Adversarial Attacks on Aerial Imagery

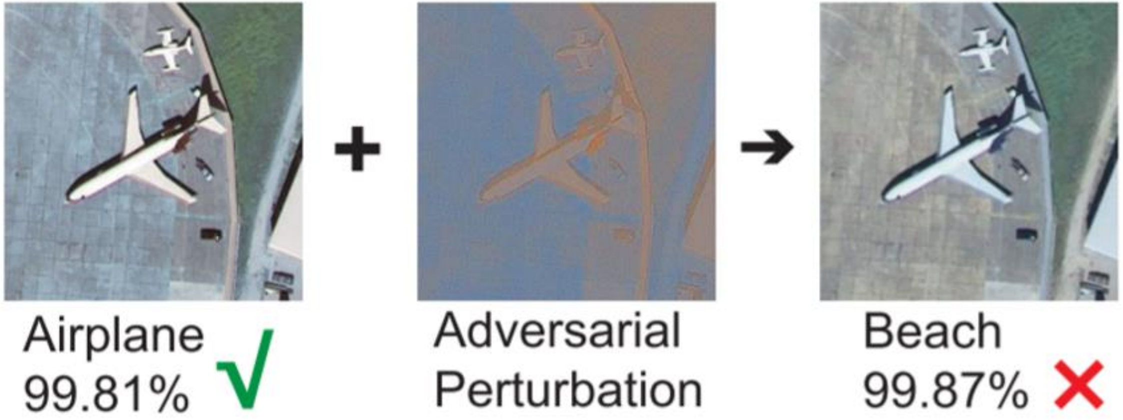
Made By:-
Shorya Sharma
IIT Bhubaneswar

Introduction: Aerial Scene Classification

- Aerial scene classification is the foundation and important technology of ground object detection, land use management and geographic analysis
- RSIs with higher resolution have led to the wide use of RSI scene classification systems in crop classification, forest resource surveys, land cover classification, building detection, and other fields.
- During recent years, convolutional neural networks (CNNs) have achieved significant success and are widely applied in RSI scene classification.
- Automatic interpretations of RSIs with robust and high accuracy can deliver high economic efficiency.

The Adversarial Problem

- CNNs have many limitations and still have room for improvements, especially concerning security problems. This happens when we deliberately add noises to the input images to fool the CNN classifier and prompt it to make wrong predictions with high confidence. The modified images are called adversarial examples.
- Goodfellow et al. stated that adversarial examples in CNNs are produced due to the linear operations of the model in a high-dimensional space.
- The classification problem of RSIs presents a great security risk, especially for applications in the military and automatic driving fields.



On the left is the original airplane. The RSI scene classification system can correctly classify it as an airplane with confidence of 99.81%. However, after adding an adversarial perturbation to the image, the RSI scene classification system can classify it as a beach with 99.87% confidence. The latter result is wrong, and we refer to the modified image as an adversarial example. Obviously, the adversarial example cannot affect human classification, but it leads to system errors and has serious consequences.

Adversarial Attack Algorithm

- We first generate a classification model F from a clean training set and give a legitimate input x. The correct label for sample x is y.
- If x' obtained by adding a small distortion r to the instance x can cause F to obtain the wrong classification result $F(x') \neq y$, we will refer to x' as an un-targeted adversarial example; If the attacker presents a target label $T \neq y$ and misleads the classification result of x' to $F(x') = T$, while the added distortion r is less than a certain threshold (the added distortion is small enough), we will refer to x' as a targeted adversarial example.
- The amount of distortion is an important measure of the quality of adversarial example. The smaller the distortion of adversarial example, the closer it is to the original example, the more difficult it is to detect and recognize.
- Most adversarial examples attacks use an L_p distance to define closeness,defined as $\|v\|_p = (\sum_{i=1}^{n} |v_i|^p)^{\frac{1}{p}}$. There are three commonly used L_p distances: L_0 distance represents the number of pixels that change; L_2 distance measures the standard Euclidean distance between x and x'; L_∞distance measures the maximum change to any of the coordinates. Different attack algorithms may use different L_p distances to define closeness

