Triangle Attack: A Query-efficient Decision-based Adversarial Attack

Xiaosen Wang[1,2], Zeliang Zhang[1], Kangheng Tong[1], Dihong Gong[2], Kun He[1], Zhifeng Li[2] and Wei Liu[2]
[1] School of Computer Science and Technology, Huazhong University of Science and Technology
[2] Data Platform, Tencent

ECCV TEL AVIV 2022

Introduction

Adversarial Examples pose serious threats to security-sensitive applications, despite the unprecedented progress of Deep Neural Networks (DNNs).

Black-box attack can only access the model output, which is more applicable in **real-world scenarios**. Among the black-box attacks, **decision-based attack** is more challenging and practical due to the minimum information requirement for attack.

Background: existing decision-based attacks need to restrict adversarial samples on the decision boundary or estimate the gradient at each iteration, which leading to **inefficiency** for query.

Related Work

- BoundaryAttack [Brendel et al., 2018] initializes a large perturbation and performs random walks on the decision boundary while keeping adversarial.
- Recent works adopt various gradient estimation strategies to efficiently optimize the perturbation, such as HSJA [Jianbo et al., 2020], QEBA [Huichen et al., 2020], GeoDA [Ali et al., 2020], etc.
- Surfree [Thibault et al., 2021]: Iteratively construct a circle on the decision boundary and adopt binary search to find the intersection of the constructed circle and decision boundary as the adversary without any gradient estimation.

