BRUSLEATTACK: A QUERY-EFFICIENT SCORE- BASED BLACK-BOX SPARSE ADVERSARIAL ATTACK

THE UNIVERSITY of ADELAIDE
Viet Quoc Vo, Ehsan Abbasnejad, Damith C. Ranasinghe

ICLR 2024

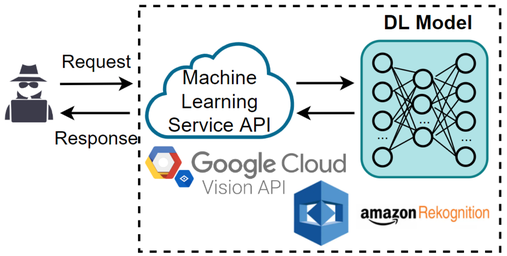
Project page: https://brusliattack.github.io/

SCAN ME

Introduction

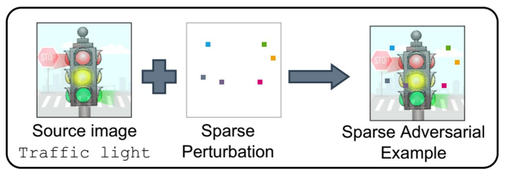
Threat Model

- In real-world systems, the model is hidden from users except for the access to the model's outputs **(*a black-box*)**, e.g. confidence scores.
- This is a pragmatic threat scenario for operational systems, as it underscores the potential for adversaries to exploit even the minimal information available for launching attacks. Thus, we are interested in exploring the vulnerability of models in this context.



Sparse Attacks

Sparse attacks pose an insidious threat as they aim to perturb only **a minimal number of pixels** in an image (l_0 norm) to **mislead a model**. However, sparse attacks have received relatively limited attention in research, we are motivated to explore highly query-efficient sparse attack strategies.



Challenges

1. **The search space (Width × Height × Channels × Color values) is incredibly enormous** to discover a minimum number of pixels. Hence, with limited information from the model's responses, attacks require excessive queries, which is undesirable.

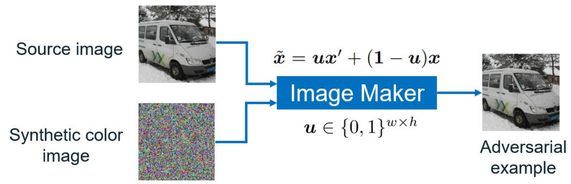
2. The **NP-hard problem** [1, 2]
   - **Discrete** and **non-differentiable search space** (mixed discrete and continuous) [3].

Problem Formulation and *BruSLeAttack* Methodology

1 Reduce the search space

Problem formulation, $\boldsymbol{x}^* = \underset{\tilde{\boldsymbol{x}}}{\operatorname{argmin}} L(f(\tilde{\boldsymbol{x}}), y_{target})\ s.t.\ \|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|_0 < B$ **,** leads to an *enormous search space*.

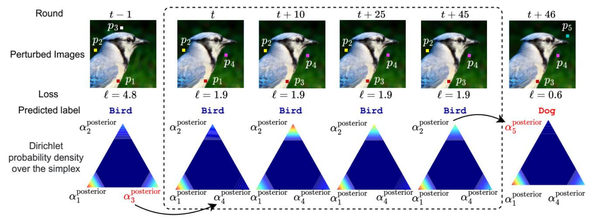
Our **IDEA** is to search and replace pixels in a **source image** with their *corresponding pixels* in a **synthetic color image** so that the malicious loss is reduced, in contrast to attempting to search for pixel values and locations.



Then, we introduce a **new formulation**: $\boldsymbol{u}^* = \underset{\boldsymbol{u}}{\operatorname{argmin}}\, \ell(\boldsymbol{u})\ \ s.t.\ \|\boldsymbol{u}\|_0 < B,$

where $\ell(\boldsymbol{u}) = L(f(\tilde{\boldsymbol{x}}), y_{target})$, a binary matrix u *is used to encode an adversarial example.* Elements 0 and 1 denote pixels from the source and synthetic color images, respectively.

2 Remedy the NP-hard problem and handle a non-differentiable search space



Intuition: If a pixel has more influence on the model's decision, replacing it is more likely to increase the loss. Thus, it should be less likely to be replaced to circumvent a bad solution. Thus, we employ the **Bayesian Framework** and **history of pixel manipulation** to learn this influence and search for adversarial pixels.

Bayesian Framework

1. Prior: $\mathrm{P}(\theta, \alpha) = \mathrm{Dir}(\alpha)$

2. Sampling $u^{(t)}$:

$v_1^{(t)}, \dots v_b^{(t)} \sim Cat(v|\theta^{(t)}, u^{(t-1)} = 1)$

$q_1^{(t)}, \dots q_{B-b}^{(t)} \sim Cat(q|\theta^{(t)}, u^{(t-1)} = 0)$

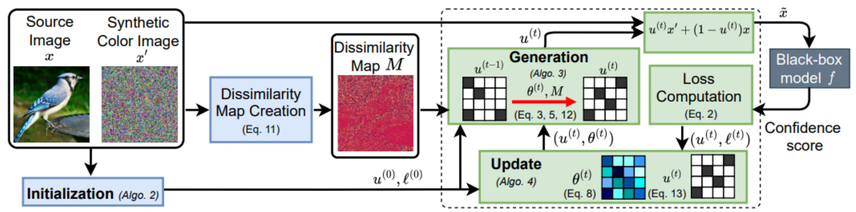
$u^{(t)} = [\mathrm{V}_{k=1}^{b} v_k^{(t)}] \mathrm{V} [\mathrm{V}_{r=1}^{B-b} v_r^{(t)}]$