Classification of Adversarial Attacks

- The attack algorithms of adversarial examples are also diverse. According to the number of iterations, attack algorithms can be divided into the **one-step attack** and **iterative methods**. These attack algorithms are **gradient-based optimization methods** that try to maximize the loss of the objective function
- According to the characteristics of the attack algorithm, Premlatha et al. and Chakraborty et al. classified attack scenarios into three categories: **evasive attacks**, **poisoned attacks** and **exploratory attacks**. Attackers in an evasive attack attempt to avoid detection by the system by constructing malicious input. In this scenario, the attacker cannot influence the training data of the model. The attacker in a poisoned attack scenario attempts to inject constructed malicious data to poison the model during the training process. In the exploratory attack scenario, the attacker cannot affect the training dataset or the details of the model and acquires knowledge of the learning algorithms and training data by testing the response of the model to the input data, thereby constructing effective adversarial examples

- Attack algorithms can also be divided into other types. Attack algorithms can be divided into **white-box attacks** and **black-box attacks** according to the model information acquired by the attacker. In a white-box attack, the attacker knows the type of CNN, the number of layers, and the optimization method used for training and can obtain the training data and even discern the hyperparameters of the model.
- According to whether the attack class is clear, attack algorithms are also divided into **targeted attacks** and **untargeted attacks**. Targeted attacks attempt to make the model classify the adversarial example as a specific class, whereas untargeted attacks do not care about the prediction labels of adversarial examples. Untargeted attacks require only the model to misclassify data that were originally correctly classified.

Our Contributions

- We implemented white-box adversarial attacks using 5 different neural network architectures on different aerial scene classification datasets under two different configurations: eps=0.0005 and eps=1.0.
- We tabulated our results and inferred the best-attack on each dataset using a particular network architecture. Confusion Matrix were also plotted for evaluation purposes.
- We implemented the Transferable Sparse Adversarial Attack(TSAA) and deployed it on our dataset to infer the transferability of this black-box attack on our datasets using different neural network architectures.
- We implemented state-of-the-art black box attacks on our datasets using Adversarial Attack libraries.

Literature Survey

- Attack Selectivity of Adversarial Examples in Remote Sensing Image Scene Classification
- An Empirical Study of Adversarial Examples on Remote Sensing Image Scene Classification
- Project Gradient Descent Adversarial Attack against Multisource Remote Sensing Image Scene Classification
- Assessing the Threat of Adversarial Examples on Deep Neural Networks for Remote Sensing Scene Classification: Attacks and Defenses
- Adversarial Examples in Remote Sensing
- Generating natural adversarial Remote Sensing Images

