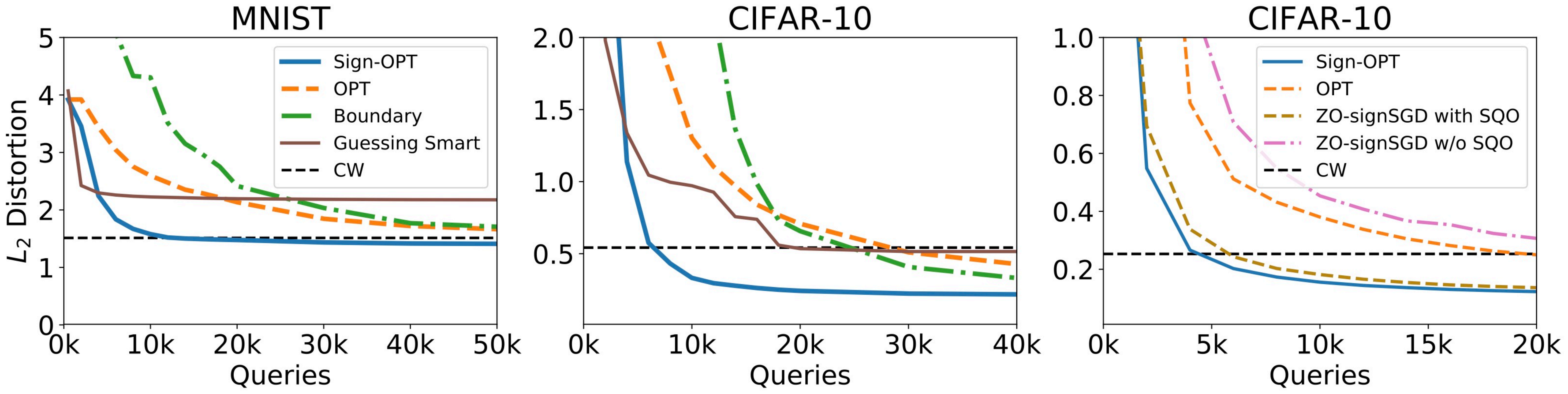


Figure 4: Untargeted attack: Median distortion vs Queries for different datasets.



(a) (b)

Figure 5: (a) Targeted Attack: Median distortion vs Queries of different attacks on MNIST and CIFAR-10. (b) Comparing Sign-OPT and ZO-SignSGD with and without single query oracle (SQO).

Results

Quantitive evaluation

	MNIST			CIFAR10			ImageNet (ResNet-50)		
	#Queries	Avg L_2	SR($\epsilon = 1.5$)	#Queries	Avg L_2	SR($\epsilon = 0.5$)	#Queries	Avg L_2	SR($\epsilon = 3.0$)
Boundary attack	4,000	4.24	1.0%	4,000	3.12	2.3%	4,000	209.63	0%
	8,000	4.24	1.0%	8,000	2.84	7.6%	30,000	17.40	16.6%
	14,000	2.13	16.3%	12,000	0.78	29.2%	160,000	4.62	41.6%
OPT attack	4,000	3.65	3.0%	4,000	0.77	37.0%	4,000	83.85	2.0%
	8,000	2.41	18.0%	8,000	0.43	53.0%	30,000	16.77	14.0%
	14,000	1.76	36.0%	12,000	0.33	61.0%	160,000	4.27	34.0%
Guessing Smart	4,000	1.74	41.0%	4,000	0.29	75.0%	4,000	16.69	12.0%
	8,000	1.69	42.0%	8,000	0.25	80.0%	30,000	13.27	12.0%
	14,000	1.68	43.0%	12,000	0.24	80.0%	160,000	12.88	12.0%
Sign-OPT attack	4,000	1.54	46.0%	4,000	0.26	73.0%	4,000	23.19	8.0%
	8,000	1.18	84.0%	8,000	0.16	90.0%	30,000	2.99	50.0%
	14,000	1.09	94.0%	12,000	0.13	95.0%	160,000	1.21	90.0%
C&W (white-box)	-	0.88	99.0%	-	0.25	85.0%	-	1.51	80.0%

Evaluating test-time integrity

Other Domains

- Evaluating test-time integrity on text classification model
- Evaluating test-time integrity on seq2seq model
- Evaluating test-time integrity on dialog system

Source input seq	A child is splashing in the water.
Adv input seq	A children is **unionists** in the water.
Source output seq	Ein Kind im Wasser.
Adv output seq	**Kinder sind in der Wasser @-@ <unk>**.
Source input seq	Two men wearing swim trunks jump in the air at a moderately populated beach.
Adv input seq	Two men wearing **dog Leon comes** in the air at a moderately populated beach.
Source output seq	Zwei Mnner in Badehosen springen auf einem mig belebten Strand in die Luft.
Adv output seq	Zwei Mnner tragen **Hund** , der in der Luft **sitzt** , hat <unk> <unk> .

Input		
Adv agent	1x**book** value 1 4x**hat** value 1 1x**ball** value 5	
RL agent	1x**book** value 2 4x**hat** value 1 1x**ball** value 4	
Adv agent	i want the hats and 2 balls	
RL agent	i need the balls and the hat	
Adv agent	take book you get rest	
RL agent	deal	
Adv agent	$\langle selection \rangle$	
Output		**Reward**
Adv agent	4x**hat** 1x**ball**	9/10
RL agent	1x**book**	2/10