3. Update $\theta^{(t)}$:

$\alpha_{i,j}^{\text{posterior}} = \alpha_{i,j}^{\text{prior}} + s_{i,j}^{(t)}$

$\mathrm{P}(\theta|\alpha, u^{(t-1)}, \ell^{(t-1)}) = \mathrm{Dir}(\alpha^{\text{posterior}})$

$\theta^{(t)} = \mathbb{E}_{\theta \sim \mathrm{P}(\theta|\alpha, u^{(t-1)}, \ell^{(t-1)})}[\theta]$

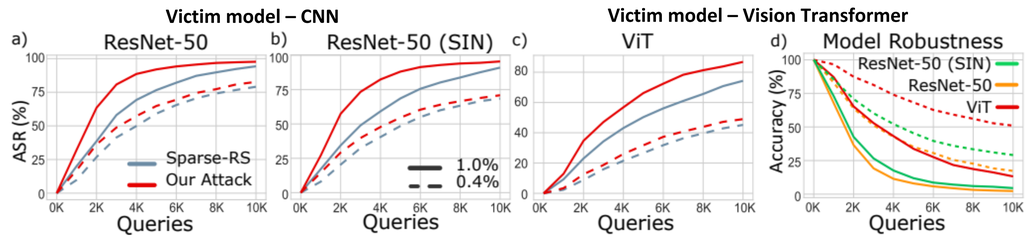


BruSLeAttack Algorithm

1. Create a **dissimilarity map M** between the source and the synthetic color images.
2. Initialize some solutions randomly and choose the best for $u^{(0)}$.
3. Sample new $u^{(t)}$ based on $\theta^{(t)}$, $u^{(t-1)}$and M. Then craft an adversarial image $\tilde{x}$ from $u^{(t)}, x$ and x'.
4. Query a black-box model f and calculate loss $\ell^{(t)}$.
5. Update both $\theta^{(t)}$ and $u^{(t)}$ based on the change in the loss and the current solution $u^{(t)}$.

Results

Attack Transformers & Convolutional Nets



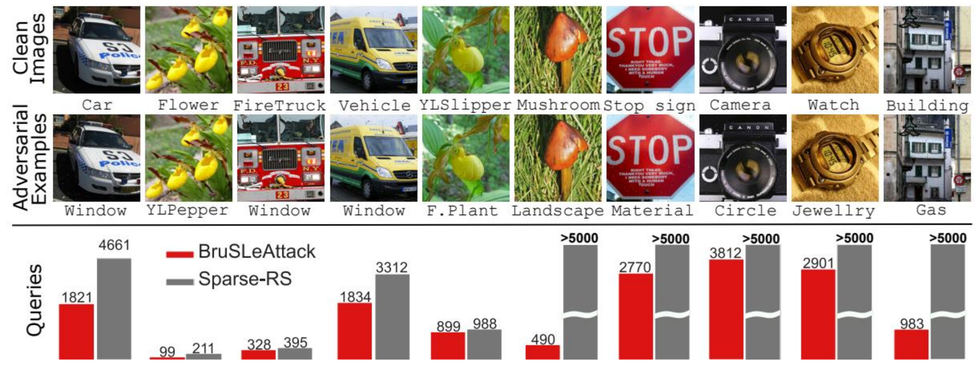
- **Query Efficiency**: Within 10K queries, **BruSLeAttack** outperforms state-of-the-art Sparse-RS [4].
- **Attack Success Rate (ASR,** up to 10K queries): **BruSLeAttack** can achieve a much higher ASR than Sparse-RS across different query budgets.

Attack Defended Models

Sparsity	Undefended Model		l_∞-AT		l_2-AT		RND	
	SPARSE-RS	BRUSLEATTACK	SPARSE-RS	BRUSLEATTACK	SPARSE-RS	BRUSLEATTACK	SPARSE-RS	BRUSLEATTACK
0.04%	33.6%	**24.0%**	43.8%	**42.2%**	89.8%	**88.4%**	90.8%	**85.0%**
0.08%	13.2%	**6.8%**	26.8%	**24.4%**	81.2%	**79.2%**	82.2%	**72.6%**
0.12%	7.6%	**2.6%**	19.0%	**18.4%**	75.8%	**73.8%**	73.6%	**61.0%**
0.16%	5.2%	**1.0%**	16.6%	**14.8%**	71.4%	**69.2%**	64.8%	**51.4%**
0.2%	4.6%	**1.0%**	12.2%	**11.8%**	68.4%	**66.4%**	56.8%	**42.6%**

BruSLeAttack consistently outweighs Sparse-RS against different defense methods and sparsity levels.

Attack Real-world System - Google Cloud Vision



BruSLeAttack is more query efficient than State-of-the-art Sparse-RS.

Conclusions

- **BruSLeAttack** is able to remedy the NP-hard problem
- **BruSLeAttack** is capable of handling a discrete and non-differentiable search space.
- **BruSLeAttack** is more query-efficient than Sparse-RS.

References

[1] Modas and P. Moosavi-Dezfooli, S. Frossard. Sparsefool: a few pixels make a big difference. CVPR 2019.
[2] X. Dong, D. Chen, J. Bao, C. Qin, L. Yuan, W. Zhang, N. Yu, and D. Chen. GreedyFool: DistortionAware Sparse Adversarial Attack, NeurIPS, 2020.
[3] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. IEEE SSP, 2017.
[4] Croce F., Andriushchenko M., Singh N. D., Flammarion N., and Hein M. Sparse-RS: A Versatile Framework for Query-Efficient Sparse Black-Box Adversarial Attacks, AAAI 2022.