FoolBox

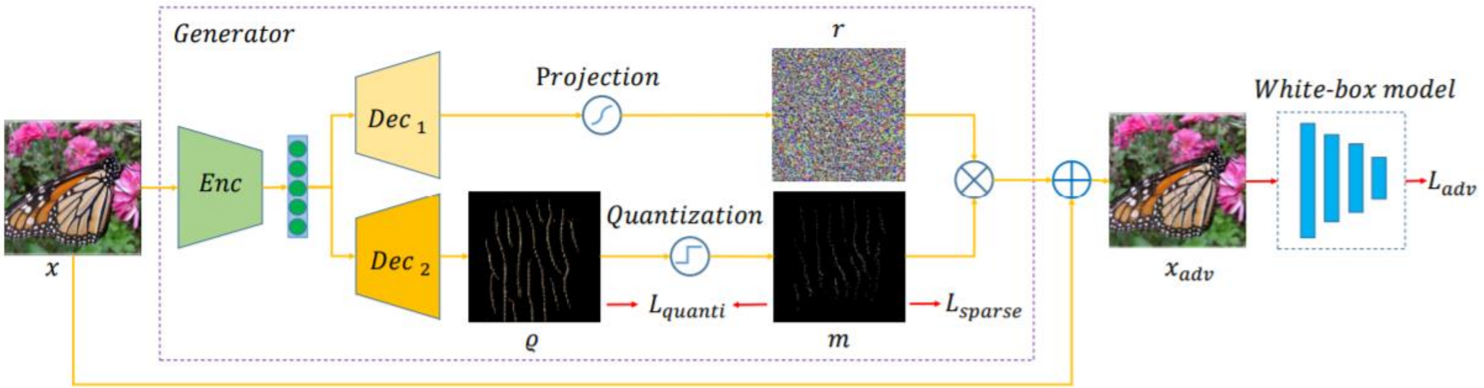
- FoolBox is a Python library that lets you easily run adversarial attacks against machine learning models like deep neural networks. It is built on top of EagerPy and works natively with models in PyTorch, TensorFlow, and JAX.
- Foolbox provides a large collection of state-of-the-art gradient-based and decision-based adversarial attacks.
- Crafting adversarial examples requires five elements:
  - ❑ First, a model that takes an input (e.g. an image) and makes a prediction (e.g. class-probabilities).
  - ❑ Second, a criterion that defines what an adversarial is (e.g. misclassification). Third, a distance measure that measures the size of a perturbation (e.g. L1-norm).
  - ❑ Finally, an attack algorithm that takes an input and its label as well as the model, the adversarial criterion and the distance measure to generate an adversarial perturbation

- **FoolBox Attacks**: Foolbox implements a large number of adversarial attacks, each attack takes a model for which adversarials should be found and a criterion that defines what an adversarial is. The default criterion is misclassification.
  - ❑ **Gradient-Based Attacks**: Gradient Attack, Gradient Sign Attack (FGSM), Iterative Gradient Attack, Iterative Gradient Sign Attack, DeepFool L2 Attack, DeepFool L∞ Attack, L-BFGS Attack, SLSQP Attack, Jacobian-Based Saliency Map Attack
  - ❑ **Score-Based Attacks**: Single Pixel Attack, Local Search Attack, Approximate L-BFGS Attack
  - ❑ **Decision-Based Attacks**: Boundary Attack, Pointwise Attack, Additive Uniform Noise Attack, Additive Gaussian Noise Attack, Salt and Pepper Noise Attack, Contrast Reduction Attack, Gaussian Blur Attack, Precomputed Images Attack

- **Foolbox Distances:** Distance measures are used to quantify the size of adversarial perturbations. FoolBox implements Mean Squared Distance, Mean Absolute Distance, L∞, L0

Transferable Sparse Adversarial Attack(TSAA)

- TSAA is a sparse adversarial attack based on the L0 norm constraint, which can succeed by only modifying a few pixels of an image.
- Prior sparse attack methods achieve a low transferability under the black-box protocol because they methods rely on the target model's accurate gradient information or its approximation, causing the generated adversarial examples overfitting the target model. TSAA is a trainable generator-based architecture which tends to alleviate the overfitting issue by learning to translate a benign image into an adversarial example and thus efficiently craft transferable sparse adversarial examples.



Framework:-

- In the proposed framework, the origin image is fed into the generator and the output adversarial image is fast obtained through only one feedforward inference without gradient backpropagation. The significant difference from previous generator-based methods is the sparsity of perturbations.
- A generator is designed to translate a benign image into an adversarial image. Denote the generator as G, the adversarial image is crafted by xadv = x+G(x). The generator mainly includes one encoder and two decoder branches.
- TSAA decouples the adversarial perturbation into two components which control distortion magnitude and perturbed pixel location respectively.
- Once G is trained on the training data and the white-box model, it can produce perturbations for any input instance to perform a transfer attack

Datasets used for Adversarial Attacks

❖ NWPU-RESEIC45

RESISC45 dataset is a publicly available benchmark for Remote Sensing Image Scene Classification (RESISC), created by Northwestern Polytechnical University (NWPU). This dataset contains **31,500** images, covering **45** scene classes with **700** images in each class.