Methodology

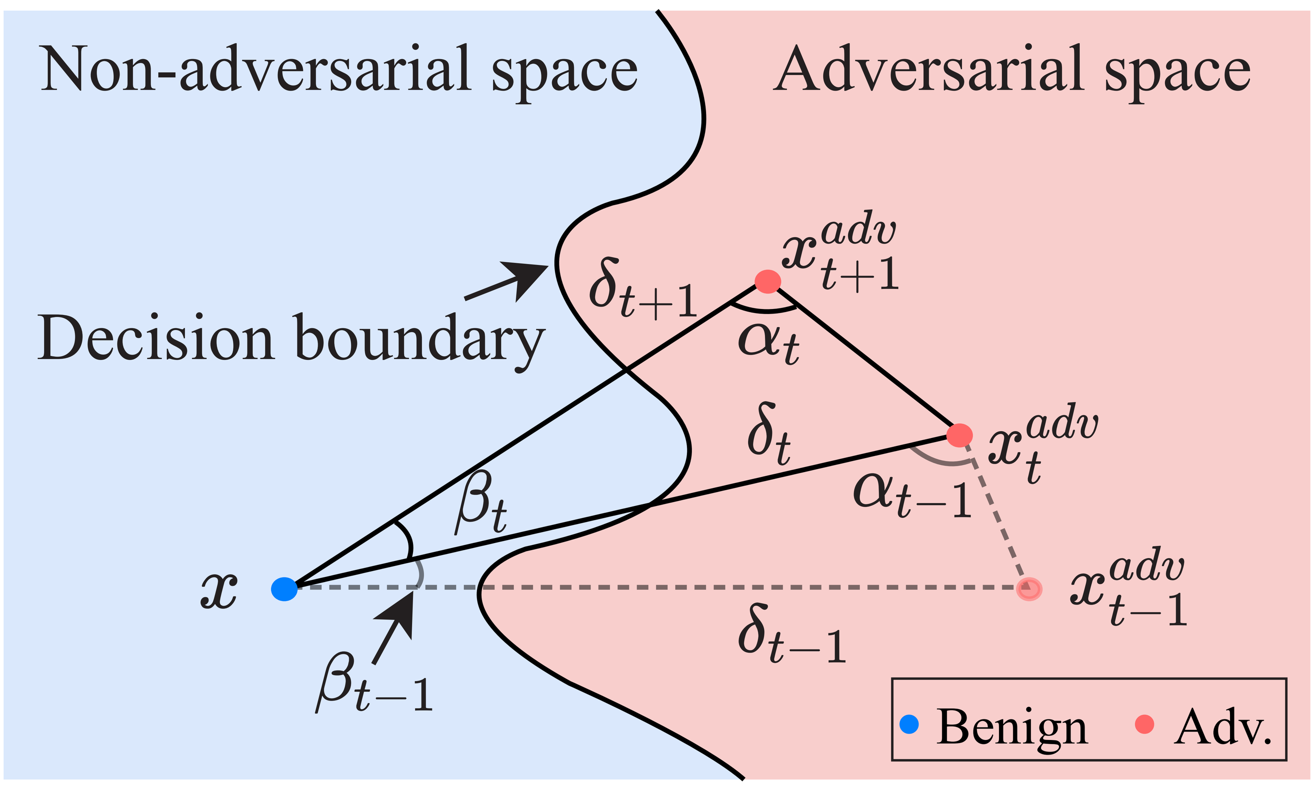


Figure 1: Illustration of the optimization procedure of TA.

- **Motivation**: At the t-th iteration, x, x_t^{adv} and x_{t+1}^{adv} can naturally construct a triangle for any iterative attacks. Could we utilize such triangle for efficient attack?
- **Optimization**: Based on the **law of sines**, To decrease the perturbation ($\delta_t > \delta_{t+1}$), we should guarantee $\pi - (\alpha_t + \beta_t) < \alpha_t$.

Methodology (cont.)

- **Sampling the 2-D subspace S of frequency space**: Thanks to the generality of the geometric property, we optimize the perturbation in the **low frequency space**.
- **Searching the candidate triangle**. x, x_t^{adv}, α and β_t can determine a triangle to find the next adversarial example x_{t+1}^{adv}. α is a learnable angle and we conduct binary search to find the angle β.
- **Adjusting** α: With the same angle β, a smaller angle α makes it easier to find an adversarial example while a larger angle α leads to smaller perturbation. Hence, we **increase or decrease** the angle α based on if it leads to adversarial example.

Algorithm

Algorithm 1 Triangle Attack

Input: Target classifier $\boldsymbol{f}$ with parameters $\boldsymbol{\theta}$; Benign sample $\boldsymbol{x}$ with ground-truth label $\boldsymbol{y}$; Maximum number of queries Q; Maximum number of iteration N for each sampled subspace; Dimension of the directional line d; Lower bound $\underline{\beta}$ for angle β.

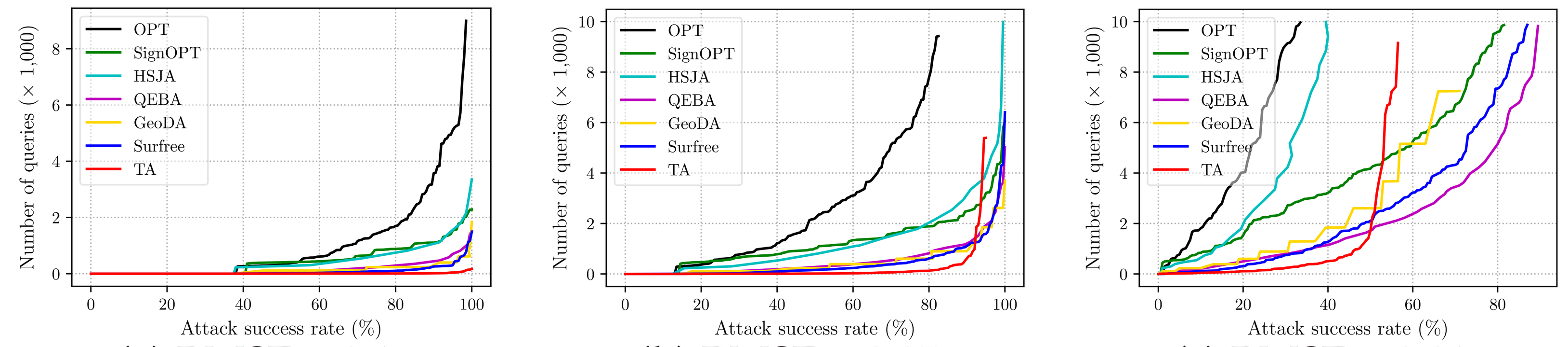
Output: An adversarial example x^{adv}.

