University of Idaho
Department of Computer Science

CS 487/587
Adversarial Machine Learning

Dr. Alex Vakanski

Lecture 5

Evasion Attacks against Black-box Machine Learning Models

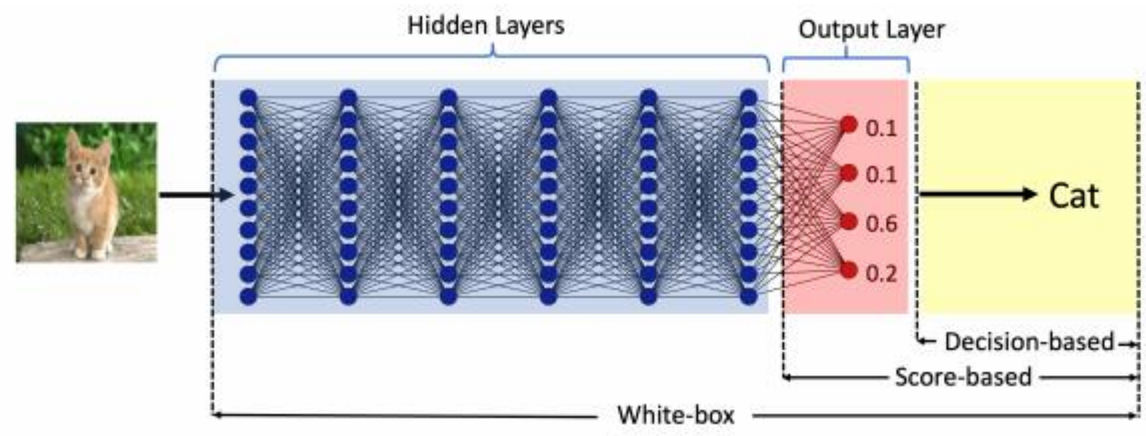
Lecture Outline

- Bhagoji et al. (2017) Exploring the Space of Black-box Attacks on Deep Neural Networks
- Brendel et al. (2018) Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models
- Transferability in Adversarial Machine Learning
  - Substitute model attack
  - Ensemble of local models attack
- Other black-box evasion attacks
  - HopSkipJump attack
  - ZOO attack
  - Simple black-box attack

Evasion Attacks against Black-box Models

Black-box Evasion Attacks

- Black-box adversarial attacks can be classified into two categories:
  - *Query-based attacks*
    - The adversary queries the model and creates adversarial examples by using the provided information to queries
    - The queried model can provide:
      - Output class probabilities (i.e., confidence scores per class) used with score-based attacks
      - Output class, used with decision-based attacks
  - *Transfer-based attacks* (or *transferability attacks*)
    - The adversary does not query the model
    - The adversary trains its own substitute/surrogate local model, and transfers the adversarial examples to the target model
    - This type of approaches are also referred to as zero queries attacks



Gradient Estimation Attack

- ***Bhagoji, He, Li, Song (2017) Exploring the Space of Black-box Attacks on Deep Neural Networks***
- The paper introduces an approach known as ***Gradient Estimation attack***
- Score-based black-box attack
  - Based on query access to the model's class probabilities
  - Both targeted and untargeted attacks are achieved
- Validated on MNIST and CIFAR-10 datasets
  - The attack is also evaluated on real-world models hosted by Clarifai
- Advantages:
  - Outperformed other black-box attacks
  - Performance results are comparable to white-box attacks
  - Good results against adversarial defenses

Gradient Estimation Attack

- Gradient Estimation (GE) approach
  - Uses queries to directly estimate the gradient and carry out black-box attacks
  - The output to a query is the vector of class probabilities $\mathbf{p}^f(\mathbf{x})$ (i.e., confidence scores per class) for an input $\mathbf{x}$
    - The logits can also be recovered from the probabilities, by taking $\log\left(\mathbf{p}^f(\mathbf{x})\right)$
- The authors employed the method of finite differences for gradient estimation
  - Let $g(\mathbf{x})$ is a function whose gradient needs to be estimated
  - Finite difference (FD) estimation of the gradient of g with respect to input $\mathbf{x}$ is given by

$$\mathrm{FD}_{\mathbf{x}}(g(\mathbf{x}), \delta) = \begin{bmatrix} \frac{g(\mathbf{x}+\delta\mathbf{e}_1)-g(\mathbf{x}-\delta\mathbf{e}_1)}{2\delta} \\ \vdots \\ \frac{g(\mathbf{x}+\delta\mathbf{e}_d)-g(\mathbf{x}-\delta\mathbf{e}_d)}{2\delta} \end{bmatrix}$$

  - δ is a parameter that controls the estimation accuracy (selected 0.01 or 1)
  - $\mathbf{e}_i$ are basis vectors such that $\mathbf{e}_i$ is 1 only for the i^{th} component and 0 everywhere else
  - If the gradient exists, then the finite differences method can calculate an approximation of the gradient: $\lim_{\delta\to 0} \mathrm{FD}_{\mathbf{x}}(g(\mathbf{x}),\delta) \approx \nabla_{\mathbf{x}}\, g(\mathbf{x})$

Gradient Estimation Attack

Gradient Estimation Attack

- Approximate FGSM attack with finite difference GE method
  - Gradient of a model f is taken with respect to the cross-entropy loss $\ell_f(\mathbf{x}, y)$
    - For input $\mathbf{x}$ with true class label y, the loss is

$$\ell_f(\mathbf{x}, y) = -\sum_{j=1}^{|\mathcal{Y}|} \mathbf{1}[j = y] \log p_j^f(\mathbf{x}) = -\log p_y^f(\mathbf{x})$$

    - Recall that the derivative of a log function is $\frac{d}{dx}\log(x) = \frac{1}{x}$ and thus $\frac{d}{dx}\log(h(x)) = \frac{h'(x)}{h(x)}$
  - Therefore, the gradient of the loss function $\ell_f(\mathbf{x}, y)$ with respect to the input $\mathbf{x}$ is

$$\nabla_{\mathbf{x}} \ell_f(\mathbf{x}, y) = -\frac{\nabla_{\mathbf{x}} p_y^f(\mathbf{x})}{p_y^f(\mathbf{x})}$$

  - An untargeted FGSM adversarial sample can be generated by using the FD estimate of the gradient $\nabla_{\mathbf{x}} p_y^f(\mathbf{x})$, i.e.,

$$\mathbf{x}_{\text{adv}} = \mathbf{x} + \epsilon \cdot \text{sign}\left(\frac{\text{FD}_{\mathbf{x}}(p_y^f(\mathbf{x}), \delta)}{p_y^f(\mathbf{x})}\right)$$

  - Similarly, a targeted FGSM adversarial sample with class T can be found by using

$$\mathbf{x}_{\text{adv}} = \mathbf{x} - \epsilon \cdot \text{sign}\left(\frac{\text{FD}_{\mathbf{x}}(p_T^f(\mathbf{x}), \delta)}{p_T^f(\mathbf{x})}\right)$$

Gradient Estimation Attack

Gradient Estimation Attack

- Approximate C-W attack with finite difference GE method
  - Carlini & Wagner attack uses a loss function based on the logits values $\phi(\cdot)$

$$\ell(\mathbf{x}, y) = \max(\phi(\mathbf{x}+\delta)_y - \max\{\phi(\mathbf{x}+\delta)_i : i \neq y\}, -\kappa)$$

  - Logits values $\phi(\cdot)$ can be computed by taking the logarithm of the softmax probabilities, up to an additive constant
  - For an untargeted C-W attack, the loss is the difference between the logits for the true class y and the second-most-likely class y', i.e., $\phi(x+\delta)_y - \phi(x+\delta)_{y'}$
    - Since the loss is the difference of logits, the additive constant is canceled
    - By using FD approximation of the gradient, it is obtained

$$\mathbf{x}_{\text{adv}} = \mathbf{x} + \epsilon \cdot \text{sign}(\text{FD}_{\mathbf{x}}(\phi(\mathbf{x})_{y'} - \phi(\mathbf{x})_y, \delta))$$

  - For a targeted C-W attack, the adversarial sample is

$$\mathbf{x}_{\text{adv}} = \mathbf{x} - \epsilon \cdot \text{sign}(\text{FD}_{\mathbf{x}}(\max(\phi(\mathbf{x})_i : i \neq T) - \phi(\mathbf{x})_T, \delta))$$

Gradient Estimation Attack

Gradient Estimation Attack

- Iterative FGSM attack with finite difference GE method
  - This is similar to the Projected Gradient Descent attack, which uses several iterations of the FGSM attack and achieves higher success rate than the single step FGSM attack
  - An iterative FD attack with $t+1$ iterations using the cross-entropy loss is

$$\mathbf{x}_{\text{adv}}^{t+1} = \mathbf{x}_{\text{adv}}^{t} + \alpha \cdot \text{sign}\left(\frac{\text{FD}\left(\nabla_{\mathbf{x}_{\text{adv}}^{t}} p_y^f\left(\mathbf{x}_{\text{adv}}^{t}\right), \delta\right)}{p_y^f\left(\mathbf{x}_{\text{adv}}^{t}\right)}\right)$$

- Iterative C-W attack is also applied in a similar manner by modifying the single-step approach presented on the previous page

$$\mathbf{x}_{\text{adv}}^{t+1} = \mathbf{x}_{\text{adv}}^{t} + \alpha \cdot \text{sign}\left(\text{sign}\left(\text{FD}\left(\phi(x)_{y'} - \phi(x)_{y}, \delta\right)\right)\right)$$

Experimental Validation

Gradient Estimation Attack

- Validation of non-targeted black-box attacks using Gradient Estimation with FD
  - The table presents the success rate and average distortion (in parenthesis)
  - Baseline methods:
    - D. of M. – Difference of Means attack, uses the mean difference between the true class and the target class as added perturbation
    - Rand. – Random perturbation by adding random noise from a distribution (e.g., Gaussian)
  - 'xent' is for cross-entropy loss, 'logit' is C-W logits loss, 'I' is iterative
  - MNIST with L_∞ constraint of $\epsilon = 0.3$, and CIFAR-10 with L_∞ constraint of $\epsilon = 8$
  - Iterative C-W attack (IFD-logit) produced best results

MNIST	**Baseline**		**Gradient Estimation using Finite Differences**			
			Single-step		Iterative	
Model	D. of M.	Rand.	FD-xent	FD-logit	IFD-xent	IFD-logit
A	44.8 (5.6)	8.5 (6.1)	51.6 (3.3)	92.9 (6.1)	75.0 (3.6)	**100.0** (2.1)
B	81.5 (5.6)	7.8 (6.1)	69.2 (4.5)	98.9 (6.3)	86.7 (3.9)	**100.0** (1.6)
C	20.2 (5.6)	4.1 (6.1)	60.5 (3.8)	86.1 (6.2)	80.2 (4.5)	**100.0** (2.2)
D	97.1 (5.6)	38.5 (6.1)	95.4 (5.8)	**100.0** (6.1)	98.4 (5.4)	**100.0** (1.2)
CIFAR-10	**Baseline**		**Gradient Estimation using Finite Differences**			
			Single-step		Iterative	
Model	D. of M.	Rand.	FD-xent	FD-logit	IFD-xent	IFD-logit
Resnet-32	9.3 (440.5)	19.4 (439.4)	49.1 (217.1)	86.0 (410.3)	62.0 (149.9)	**100.0** (65.7)
Resnet-28-10	6.7 (440.5)	17.1 (439.4)	50.1 (214.8)	88.2 (421.6)	46.0 (120.4)	**100.0** (74.9)
Std.-CNN	20.3 (440.5)	22.2 (439.4)	80.0 (341.3)	98.9 (360.9)	66.0 (202.5)	**100.0** (79.9)

Experimental Validation

Gradient Estimation Attack

- Validation of targeted black-box attacks using Gradient Estimation with FD
  - Iterative FGSM (IFD-xent) attack produced best results on MNIST
  - Iterative C-W (IFD-logit) attack produced best results on CIFAR-10

MNIST	Baseline	Gradient Estimation using Finite Differences			
		Single-step		Iterative	
Model	D. of M.	FD-xent	FD-logit	IFD-xent	IFD-logit
A	15.0 (5.6)	30.0 (6.0)	29.9 (6.1)	**100.0** (4.2)	99.7 (2.7)
B	35.5 (5.6)	29.5 (6.3)	29.3 (6.3)	**99.9** (4.1)	98.7 (2.4)
C	5.84 (5.6)	34.1 (6.1)	33.8 (6.4)	**100.0** (4.3)	99.8 (3.0)
D	59.8 (5.6)	61.4 (6.3)	60.8 (6.3)	**100.0** (3.7)	99.9 (1.9)
CIFAR-10	**Baseline**	**Gradient Estimation using Finite Differences**			
		Single-step		Iterative	
Model	D. of M.	FD-xent	FD-logit	IFD-xent	IFD-logit
Resnet-32	1.2 (440.3)	23.8 (439.5)	23.0 (437.0)	**100.0** (110.9)	**100.0** (89.5)
Resnet-28-10	0.9 (440.3)	29.2 (439.4)	28.0 (436.1)	100.0 (123.2)	**100.0** (98.3)
Std.-CNN	2.6 (440.3)	44.5 (439.5)	40.3 (434.9)	**99.0** (178.8)	95.0 (126.8)