❖ UC Merced Land Use Dataset

UC Merced is a **21** class land use remote sensing image dataset, with **100** images per class. The dataset contains **2100** images which were manually extracted from large images from the USGS National Map Urban Area Imagery collection for various urban areas around the country. The pixel resolution of this public domain imagery is **0.3** m.

Experiments Details

- Deep Learning Framework: Pytorch
- GPU: Tesla M60(batch size=16)
- Neural Network Architectures:
  - Alex Net
  - ResNet50
  - ResNet101
  - MobileNetV2
  - DenseNet121
- Optimizer: Adam Optimiser(learning rate=0.01, weight decay rate= 0.001)
- Loss Function: Cross Entropy Function
- Epochs: 100
- Train Set : Test Set :: 80:20
- Epsilon: 0.0005; 1.0
- Evaluation Metrics: Accuracy, Confusion Matrix

Results

- **NWPU-RESIC45**
  - White-Box Attacks
    - Epsilon=0.0005

Attack No.	Attack Name	AlexNet	Resnet50	Resnet101	MobileNet_v2	DensetNet
	Original Accuracy	0.747058824	0.885373609	0.892686804	0.894117647	0.914149444
	Best Attack Accuracy	0.185	0.479	0.381	0.573	0.449
	Percentage of Performance Drop	**75.23622047**	45.89854552	57.31985752	35.91447368	50.88330435
1	L2 Contrast Reduction Attack	0.206999958	0.582499981	0.55249998	0.604999989	0.653999984
2	Virtual Adversarial Attack	**0.185499966**	0.566999972	0.567499965	0.605999976	0.655499995
3	DDNA Attack	0.204999983	0.565999985	0.566499978	0.597499967	0.660999984
4	L2 Projected Gradient Descent Attack	0.20449996	0.549499989	0.56249997	0.602999985	0.648499995
5	Linf Projected Gradient Descent Attack	0.20449996	0.552999973	0.561499983	0.592499971	0.661999971
6	L2 Basic Iterative Attack	0.194999933	0.57249999	0.563999981	0.583499968	0.67049998
7	Linf Basic Iterative Attack	0.20449996	0.555999964	0.554499984	**0.573499978**	0.631499976
8	L2 Fast Gradient Attack	0.21299994	0.555999964	0.557999969	0.627999991	0.66049999
9	Linf Fast Gradient Attack	0.201499939	0.563999981	0.551999986	0.602499992	0.660999984
10	L2 Repeated Additive Gaussian Noise	0.21299994	0.552999973	0.552999973	0.593499988	0.664499998
11	L2 Repeated Additive Uniform Noise	0.206499934	0.584999979	0.539499968	0.617499977	0.662999988
12	L2 Clipping Aware Repeated Additive Gaussian Noise Attack	0.20599997	0.564499974	0.559999973	0.604999989	0.661499977
13	L2 Clipping Aware Repeated Additive Uniform Noise Attack	0.193499982	0.588999987	0.573999971	0.603499979	0.666999996
14	Attack	0.218999982	0.564499974	0.576499969	0.59799999	0.673999995
15	Newton Fool Attack	0.201499939	**0.479499996**	**0.380499959**	0.615999997	**0.449499965**
16	Linf Deep Fool Attack	0.213499963	0.539999992	0.552999973	0.594999969	0.661499977
17	Salt And Pepper Noise Attack	0.19599998	0.546999991	0.564999968	0.593499988	0.659999996
18	L2 Deep Fool Attack	0.20599997	0.548499972	0.57249999	0.586499989	0.66049999
19	L2 Additive Gaussian Noise Attack	0.237999976	0.536999971	0.548999965	0.613999993	0.676999986
20	L2 Additive Gaussian Noise Attack	0.23149997	0.532999992	0.563499987	0.602499992	0.662999988
21	L2 Clipping Aware Additive Gaussian	0.219499946	0.532999992	0.569499969	0.57949999	0.675499976
22	Attack	0.222499967	0.539499968	0.561499983	0.572999984	0.672499985
23	Linf Additive Uniform Noise Attack	0.224499941	0.523499966	0.570499986	0.59799999	0.659499973
24	L2 Carlini Wagner Attack	0.229499936	0.532499969	0.57249999	0.597499967	0.656499982
25	FGM	0.229999959	0.544999987	0.566999972	0.599499971	0.672999978
26	FGSM	0.220999956	0.54549998	0.533499986	0.584499985	0.663999975
27	L2 PGD	0.227999985	0.522999972	0.571999967	0.573999971	0.657999992
28	Linf PGD	0.229499936	0.548999965	0.530499965	0.596999973	0.667499989
29	PGD	0.209999979	0.565999985	0.564499974	0.600499988	0.664999992