Initialize a large adversarial perturbation δ_0;
$x_0^{adv} = x + \delta_0$, $q = 0$, $t = 0$, $\alpha_0 = \pi/2$;
while $q < Q$ **do**
  Sampling 2-D subspace $\mathcal{S}_t$ in the low frequency space;
  $\beta_{t,0} = \max(\pi - 2\alpha, \underline{\beta})$;
  if $f(\mathcal{T}(x, x_t^{adv}, \alpha_{t,0}, \beta_{t,0}, \mathcal{S}_t); \theta) = f(x; \theta)$ **then**
    $q = q + 1$, update $\alpha_{t,0}$
    if $f(\mathcal{T}(x, x_t^{adv}, \alpha_{t,0}, -\beta_{t,0}, \mathcal{S}_t); \theta) = f(x; \theta)$ **then**
      $q = q + 1$, update $\alpha_{t,0}$
      Continue; ▷ give up this subspace
    end if
  end if
  $\overline{\beta}_{t,0} = \min(\pi/2, \pi - \alpha)$;
  for $i = 0 \to N$ **do** ▷ binary search for angle β
    $\beta_{t,i+1} = (\overline{\beta}_{t,i} + \beta_{t,i})/2$;
    if $f(\mathcal{T}(x, x_t^{adv}, \alpha_{t,i}, \beta_{t,i+1}, \mathcal{S}_t); \theta) = f(x; \theta)$ **then**
      $q = q + 1$, update $\alpha_{t,i}$
      if $f(\mathcal{T}(x, x_t^{adv}, \alpha_{t,i}, -\beta_{t,i+1}, \mathcal{S}_t); \theta) = f(x; \theta)$ **then**
        $\overline{\beta}_{t,i+1} = \beta_{t,i+1}, \beta_{t,i+1} = \beta_{t,i}$;
      end if
    end if
    $q = q + 1$, update $\alpha_{t,i+1}$
  end for
  $x_{t+1}^{adv} = \mathcal{T}(x, x_t^{adv}, \alpha_{t,i+1}, \beta_{t,i+1}, \mathcal{S}_t)$, $t = t + 1$;
end while
return x_t^{adv}.

Experiments

Model	VGG-16			Inception-v3			ResNet-18			ResNet-101			DenseNet-121		
RMSE	0.1	0.05	0.01	0.1	0.05	0.01	0.1	0.05	0.01	0.1	0.05	0.01	0.1	0.05	0.01
OPT	76.0	38.5	5.5	34.0	17.0	4.0	67.0	36.0	6.0	51.5	21.0	5.0	51.5	29.0	5.5
SignOPT	94.0	57.5	12.5	50.5	27.0	8.0	84.5	49.5	13.0	69.0	33.0	8.0	69.5	44.0	10.0
HSJA	92.5	58.5	13.0	32.5	14.0	4.0	83.0	51.0	12.5	71.5	37.5	12.0	70.5	43.5	10.5
QEBA	98.5	86.0	29.0	78.5	54.5	17.0	98.0	81.5	34.5	94.0	59.0	20.5	91.0	66.0	24.0
BO	96.0	72.5	17.0	75.5	43.0	10.0	94.5	74.0	16.0	89.5	63.0	16.5	93.0	64.5	16.5
GeoDA	99.0	94.0	35.0	89.0	61.5	23.5	99.5	90.0	30.5	98.0	81.5	22.0	**100.0**	84.5	27.5
Surfree	99.5	92.5	39.5	87.5	67.5	24.5	98.5	87.0	36.0	95.5	76.5	27.0	97.0	78.0	29.0
TA (**Ours**)	**100.0**	**95.0**	**44.5**	**96.5**	**81.5**	**30.0**	**100.0**	**94.0**	**51.5**	**99.0**	**88.5**	**40.0**	99.5	**92.5**	**43.5**

Table 1: Attack success rate (%) under different RMSE thresholds.



(a) RMSE = 0.1 (b) RMSE = 0.05 (c) RMSE = 0.01

Figure 2: Attack success rate under various number of query.