Query Reduction

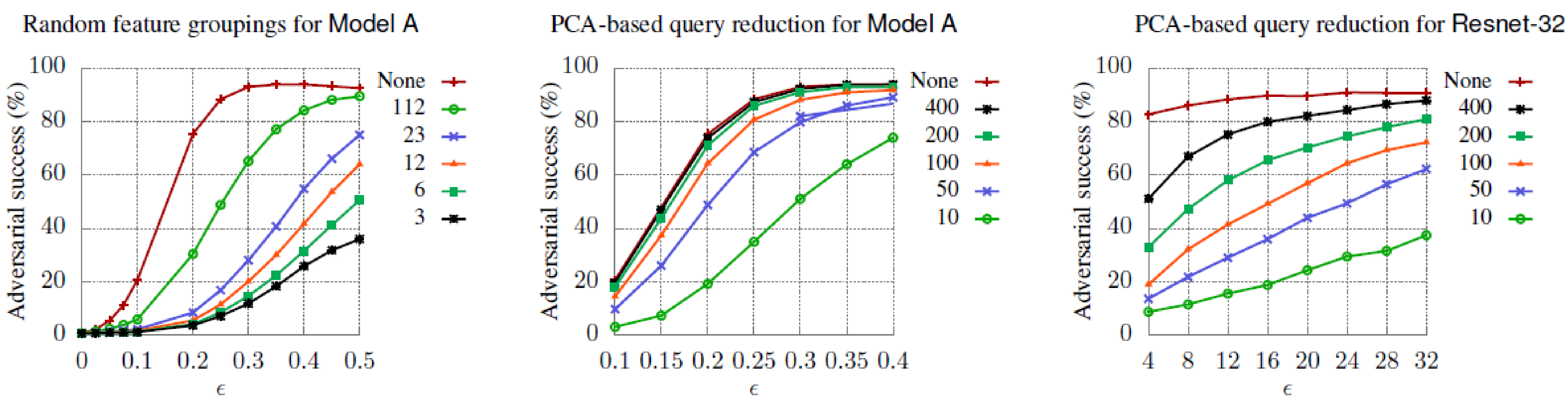
Gradient Estimation Attack

- Shortcoming of the proposed approach:
  - Requires $O(d)$ queries per input, where d is the dimension of the input (e.g., number of pixels in images)
  - The presented FD approximation required $2 \cdot d$ queries
- The authors propose two approaches for reducing the number of queries
  - Random grouping
    - The gradient is estimated only for a random group of selected pixels, instead of estimating the gradient per each pixel
  - PCA (Principal Component Analysis)
    - Compute the gradient only along a number of principal component vectors

Query Reduction

Gradient Estimation Attack

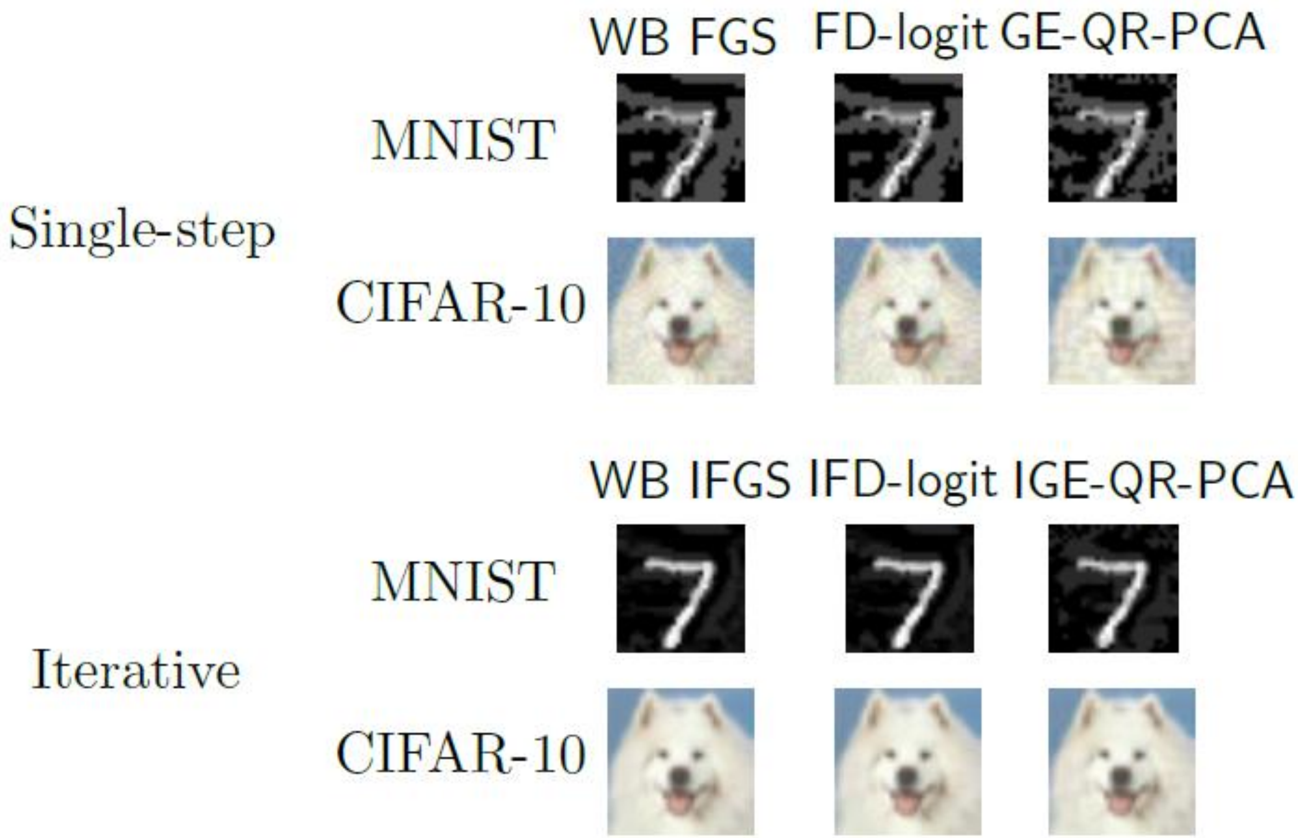
- Validation of the methods for query reduction
  - For random grouping, the success rate decreases with decreasing the group size (left figure)
    - I.e., using only 3 group of pixels to estimate the gradient is less efficient than using 112 groups of pixels
  - For PCA, the success rate decreases as the number of PC is decreased (middle and right figure)
    - The success rate is still high for smaller number of PC



Adversarial Samples

Gradient Estimation Attack

- Non-targeted adversarial samples
  - WB-IFGS – white-box iterative FGSM attack
  - IFD-logit – black-box iterative C&W attack (logit loss)
  - IGE-QR-PCA - black-box Iterative Gradient Estimation with Query Reduction using PCA



Defense Evaluation

Gradient Estimation Attack

- Evaluation of adversarial samples against three adversarial defenses
  - Adversarial training (Szagedy et al, 2014): Adv column in the table
  - Ensemble adversarial training (Tramer et al, 2017): Adv-Ens column
  - Iterative adversarial training (Madry et al, 2017): Adv-Iter column
- The accuracy is almost the same as for benign (non-attacked) images (first column in the table)

Dataset (Model)	Benign	Adv	Adv-Ens	Adv-Iter
MNIST (A)	99.2	99.4	99.2	99.3
CIFAR-10 (Resnet-32)	92.4	92.1	91.7	79.1

Attacks on Real Models

Gradient Estimation Attack

- Attacks on two real-world models hosted by Clarifai
  - Not Safe For Work (NSFW) model
    - Two categories: ‘safe’, ‘not safe’
  - Content Moderation model
    - Five categories: ‘safe’, ‘suggestive’, ‘explicit’, ‘drug,’ and ‘gore’
    - Example: an adversary could upload violent adversarially-modified images, which may be marked incorrectly as ‘safe’ by the Content Moderation model

Original image
Class: ‘drug’
Confidence: 0.99

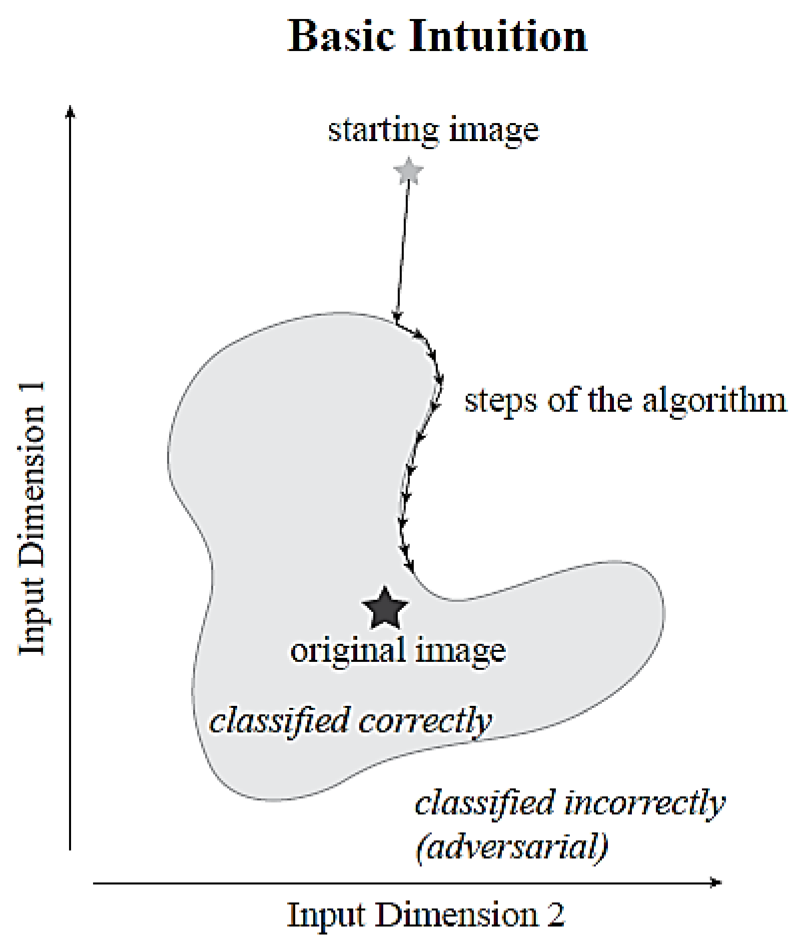
Adversarial image
Class: ‘safe’
Confidence: 0.96

Boundary Attack

- ***Brendel, Rauber, and Bethge (2018) Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models***
- A query-based black-box attack called ***Boundary Attack***
  - This is a decision-based attack, i.e., it requires only queries of the output class, and not the logits or output probabilities
  - Can perform both non-targeted and targeted attacks
- Advantage:
  - Finds low-perturbation images only by using the output class information
  - Relevant to real-world application, where access to the model may not be possible
- Disadvantage:
  - Requires many iterations to converge (i.e., large number of queries)
- Validation on MNIST, CIFAR-10, and ImageNet
  - And, on real-world applied models

Boundary Attack

- Boundary Attack intuition
  - The starting image is drawn from a uniform random distribution (random noise), and is adversarial (i.e., different than the true label)
  - Iteratively reduce the L_2 distance to the original image by adding small perturbations
  - Walk along the boundary between the adversarial and the non-adversarial region, but stay in the adversarial region
    - I.e., whenever the added perturbation results in correct classification, reject those samples (a.k.a., sample rejection)
  - When the distance to the original image cannot be further reduced, or when the number of set iteration steps is reached, stop



Boundary Attack Algorithm

Boundary Attack

- Boundary Attack algorithm
  - The initial image $\tilde{\mathbf{o}}^0$ is sampled from a uniform distribution $\mathcal{U}(0,1)$
  - The adversarially perturbed image at the k^{th} step is denoted $\tilde{\mathbf{o}}^k$
  - Adversarial criterion $c(\cdot)$ is: misclassification
    - I.e., different class than the true class (non-targeted attack), or the target class (targeted attack)
  - Decision of model $d(\cdot)$ is: L_2 distance between the perturbed and the original image
  - The proposal distribution for the perturbation η_k is discussed on next page