- Epsilon=1.0

Attack No.	Attack Name	AlexNet	Resnet50	Resnet101	MobileNet_v2	DensetNet
	Original Accuracy	0.747058824	0.885373609	0.892686804	0.894117647	0.914149444
	Best Attack Accuracy	0.718600959	0.848807633	0.841494441	0.885055646	0.505723357
	Percentage of Performance Drop	3.809320504	4.130005188	5.734638752	1.013513243	**44.67826232**
1	**L2 Contrast Reduction Attack**	0.744038165	0.887599364	0.896820351	0.89125596	0.91462639
2	**Virtual Adversarial Attack**	0.746263921	0.887758344	0.894117646	0.891732909	0.915103339
3	**DDNA Attack**	0.750238478	0.886645466	0.895389505	0.892368838	0.915262319
4	**L2 Projected Gradient Descent Attack**	0.748012722	0.886327505	0.892686807	0.891096979	0.915739268
5	**Linf Projected Gradient Descent Attack**	0.746740848	0.883942768	0.892209858	0.889984101	0.916375197
6	**L2 Basic Iterative Attack**	0.747694761	0.886804454	0.895707473	0.892368838	0.917329095
7	**Linf Basic Iterative Attack**	0.74244833	0.8836248	0.894753575	0.892209858	0.915580288
8	**L2 Fast Gradient Attack**	0.743720204	0.884737678	0.895707473	0.892686807	0.914467409
9	**Linf Fast Gradient Attack**	0.740381569	0.883465819	0.892050877	0.887122415	0.912718602
10	**Attack**	0.743402213	0.886804454	0.896184422	0.893640697	0.914308429
11	**Attack**	0.746581882	0.884101748	0.895548493	0.893004768	0.91478537
12	**L2 Clipping Aware Repeated Additive Gaussian Noise Attack**	0.745469004	0.884737678	0.894594595	0.893958665	0.913036563
13	**L2 Clipping Aware Repeated Additive Uniform Noise Attack**	0.745310009	0.885532595	0.893958665	0.891891889	0.91399046
14	**Attack**	0.746422887	0.888553262	0.89793323	0.889348172	0.91399046
15	**Newton Fool Attack**	**0.718600959**	**0.848807633**	**0.841494441**	**0.89062003**	**0.505723357**
16	**Linf Deep Fool Attack**	0.741176486	0.884737678	0.889507152	0.888871223	0.915103339
17	**Salt And Pepper Noise Attack**	0.747853726	0.887758344	0.884737492	0.888871223	0.915262319
18	**L2 Deep Fool Attack**	0.747853726	0.883783787	0.887421242	0.890937999	0.916693166
19	**L2 Additive Gaussian Noise Attack**	0.743402213	0.884578697	0.894117646	0.892209858	0.912400633
20	**L2 Additive Gaussian Noise Attack**	0.750397459	0.888394274	0.893163756	0.89062003	0.916057236
21	**L2 Clipping Aware Additive Gaussian Noise Attack**	0.74562797	0.888394274	0.896025434	0.891096979	0.917170115
22	**Attack**	0.74594596	0.884101748	0.897774242	0.892368838	0.916057236
23	**Linf Additive Uniform Noise Attack**	0.744992048	0.885532595	0.892368838	0.890779011	0.915421307
24	**L2 Carlini Wagner Attack**	0.747694761	0.884101748	0.895230524	0.894117646	0.915898249
25	**FGM**	0.746581882	0.887122415	0.894594595	0.89062003	0.913354531
26	**FGSM**	0.740540534	0.880286172	0.88966614	0.885055646	0.913513511
27	**L2 PGD**	0.747535765	0.887281403	0.892209858	0.891732909	0.914308429
28	**Linf PGD**	0.747853726	0.884737678	0.893958665	0.891096979	0.914467409
29	**PGD**	0.744833082	0.887440383	0.892845787	0.893640697	0.916216217