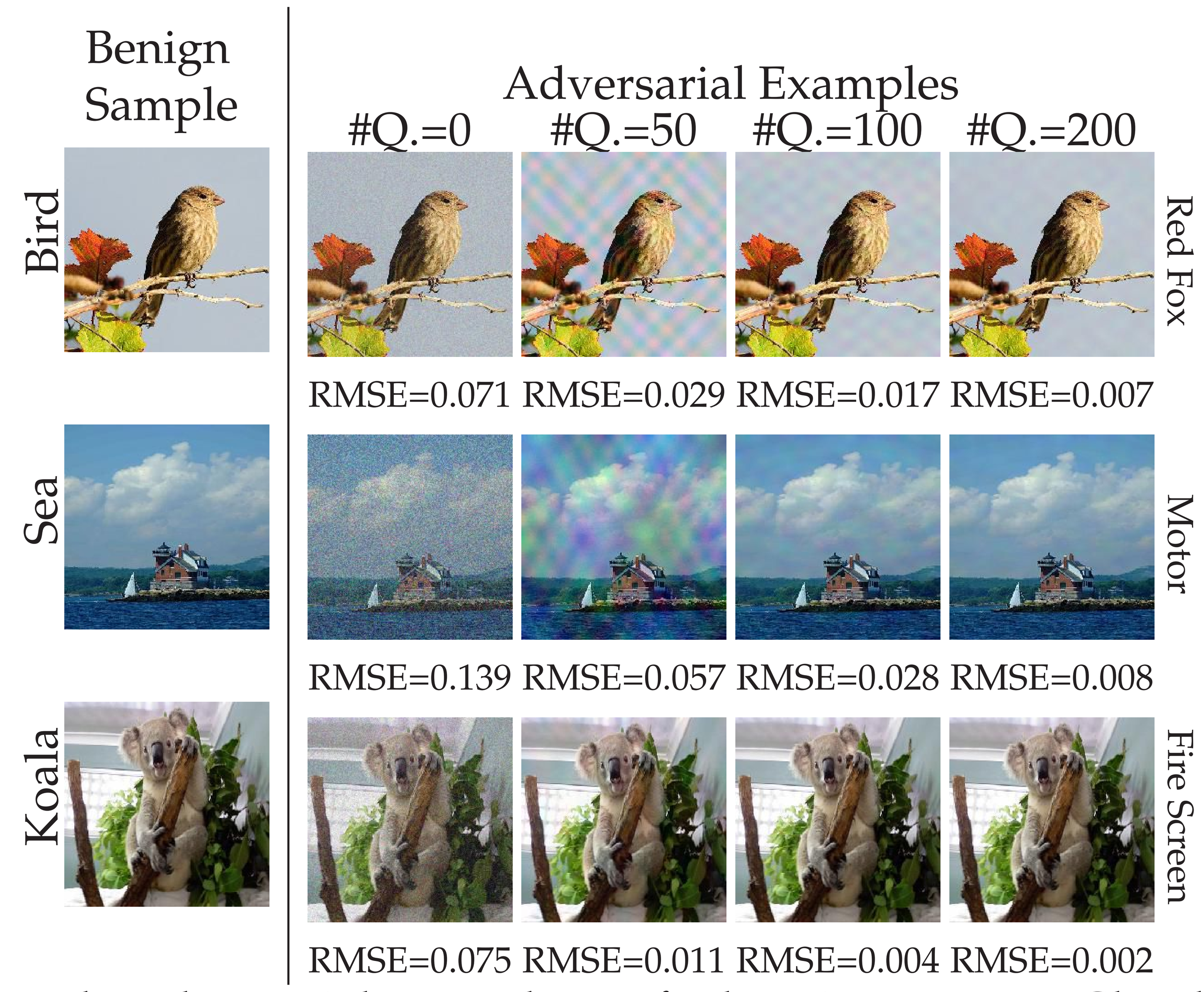


Figure 3: The adversarial examples crafted against Tencent Cloud API.

Conclusion

- Propose a novel decision-based attack, called Triangle Attack (TA), which utilizes the geometric information that **the longer side is opposite the larger angle in any triangle**.
- Optimize the adversarial perturbation in **the low frequency space** generated by DCT with much lower dimensions than the input space, to **significantly improve the query efficiency**.
- Achieve a **much higher attack success rate** within 1,000 queries and need **much less queries** for the same attack success rat.
- Demonstrate **the practical applicability on Tencent Cloud API**.