Data: original image $\mathbf{o}$, adversarial criterion $c(.)$, decision of model $d(.)$
Result: adversarial example $\tilde{o}$ such that the distance $d(o, \tilde{o}) = \|o - \tilde{o}\|_2^2$ is minimized
initialization: $k = 0$, $\tilde{o}^0 \sim \mathcal{U}(0, 1)$ s.t. $\tilde{o}^0$ is adversarial;
while $k <$ *maximum number of steps* **do**
  draw random perturbation from proposal distribution $\eta_k \sim \mathcal{P}(\tilde{o}^{k-1})$;
  if $\tilde{o}^{k-1} + \eta_k$ *is adversarial* **then**
    set $\tilde{o}^k = \tilde{o}^{k-1} + \eta_k$;
  else
    set $\tilde{o}^k = \tilde{o}^{k-1}$;
  end
  $k = k + 1$
end

Boundary Attack

Boundary Attack

- For the proposal distribution $\mathcal{P}(\tilde{\mathbf{o}}^{k-1})$ of the perturbation η_k, the authors used a Gaussian distribution $\mathcal{N}(0,1)$
  - This perturbation is denoted as #1 – random orthogonal step in the figure below
- Next, it is ensured that the proposed adversarial sample is a regular image with all pixels clipped in the range [0,1]

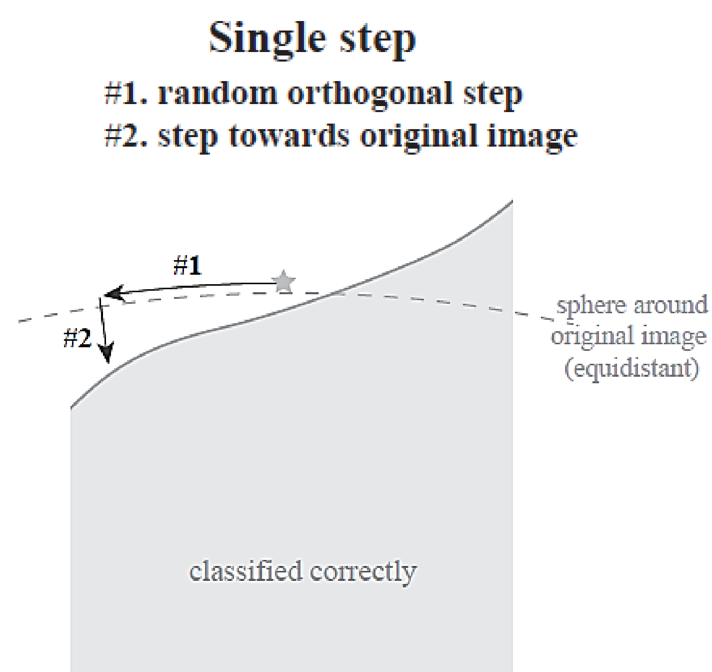
$$\tilde{\mathbf{o}}_i^{k-1} + \eta_i^k \in [0,1]$$

- It is also ensured that the perturbation η_k is within a ball with radius δ around the original image $\mathbf{o}$ (i.e., the added perturbation at each step is limited)

$$\left\|\eta^k\right\|_2 = \delta \cdot d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1})$$

- Afterward, a small movement ϵ (#2 step in the image) is made toward the original image $\mathbf{o}$, so that the distance to $\mathbf{o}$ is iteratively reduced

$$d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1} + \eta^k) - d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1}) = \epsilon d(\mathbf{o}, \tilde{\mathbf{o}}^{k-1})$$

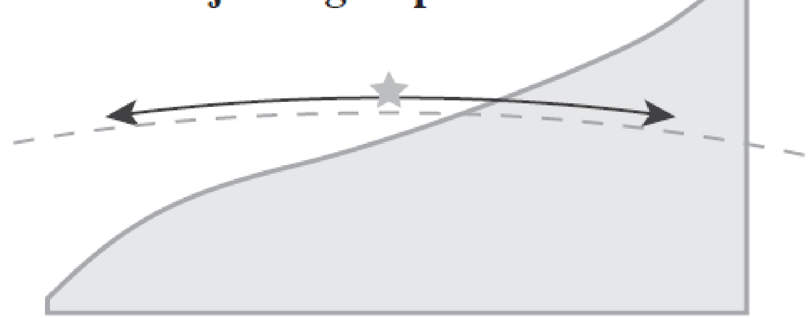


Boundary Attack

Boundary Attack

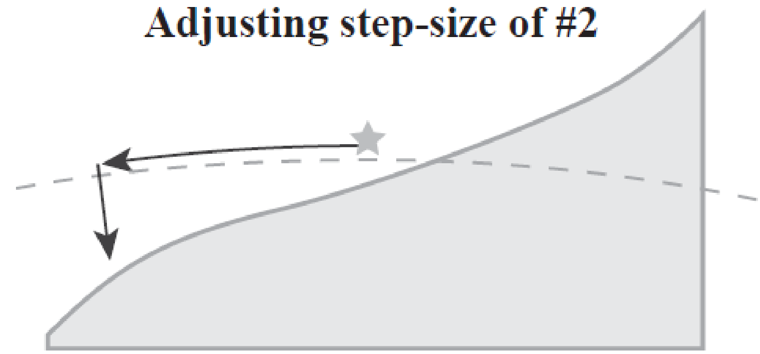
- The two parameters δ (random orthogonal step) and ϵ (step toward the original image) are adjusted dynamically
- The parameters δ is adjusted to that that about 50% of the perturbations are adversarial
  - If this ratio is much lower than 50%, the step size δ is reduced
  - In the opposite case, δ is increased
- Next, a small step ϵ toward the original image is applied
  - If the success rate is too small, ϵ is decreased
  - If it is too large, ϵ is increased
- The attack is converged whenever ϵ converges to zero
  - I.e., the L_2 distance to the original image can not be reduced anymore

Adjusting step-size of #1



~50% of orthogonal perturbations should be within adversarial region

Adjusting step-size of #2

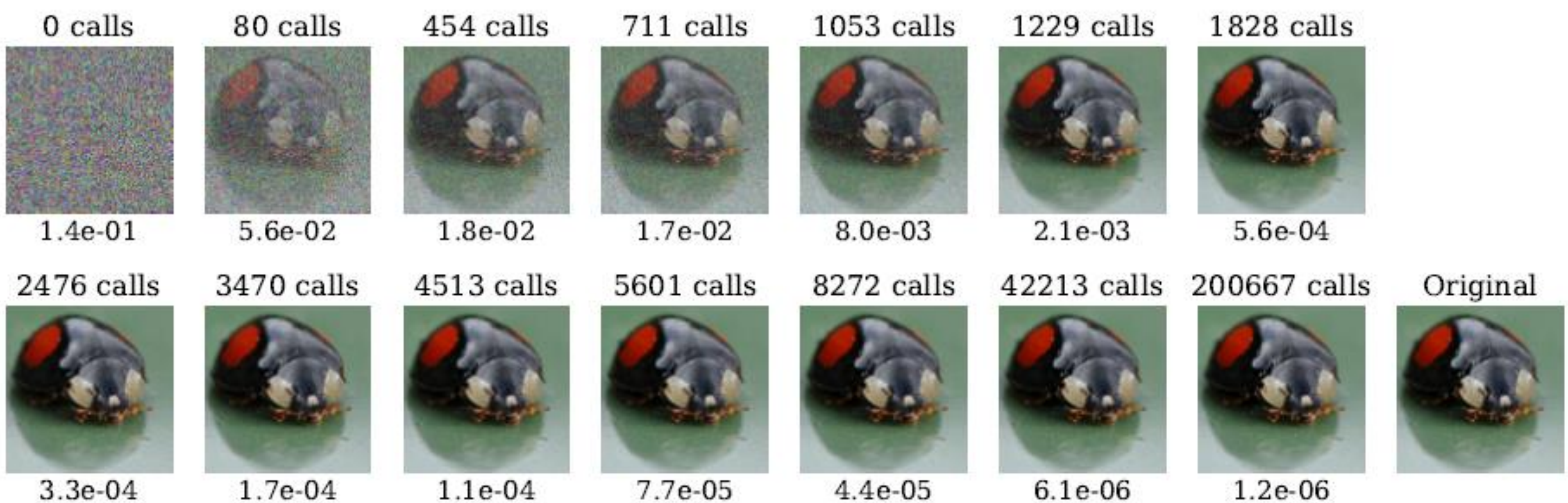


Success rate of total perturbation should be higher then threshold (e.g. 25%).

Adversarial Examples

Boundary Attack

- Example of an untargeted attack
  - Starts from upper left and proceeds to the lower right image
  - Above: total number of calls, i.e., queries
  - Below: L_2 distance between the attacked image and the original image
  - The original image used for the attack is shown in the lower right corner



Adversarial Examples

Boundary Attack

- Example of a targeted attack
  - Original class: tiger cat (lower right image)
  - Target class: Dalmatian dog (upper left image)
- Goal: create an adversarial image that is perceptually close (in L_2 distance) to a given image of a tiger cat (lower right), but is classified as a Dalmatian dog
  - The algorithm is initialized from a sample image of the target class that is correctly classified by the model (upper left image of Dalmatian dog)

Experimental Validation

Boundary Attack

- Comparison to FGSM, DeepFool, and Carlini-Wagner non-targeted attacks
  - Presented values: median L_2 distance to the original images
  - The added perturbations by the Boundary Attack are comparable and not much larger than the perturbation by white box models

				ImageNet		
	Attack Type	MNIST	CIFAR	VGG-19	ResNet-50	Inception-v3
FGSM	gradient-based	4.2e-02	2.5e-05	1.0e-06	1.0e-06	9.7e-07
DeepFool	gradient-based	4.3e-03	5.8e-06	1.9e-07	7.5e-08	5.2e-08
Carlini & Wagner	gradient-based	2.2e-03	7.5e-06	5.7e-07	2.2e-07	7.6e-08
Boundary (ours)	decision-based	3.6e-03	5.6e-06	2.9e-07	1.0e-07	6.5e-08

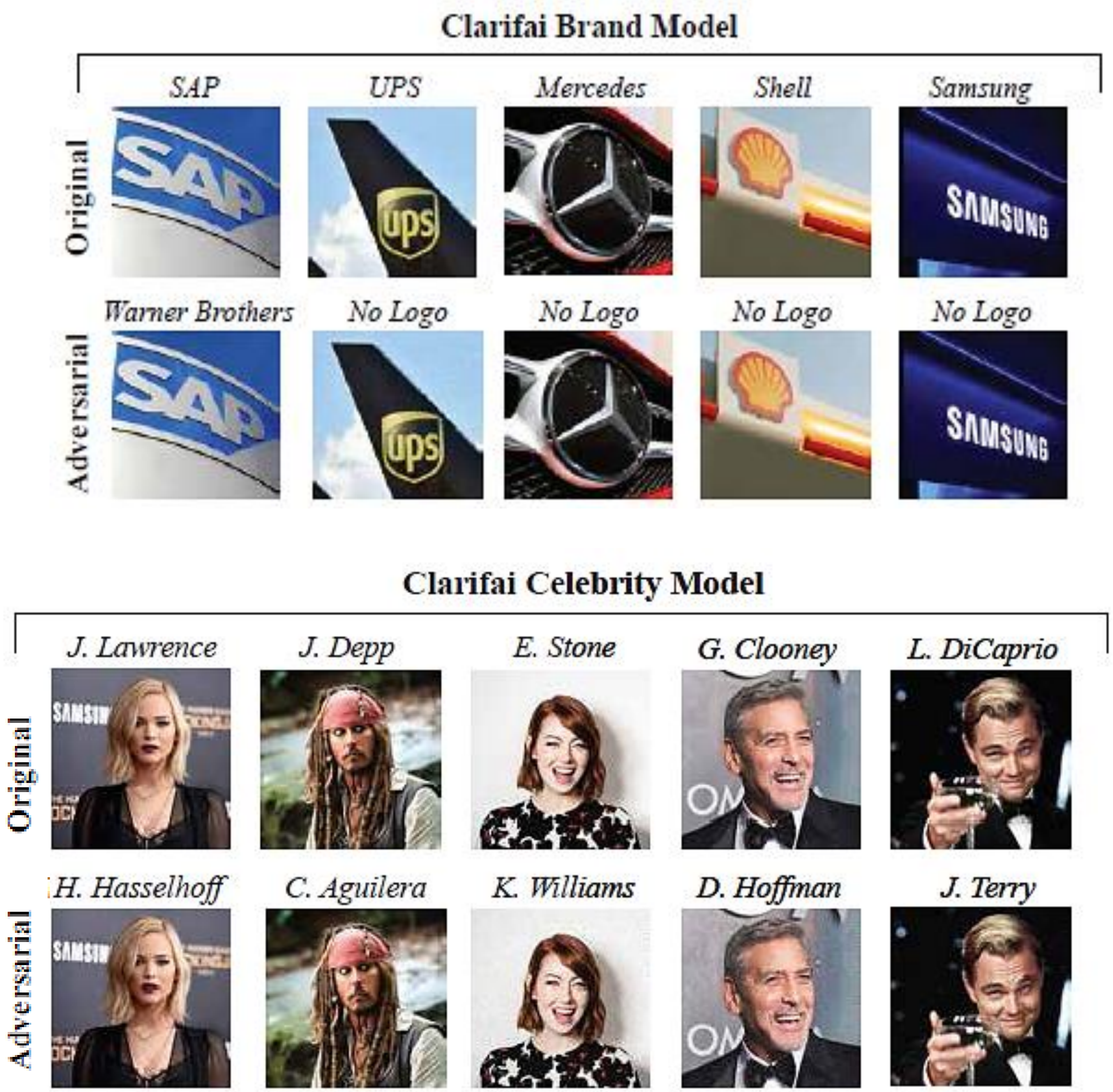
- Comparison to Carlini-Wagner targeted attack

	Attack Type	MNIST	CIFAR	VGG-19
Carlini & Wagner	gradient-based	4.8e-03	3.0e-05	5.7e-06
Boundary (ours)	decision-based	6.5e-03	3.3e-05	9.9e-06

Real-World Applications

Boundary Attack

- In many real-world applications, the attacker has no access to the model or the training data, but can only observe the final decision
  - E.g., security systems (face identification), autonomous cars, speech recognition (Alexa, Cortana)
- The authors applied Boundary Attack to two models by [Clarifai]
  - For identifying over 500 brand names in natural images
  - For identifying over 10,000 celebrities



Transfer-based Attacks

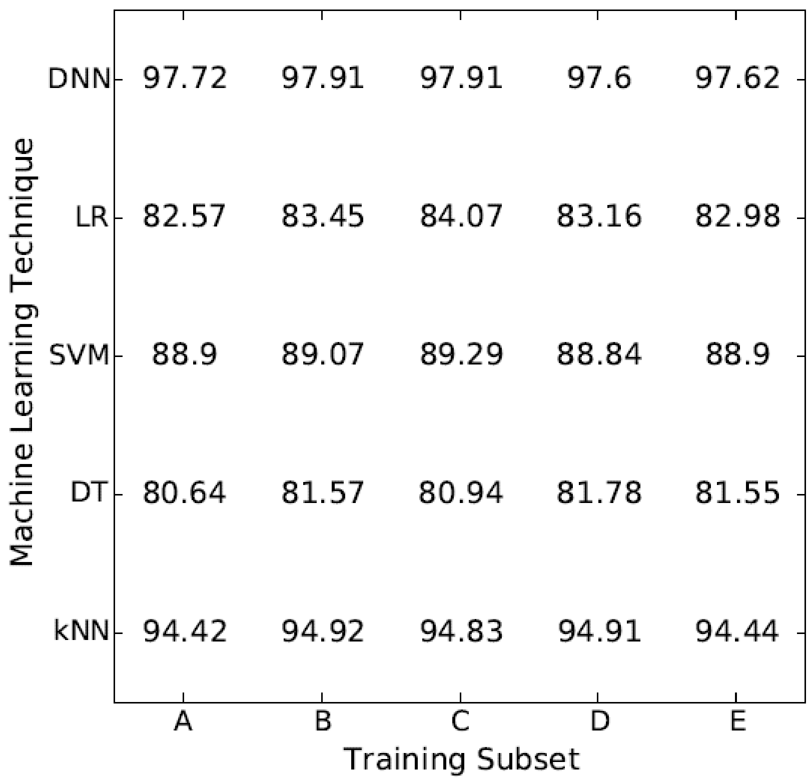
- ***Transfer-based attacks*** (or ***transferability attacks***)
  - The adversary does not query the model
- Reviewed attacks
  - **Substitute model attack** (a.k.a. surrogate local model attack)
    - Train a substitute model, and transfer the generated adversarial samples to the target model
  - **Ensemble of local models attack**
    - Use an ensemble of local models for generating adversarial examples

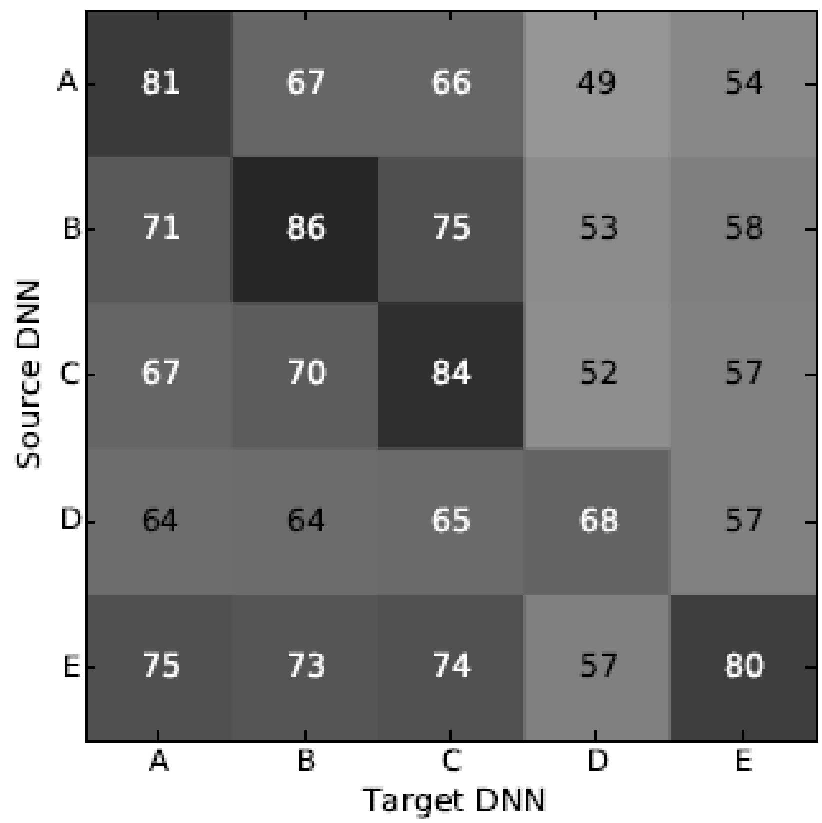
Substitute Model Attack

- ***Substitute model attack*** (or ***surrogate local model attack***)
  - Papernot et al. (2016) Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples
- Create adversarial example for a substitute model, and afterward transfer the generated examples to the target model
- Transferability between the following ML models is explored:
  - Deep neural networks (DNNs)
  - Logistic regression (LR)
  - Support vector machines (SVM)
  - Decision trees (DT)
  - *k*-Nearest neighbors (kNN)
  - Ensembles (Ens)
- Evaluated on MNIST

Substitute Model Attack

- ***Intra-technique variability***
  - Five models (A,B,C,D,E) of the same ML method are trained on different subsets of the training data and the generated adversarial examples are transferred
    - E.g., adversarial examples created by one DNN are transferred to the other DNNs
  - Model accuracies (left figure), and attack success rate for DNNs (right figure)

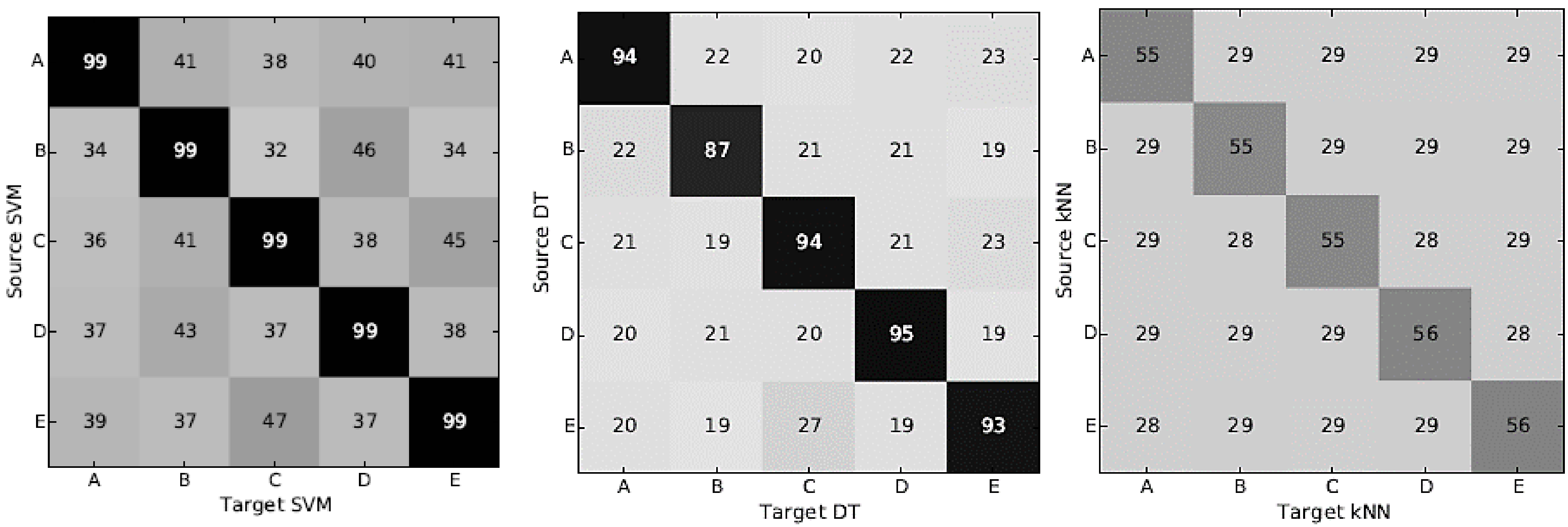




Substitute Model Attack

Substitute Model Attack

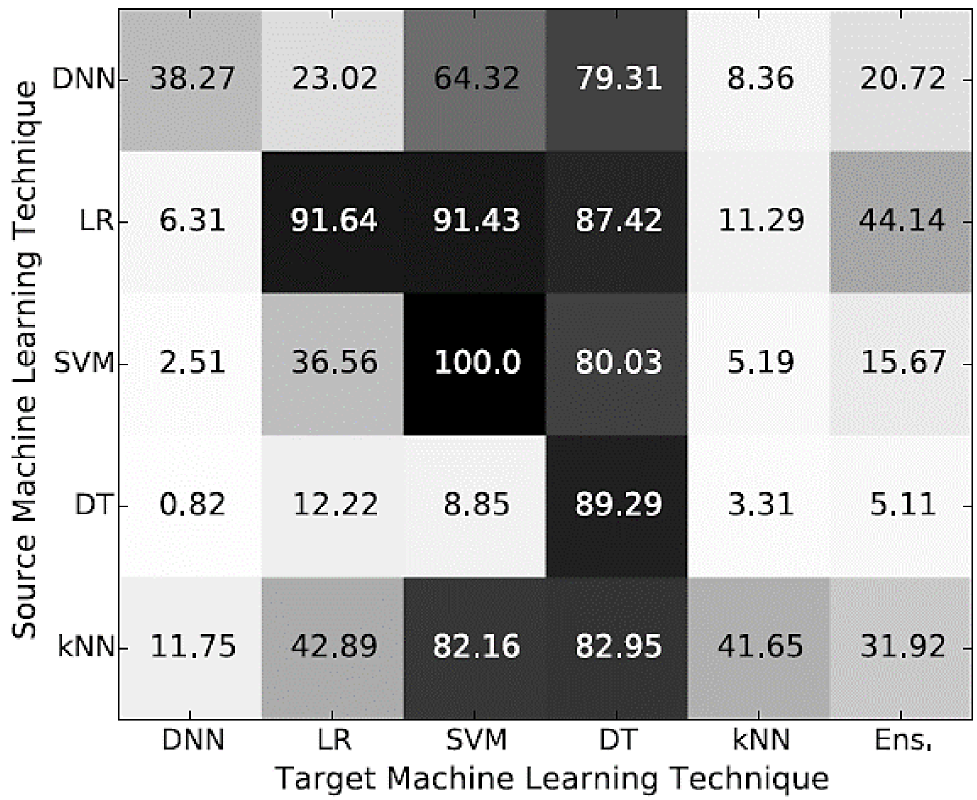
- Intra-technique variability
  - Attack success rates for SVM, DT, and kNN are shown below, when transferring examples between the models A, B, C, D, and E of the same ML method
  - Differentiable models like DNNs and LR are more vulnerable to intra-technique transferability than non-differentiable models like SVMs, DTs, and kNNs



Substitute Model Attack

Substitute Model Attack

- ***Cross-technique variability***
  - Transfer adversarial samples from one ML method to the other ML methods
    - E.g., adversarial examples created by DNN transferred to other ML models (the first row)
  - The most vulnerable model is DT: misclassification rates from 79.31% to 89.29%
  - The most resilient is DNN (first column): misclassification between 0.82% and 38.27%



Ensemble of Local Models Attack

- ***Ensemble of local models attack***
  - Liu et al. (2017) Delving into Transferable Adversarial Examples and Black-box Attacks
- Observations regarding transferability
  - Transferable non-targeted adversarial examples are easy to find
  - However, targeted adversarial examples rarely transfer with their target labels
- The proposed approach allows transferring targeted adversarial examples

Ensemble of Local Models Attack

Ensemble of Local Models Attack

- On ImageNet, targeted examples do not transfer across models
  - Only a small percentage of adversarial images retain the target label when transferred to other models (between 1% and 4%, off diagonal values in the table)
  - RMSD is the average perturbation of the used adversarial images

	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%

- On the other hand, untargeted examples transfer well

	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%

Ensemble of Local Models Attack

Ensemble of Local Models Attack

- Hypothesis: if an adversarial image remains adversarial for multiple models, it is more likely to transfer to other models as well
- Approach: solve the following optimization problem (for targeted attack):

$$\mathrm{argmin}_{x^\star} - \log\left(\left(\sum_{i=1}^{k} \alpha_i J_i(x^\star) \right) \cdot \mathbf{1}_{y^\star} \right) + \lambda d(x, x^\star)$$

- The problem is similar to C&W
  - x is a clean image
  - x^* is an adversarial image
  - $d(x, x^*)$ is distance function
  - $J_1, J_2, \dots, J_k$ are white-box models in the ensemble
  - $\alpha_1, \alpha_2, \dots, \alpha_k$ are the ensemble weights
  - $-\log(\alpha_1 J_1 \cdot \mathbf{1}_{y^*})$ is the cross-entropy loss between the prediction by model J_1 and the one-hot vector for the target class $\mathbf{1}_{y^*}$

Targeted Attack Evaluation

Ensemble of Local Models Attack

- Targeted attack using the ensemble attack
  - E.g., the first row shows the attack success rate when an ensemble of 4 models (ResNet-101, ResNet-50.VGG-16, and GoogLeNet) is trained, and the samples are transferred to ResNet-152
    - The success rate of transferred attack is 38%

	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%

Non-targeted Attack Evaluation

Ensemble of Local Models Attack

- Non-targeted ensemble attack results
  - Using an ensemble of four models, the success rate is very high for non-targeted attack

	RMSD	ResNet-152	ResNet-101	ResNet-50	VGG-16	GoogLeNet
-ResNet-152	17.17	0%	0%	0%	0%	0%
-ResNet-101	17.25	0%	1%	0%	0%	0%
-ResNet-50	17.25	0%	0%	2%	0%	0%
-VGG-16	17.80	0%	0%	0%	6%	0%
-GoogLeNet	17.41	0%	0%	0%	0%	5%

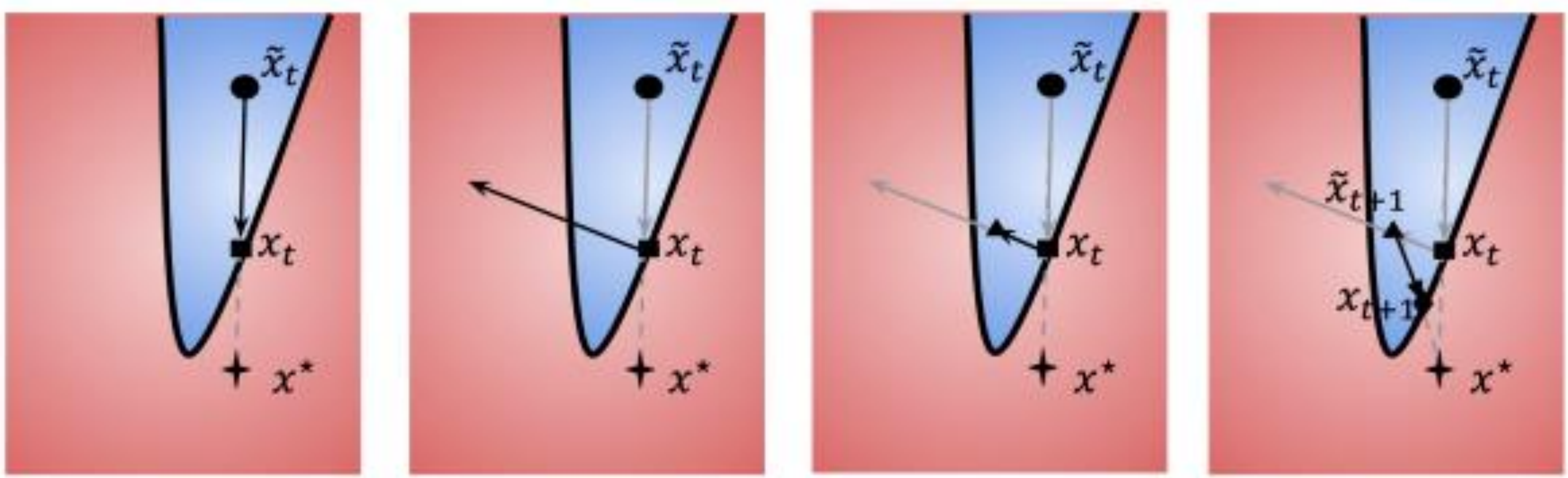
HopSkipJump Attack

- ***HopSkipJump Attack***
  - Chen and Jordan (2019) HopSkipJumpAttack: A Query-efficient Decision-based Adversarial Attack
- This attack is an extension of the Boundary Attack
  - I.e., it is a decision-based attack, and therefore has access only to the predicted output class
    - HopSkipJump Attack requires significantly fewer queries than the Boundary Attack
  - It includes both untargeted and targeted attacks
  - Proposes a a novel approach for estimation of the gradient direction along the decision boundary

HopSkipJump Attack

HopSkipJump Attack

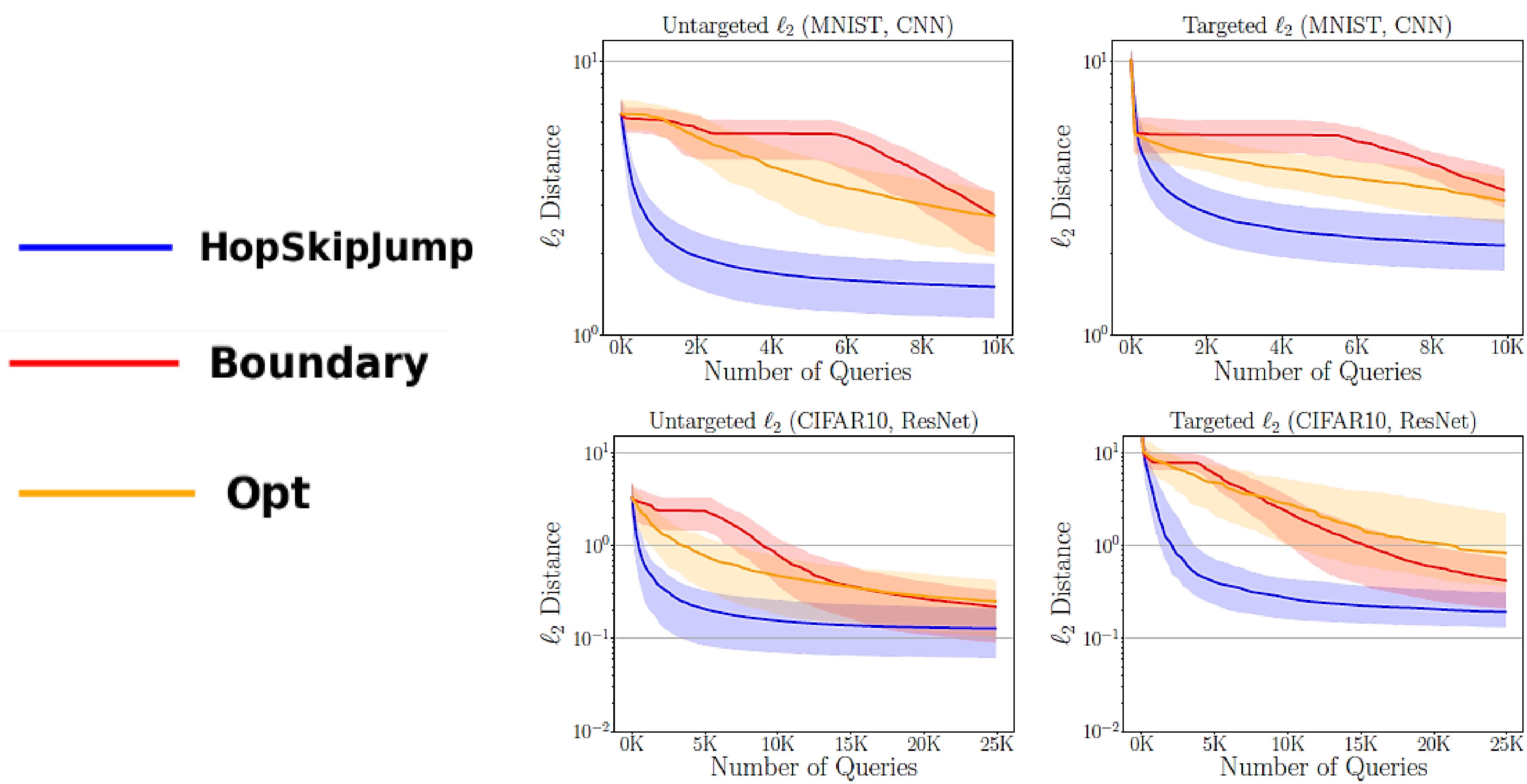
- Approach:
  1. Start from an adversarial image $\tilde{x}_t$
  2. Perform a binary search to the original image x^* to find the boundary (left figure)
  3. Estimate the gradient direction at the boundary point x_t (second figure from left)
  4. Perform a step-size search, and update to the next image $\tilde{x}_{t+1}$
  5. Search again for the next boundary point x_{t+1} (right figure)
  6. Repeat until the closest adversarial image to the original image x^* is found



HopSkipJump Attack

HopSkipJump Attack

- Experimental evaluation
  - Comparison to Boundary attack and Opt attack on CIFAR-10
  - HopSkipJump (blue curve) achieves lower ℓ_2 perturbation using fewer queries

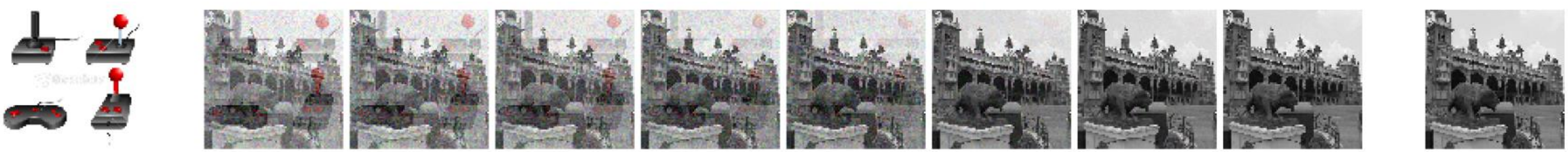


HopSkipJump Attack

HopSkipJump Attack

- Untargeted attack
  - 2^{nd} to 9th columns: images at 100, 200, 500, 1K, 2K, 5K, 10K, 25K queries
  - The original image for the attack is shown on the right

- Targeted attack



ZOO Attack

ZOO Attack

- ***ZOO attack***
  - Chen (2017) Zoo: Zeroth-order optimization based black-box attacks to deep neural networks without training substitute models
- Zeroth-order optimization refers to optimization based on access to the function values $f(x)$ only
  - As opposed to first-order optimization via the gradient $\nabla f(x)$
  - E.g., score-based and decision-based black-box approaches are zeroth-order optimization methods, as they don't require the gradient information
- ZOO attack has similarities with the Gradient Estimation Attack
- It is a score-based black-box version of the Carlini-Wagner attack

Adversarial Attack

ZOO Attack

- Recall again that the Gradient Estimation attack uses the finite difference approach to approximate the gradient as $\boldsymbol{g} = \nabla_{\boldsymbol{x}} \boldsymbol{f}(\mathbf{x}) \approx \frac{f(\mathbf{x}+h)-f(\mathbf{x}-h)}{2h}$
  - E.g., if the intensity of a pixel x_i is 150, and $h = 10$, then we will query the model to give us the predictions for $f(150 + 10) = f(160)$ and for $f(150 - 10) = f(140)$, so we can estimate the gradient $\widehat{\boldsymbol{g}_i} = \nabla_{x_i} \boldsymbol{f}(\mathbf{x})$ for the pixel x_i
  - We need to do 2 queries for each pixel, and for an images with 28×28 pixels = 784 pixels, we need to do $2 \cdot 784 = 1{,}568$ queries to estimate the gradient
- ZOO attack solves an optimization, similar to C&W targeted white-box attack

$$\text{minimize } \|\mathbf{x} - \mathbf{x_0}\|_2^2 + c \cdot \left(Z(x)_{y'} - Z(x)_T\right)$$
$$\text{subject to } \mathbf{x} \in [0,1]$$

  - ZOO solves the optimization problem with the FD estimated loss based on:

$$\text{minimize } \|\mathbf{x} - \mathbf{x_0}\|_2^2 + c \cdot FD\left(Z(x)_{y'} - Z(x)_T, h\right)$$
$$\text{subject to } \mathbf{x} \in [0,1]$$

  - *Adam optimization* is used to solve the problem

Adam Optimization Attack

ZOO Attack

- Algorithm for the ZOO attack using Adam optimization

Algorithm 2 ZOO-ADAM: Zeroth Order Stochastic Coordinate Descent with Coordinate-wise ADAM

Require: Step size η, ADAM states $M \in \mathbb{R}^p, v \in \mathbb{R}^p, T \in \mathbb{Z}^p$, ADAM hyper-parameters $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$

$M \leftarrow \mathbf{0}, v \leftarrow \mathbf{0}, T \leftarrow \mathbf{0}$
while not converged **do**
    Randomly pick a coordinate $i \in \{1, \cdots, p\}$
    Estimate $\hat{g}_i$ using (6)
    $T_i \leftarrow T_i + 1$
    $M_i \leftarrow \beta_1 M_i + (1 - \beta_1)\hat{g}_i, \quad v_i \leftarrow \beta_2 v_i + (1 - \beta_2)\hat{g}_i^2$
    $\hat{M}_i = M_i/(1 - \beta_1^{T_i}), \quad \hat{v}_i = v_i/(1 - \beta_2^{T_i})$
    $\delta^* = -\eta \frac{\hat{M}_i}{\sqrt{\hat{v}_i} + \epsilon}$
    Update $\mathbf{x}_i \leftarrow \mathbf{x}_i + \delta^*$
end while

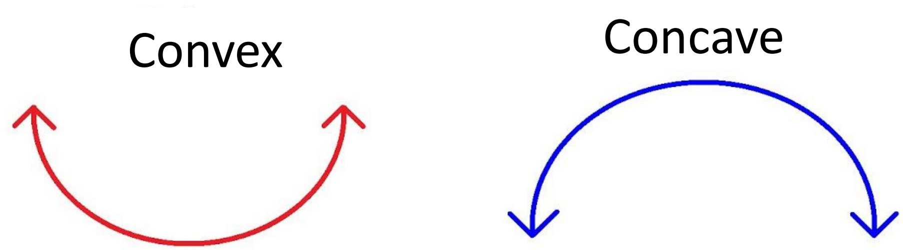
Newton Optimization Attack

ZOO Attack

- The paper proposed one more similar approach, that instead of Adam optimization uses *Newton optimization* method
  - Newton optimization method finds a minimum of $f(x)$ by performing the following iterations: $x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$
- The approximation of the Hessian matrix of the model is estimated based on

$$\boldsymbol{h} = \frac{\partial^2}{\partial \boldsymbol{x}^2} \boldsymbol{f}(\mathbf{x}) \approx \frac{f(\mathbf{x}+h) - 2f(\mathbf{x}) + f(\mathbf{x}-h)}{\boldsymbol{h}^2}$$

  - If $\boldsymbol{h} > \mathbf{0}$, then the loss function is convex, update is based on $\boldsymbol{g}/\boldsymbol{h}$ (i.e., $x_k - \frac{f'(x_k)}{f''(x_k)}$)
  - If $\boldsymbol{h} \leq \mathbf{0}$, then the loss function is concave, update is based only on the gradient $\boldsymbol{g}$ (i.e., $x_k - f'(x_k)$)



$$\frac{\partial^2 \boldsymbol{f}(\mathbf{x})}{\partial \boldsymbol{x}^2} > 0 \qquad \frac{\partial^2 \boldsymbol{f}(\mathbf{x})}{\partial \boldsymbol{x}^2} < 0$$

Newton Optimization Attack

ZOO Attack

- Algorithm for the ZOO attack with Newton optimization

Algorithm 3 ZOO-Newton: Zeroth Order Stochastic Coordinate Descent with Coordinate-wise Newton's Method

Require: Step size η

while not converged **do**
    Randomly pick a coordinate $i \in \{1, \cdots, p\}$
    Estimate $\hat{g}_i$ and $\hat{h}_i$ using (6) and (7)
    if $\hat{h}_i \leq 0$ **then**
        $\delta^* \leftarrow -\eta \hat{g}_i$
    else
        $\delta^* \leftarrow -\eta \frac{\hat{g}_i}{\hat{h}_i}$
    end if
    Update $\mathbf{x}_i \leftarrow \mathbf{x}_i + \delta^*$
end while

Experimental Evaluation

ZOO Attack

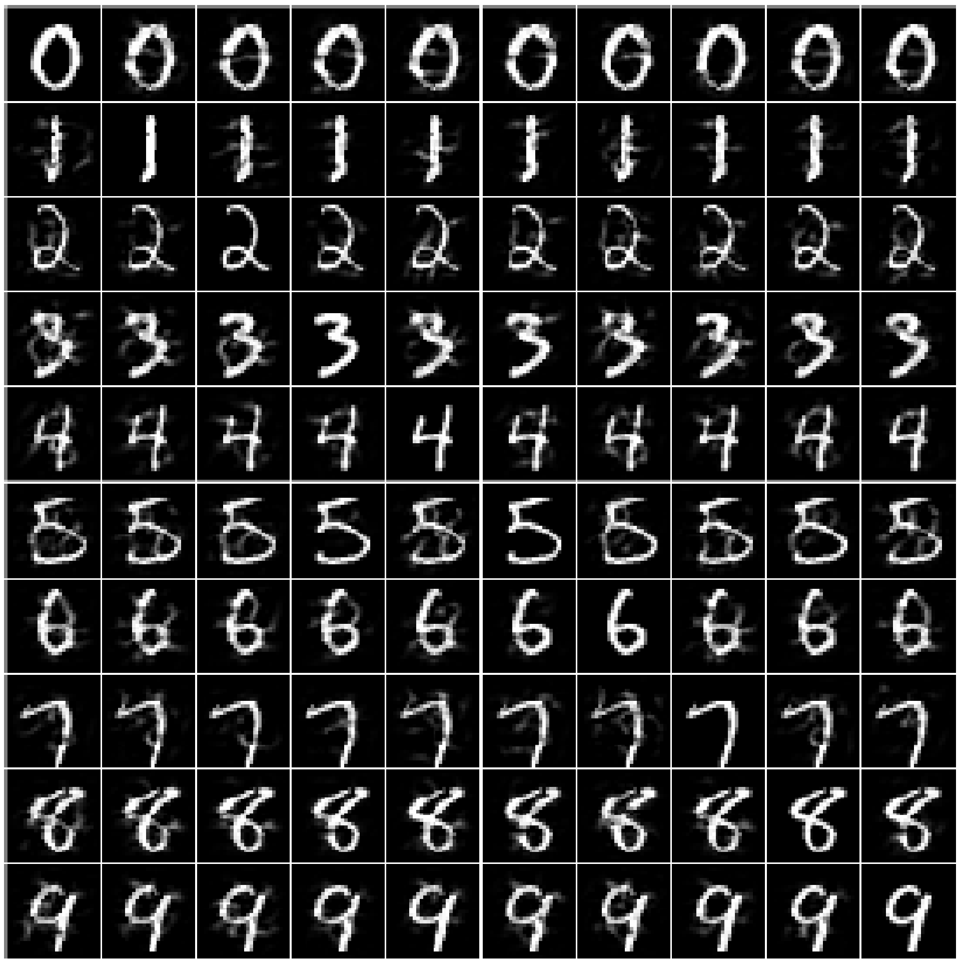
- On MNIST and Cifar-10, ZOO attacks achieved almost 100% success rate
  - The added L_2 perturbations are comparable to C&W white-box attack
  - As expected, the time for generating adversarial samples is longer than white-box attacks

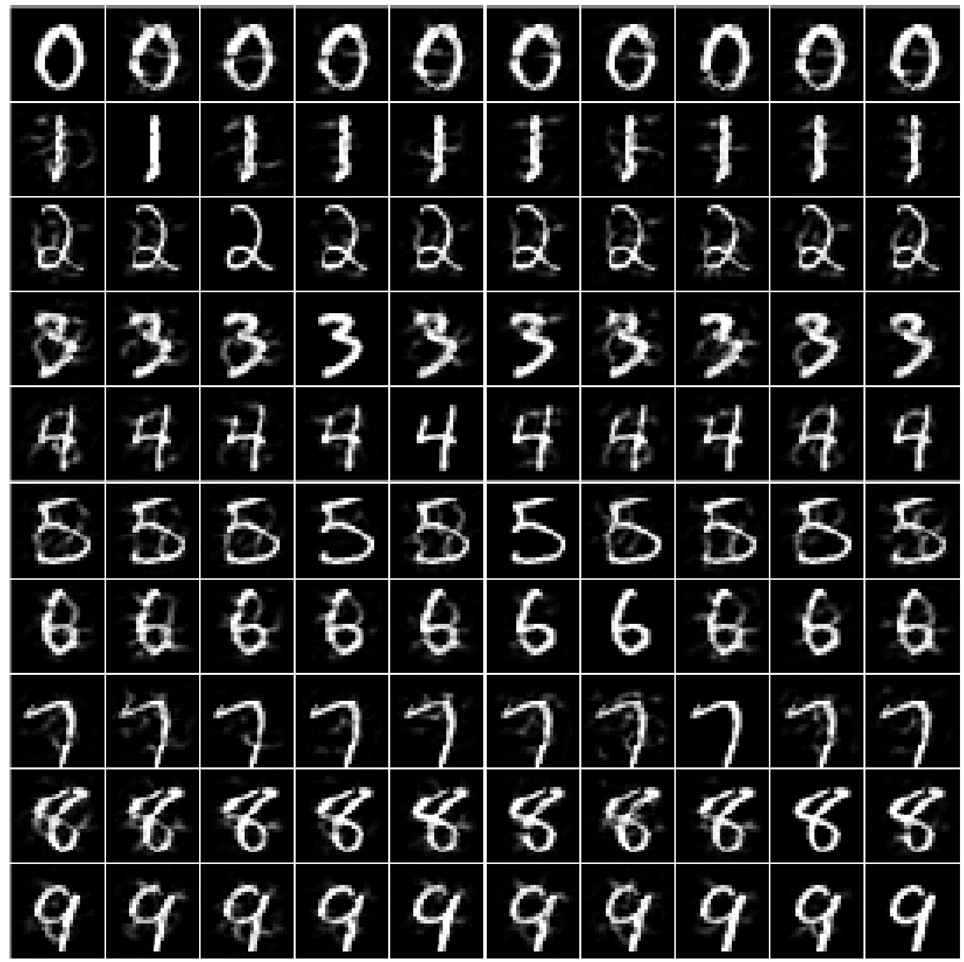
	MNIST					
	Untargeted			Targeted		
	Success Rate	Avg. L_2	Avg. Time (per attack)	Success Rate	Avg. L_2	Avg. Time (per attack)
White-box (C&W)	100 %	1.48066	0.48 min	100 %	2.00661	0.53 min
Black-box (Substitute Model + FGSM)	40.6 %	-	0.002 sec (+ 6.16 min)	7.48 %	-	0.002 sec (+ 6.16 min)
Black-box (Substitute Model + C&W)	33.3 %	3.6111	0.76 min (+ 6.16 min)	26.74 %	5.272	0.80 min (+ 6.16 min)
Proposed black-box (ZOO-ADAM)	100 %	1.49550	1.38 min	98.9 %	1.987068	1.62 min
Proposed black-box (ZOO-Newton)	100 %	1.51502	2.75 min	98.9 %	2.057264	2.06 min
	CIFAR10					
	Untargeted			Targeted		
	Success Rate	Avg. L_2	Avg. Time (per attack)	Success Rate	Avg. L_2	Avg. Time (per attack)
White-box (C&W)	100 %	0.17980	0.20 min	100 %	0.37974	0.16 min
Black-box (Substitute Model + FGSM)	76.1 %	-	0.005 sec (+ 7.81 min)	11.48 %	-	0.005 sec (+ 7.81 min)
Black-box (Substitute Model + C&W)	25.3 %	2.9708	0.47 min (+ 7.81 min)	5.3 %	5.7439	0.49 min (+ 7.81 min)
Proposed Black-box (ZOO-ADAM)	100 %	0.19973	3.43 min	96.8 %	0.39879	3.95 min
Proposed Black-box (ZOO-Newton)	100 %	0.23554	4.41 min	97.0 %	0.54226	4.40 min

Experimental Evaluation

ZOO Attack

- Comparison between C&W white-box (left) and ZOO attack (right)





Queries Reduction

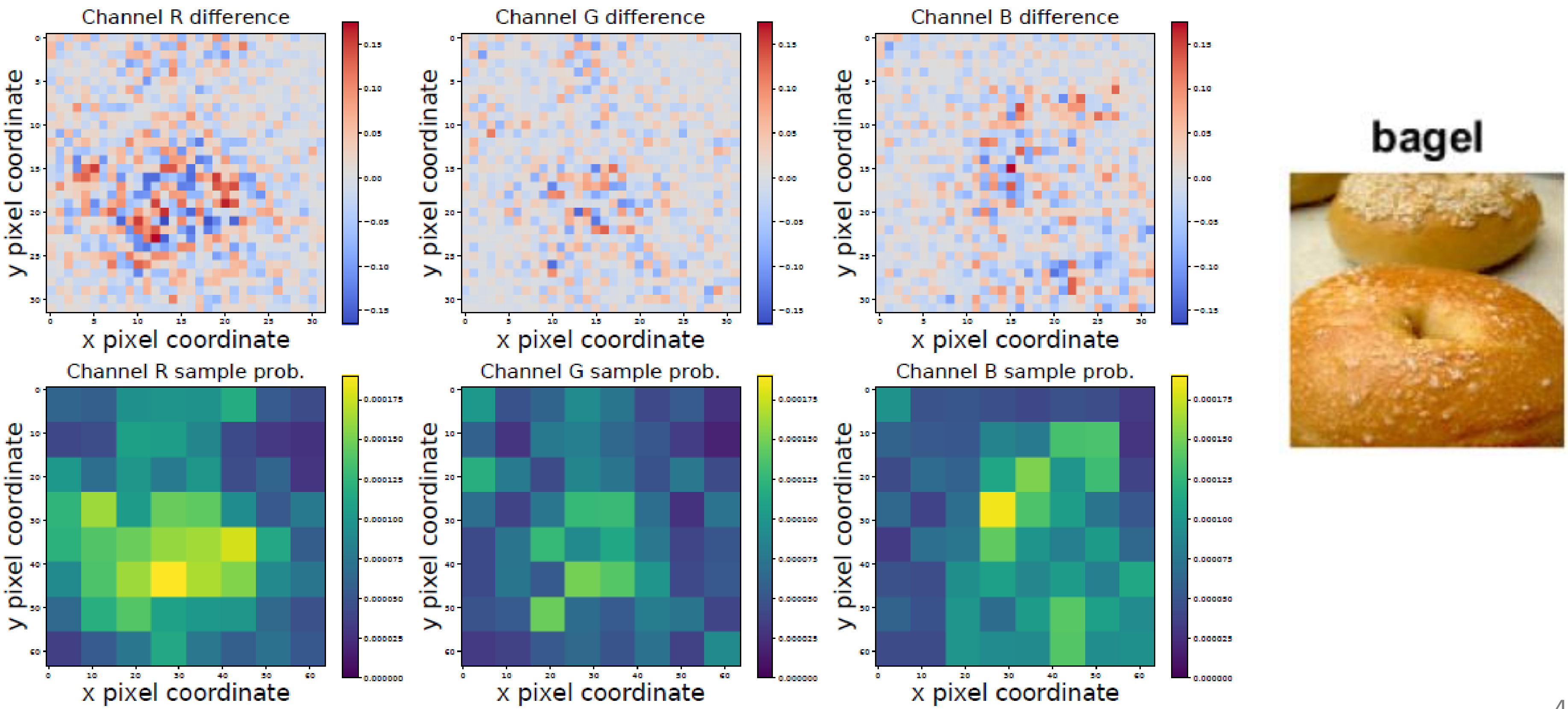
ZOO Attack

- The authors proposed techniques to reduce the number of queries
  - Note that for 28×28 pixels, we need $2 \cdot 784 = 1{,}568$ queries to estimate the gradient
  - Recall that PCA and random sets of pixels were used in Gradient Estimation attack
- The proposed approach starts with reduced resolution, and the resolution is progressively increased (referred to as hierarchical attack)
  - E.g., an original image of a size 299×299 pixels is used
  - Divide the image into 8×8 regions
    - Make only 64 queries to estimate the gradients
    - Optimize until the loss start decreasing
  - Increase to 16×16 regions
    - Make queries and optimize until the loss start decreasing
  - Increase to 32×32 regions
    - Make queries and optimize until the loss start decreasing
  - Repeat until the attack is successful

Queries Reduction

ZOO Attack

- Another technique for query reduction is based on ***importance sampling***
  - Estimate the gradient only for the most important regions in an image
    - Upper figures show the gradient for the Red, Green, and Blue channels
      - E.g., corner pixels are less important for this image, and the changes in R are more important than G and B channels
    - Lower figures shows the most important pixels for R, G, B channels, that are queried first



Experimental Evaluation

ZOO Attack

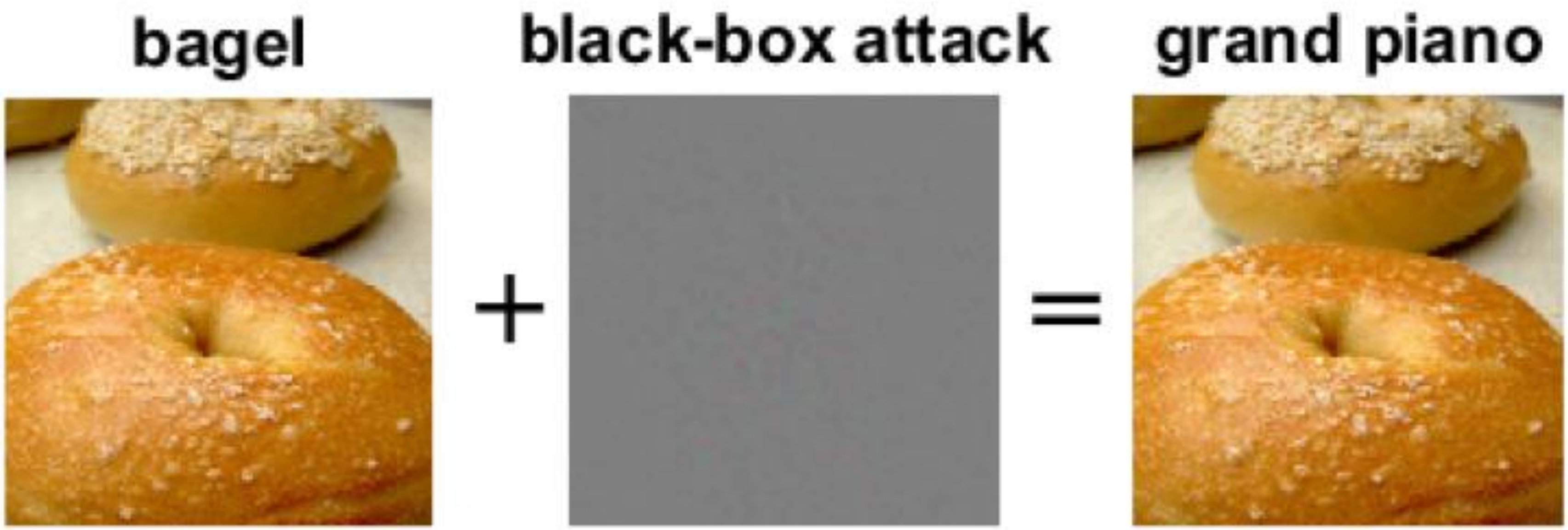
- ImageNet untargeted attack
  - Recall that there are 1,000 classes in ImageNet
  - InceptionV3 model used
  - ZOO attack required about 192,000 queries per image, 20 minutes per image
  - The success rate is lower than C&W white-box attack, but is still high

	Success Rate	Avg. L_2
White-box (C&W)	100 %	0.37310
Proposed black-box (ZOO-ADAM)	88.9 %	1.19916
Black-box (Substitute Model)	N.A.	N.A.

Examples

ZOO Attack

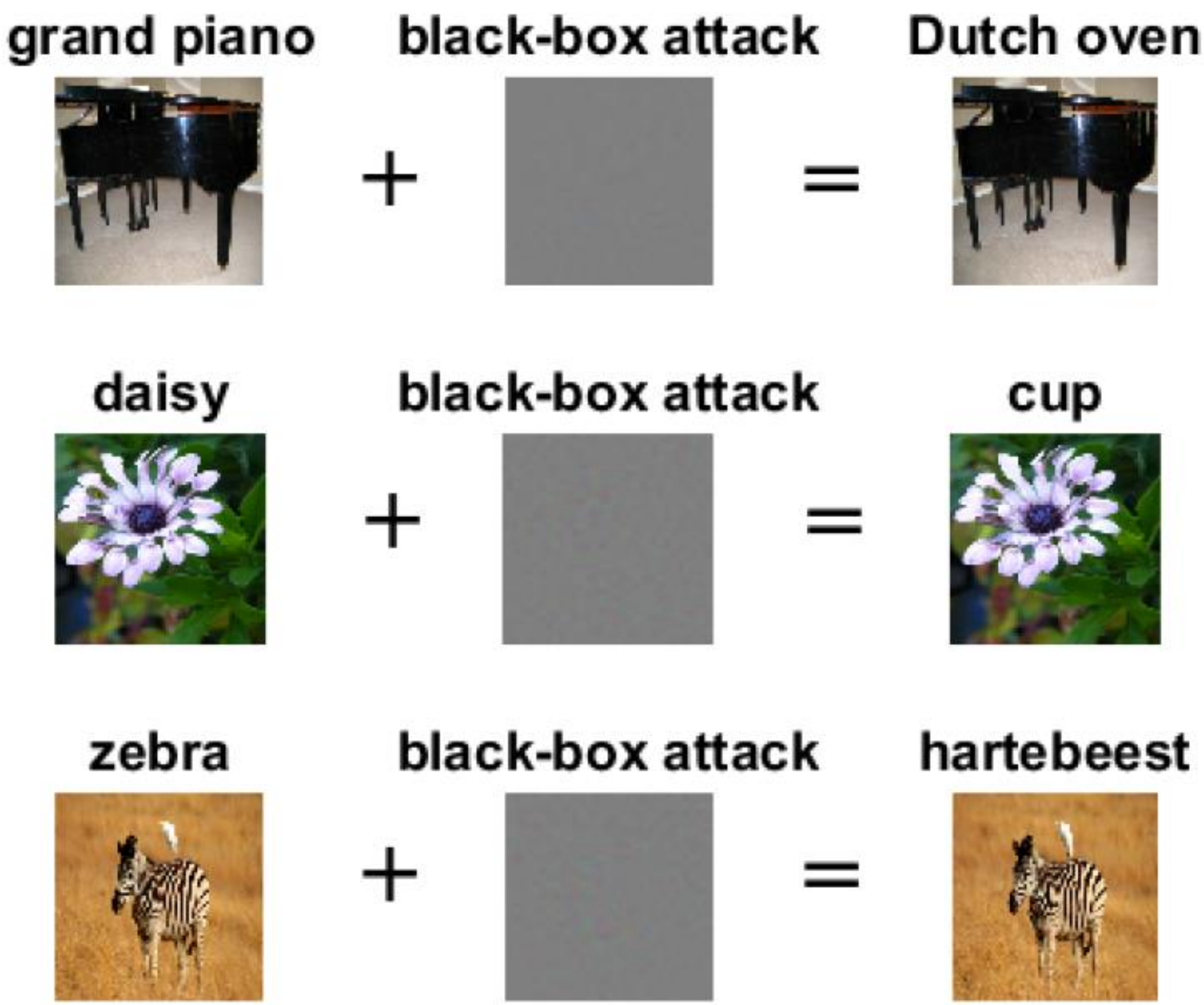
- Targeted attack
  - The added perturbations are imperceptible



Examples

ZOO Attack

- Untargeted attack



Simple Black-box Attack

SimBA Attack

- ***Simple Black-box Attack***
  - [Guo et al. (2019) Simple Black-box Adversarial Attacks]
- A.k.a. SimBA attack
  - Score-based attack (using probability vectors)
  - Focus on query efficiency
  - Both targeted and untargeted attacks were demonstrated
- Approach:
  - Use random orthonormal perturbations for each query
  - Focus on regions in images with high-frequency content to reduce the overall number of queries

Simple Black-box Attack

SimBA Attack

- Steps:
  - Randomly sample perturbation vectors from a predefined orthonormal basis
  - Query the model to obtain the probability score and find out if it is pointing toward or away from the decision boundary
  - Perturb the image by adding or subtracting the perturbation vector
- Goal:
  - Each iteration moves the image away from the original image, and towards the decision boundary

Simple Black-box Attack

SimBA Attack

- Algorithm
  - Random director vectors **q** are sampled, and perturbation with step size ϵ are added or subtracted to misclassify the image

Algorithm 1 SimBA in Pseudocode

procedure $\textsc{SimBA}(\mathbf{x}, y, Q, \epsilon)$
    $\boldsymbol{\delta} = \mathbf{0}$
    $\mathbf{p} = p_h(y \mid \mathbf{x})$
    while $\mathbf{p}_y = \max_{y'} \mathbf{p}_{y'}$ **do**
        Pick randomly without replacement: $\mathbf{q} \in Q$
        for $\alpha \in \{\epsilon, -\epsilon\}$ **do**
            $\mathbf{p}' = p_h(y \mid \mathbf{x} + \boldsymbol{\delta} + \alpha\mathbf{q})$
            if $\mathbf{p}'_y < \mathbf{p}_y$ **then**
                $\boldsymbol{\delta} = \boldsymbol{\delta} + \alpha\mathbf{q}$
                $\mathbf{p} = \mathbf{p}'$
                break
    return $\boldsymbol{\delta}$

Simple Black-box Attack

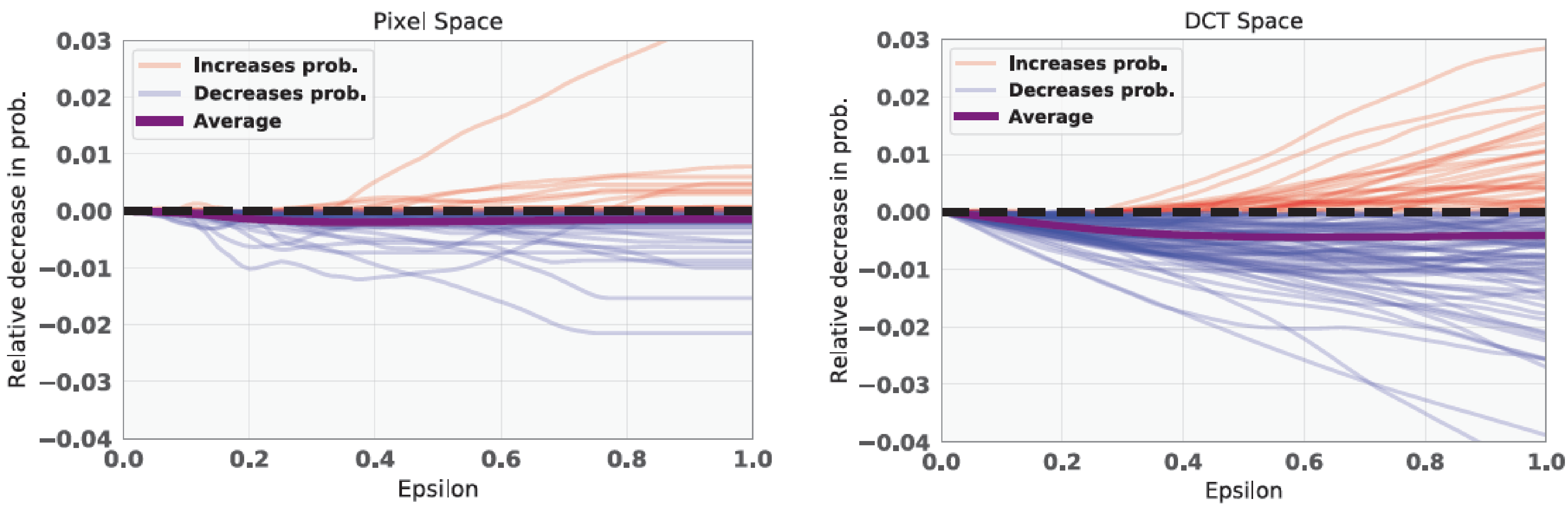
SimBA Attack

- Perturbation vectors are selected to be orthonormal
  - I.e., the random directions for each pixel do not cancel each other out, or amplify each other
- For orthonormal vectors $\mathbf{x}$ and $\mathbf{y}$, their dot product is $\mathbf{x} \cdot \mathbf{y} = 0$
  - The angle between the vectors is 90 degrees
  - I.e., they are orthogonal
- How to choose orthonormal perturbation vector?
  - One inefficient option are the vectors [1,0,0,…,0], [0,1,0,…,0], [0,0,1,…,0],…,[0,0,0,…,1]
    - I.e., only one pixel is changed at a time
  - The authors propose an approach called Discrete Cosine Transform (DCT)
    - It is based on frequency coefficients that correspond to the magnitudes of cosine functions
    - I.e., low-frequency regions in images (e.g., image background) change less at each step
    - Focus on querying high-frequency regions in images

Simple Black-box Attack

SimBA Attack

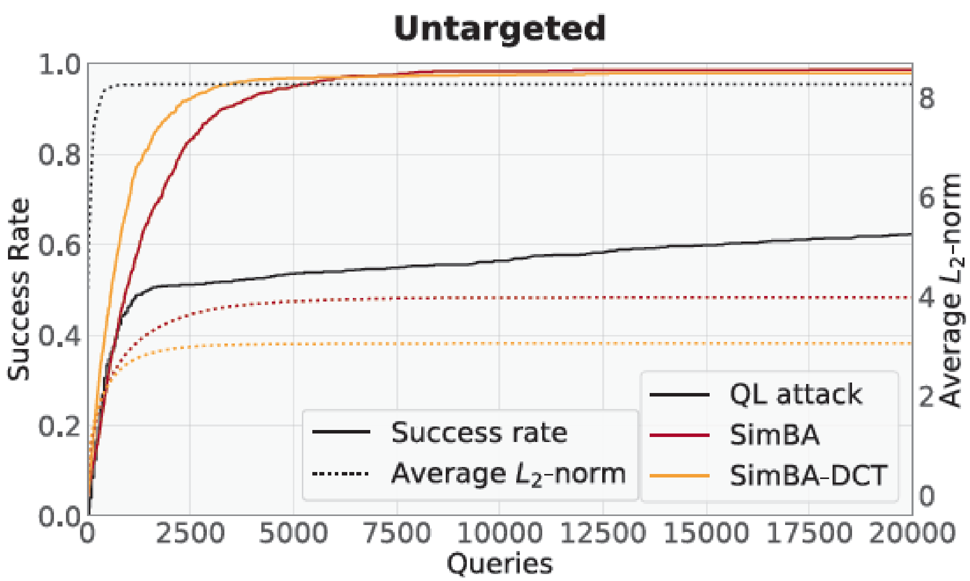
- The average change of the output probability scores is larger when the DCT approach is employed, in comparison to changing individual pixels
  - I.e., SimBA attack with DCT can find perturbations for many pixels in a single query that impact the output probability

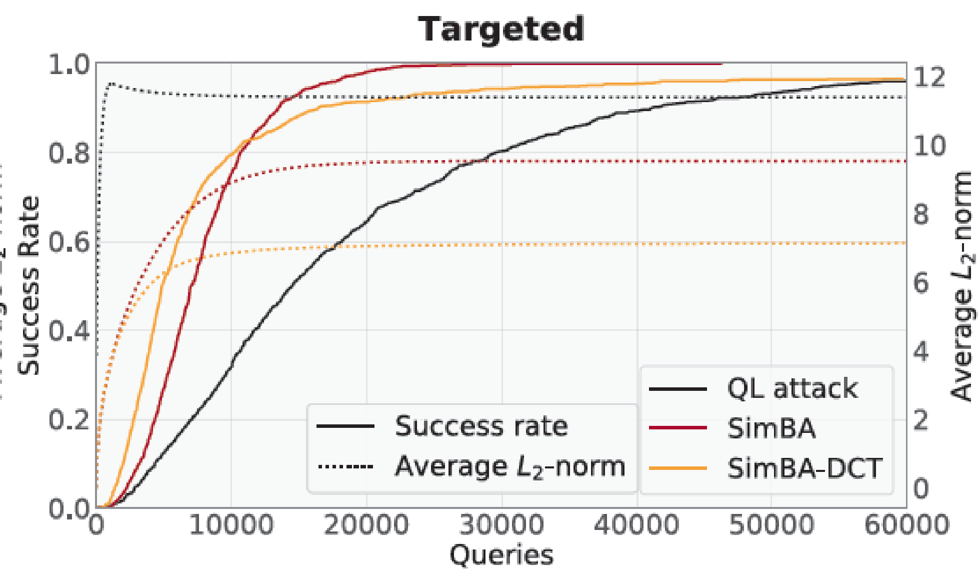


Simple Black-box Attack

SimBA Attack

- Experimental evaluation
  - Full lines display attack success rate, dotted lines display average perturbation
  - SimBA attacks achieved high success rate with small average ℓ_2 norm, and fewer queries





Simple Black-box Attack

SimBA Attack

- Experimental evaluation
  - SimBA achieved good query-efficiency

Untargeted

Attack	**Average queries**	**Average L_2**	**Success rate**
	Label-only		
Boundary attack	123,407	5.98	100%
Opt-attack	71,100	6.98	100%
LFBA	30,000	6.34	100%
	Score-based		
QL-attack	28,174	8.27	85.4%
Bandits-TD	5,251	5.00	80.5%
SimBA	1,665	3.98	98.6%
SimBA-DCT	**1,283**	3.06	97.8%

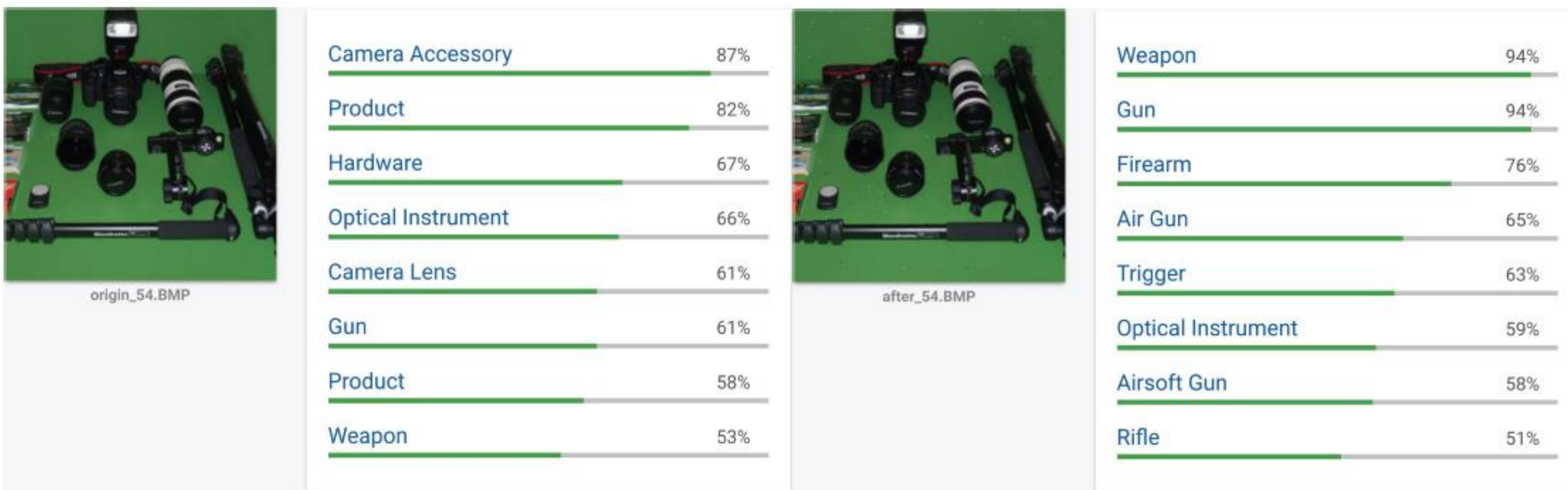
Targeted

Attack	**Average queries**	**Average L_2**	**Success rate**
	Score-based		
QL-attack	20,614	11.39	98.7%
AutoZOOM	13,525	26.74	100%
SimBA	**7,899**	9.53	100%
SimBA-DCT	8,824	7.04	96.5%

Simple Black-box Attack

SimBA Attack

- Attack on [Google Cloud Vision API]()
  - Checked on 50 random images
  - 70% success rate after 5,000 queries



Additional References

1. Nicolae et al. (2019) Adversarial Robustness Toolbox v1.0.0. https://arxiv.org/abs/1807.01069
2. Xu et al. (2019) Adversarial Attacks and Defenses in Images, Graphs and Text: A Review https://arxiv.org/abs/1909.08072