Transferable Sparse Adversarial Attack

- Epsilon=0.0005

Source(Generator)	Model	L0-Norm	Time (ms)	Adversarial Accuracy (%)	Fooling Rate (%)	Robust Accuracy (%)
AlexNet	**AlexNet***	0	0.534025853	17.774	75.262	24.73767886
	ResNet50	0	0.759529916	8.362	90.7	**9.300476948**
	ResNet101	0	1.037881371	10.874	90.509	9.491255962
	MobileNet_v2	0	1.476634477	12.258	66.121	33.87917329
	DenseNet121	0	1.144518761	5.231	73.466	26.53418124
ResNet50	**AlexNet**	0	1.502175172	17.488	75.66	24.34022258
	ResNet50*	0	1.260088889	8.458	90.604	**9.395866455**
	ResNet101	0	1.066947552	10.874	90.143	9.856915739
	MobileNet_v2	0	1.487173147	12.385	65.421	34.57869634
	DenseNet121	0	1.182302762	5.262	73.339	26.66136725
ResNet101	**AlexNet**	0	0.568486966	17.758	75.517	24.48330684
	ResNet50	0	1.283827344	8.426	90.7	**9.300476948**
	ResNet101*	0	1.169546458	10.859	90.477	9.523052464
	MobileNet_v2	0	1.401777715	12.544	65.803	34.19713831
	DenseNet121	0	1.221956092	5.39	73.577	26.42289348
MobileNet_V2	**AlexNet**	0	1.503463959	17.742	75.342	24.6581876
	ResNet50	0	1.246488947	8.347	90.541	**9.459459459**
	ResNet101	0	1.051601545	10.97	90.334	9.666136725
	MobileNet_V2*	0	1.446312221	12.496	65.135	34.86486486
	DenseNet121	0	1.244934792	5.31	73.752	26.24801272
DenseNet121	**AlexNet**	0	1.461738092	:17.361	76.169	23.83147854
	ResNet50	0	1.221987856	8.394	90.477	9.523052464
	ResNet101	0	1.088902909	10.938	90.604	**9.395866455**
	MobileNet_v2	0	1.514493832	12.544	65.215	34.78537361
	DenseNet121*	0	1.389218761	5.482	71.294	28.706

- Epsilon=1.0

Source(Generator)	Model	L0-Norm	Time (ms)	Adversarial Accuracy (%)	Fooling Rate (%)	Robust Accuracy (%)
AlexNet	AlexNet*	50063.5586	0.421248555	8.15	59.55	40.45
	ResNet50	50055.8125	0.399045587	5.5	95.85	4.15
	ResNet101	50052.793	0.400970578	5.65	72.7	27.3
	MobileNet_v2	50048.793	0.383505344	10	68	32
	DenseNet121	50043.8477	0.384764671	3.1	76.2	23.8
ResNet50	AlexNet	0	0.424089432	8.45	58.85	41.15
	ResNet50*	0	1.145537853	5	96.65	3.55
	ResNet101	0	0.383425951	5.45	75.05	24.95
	MobileNet_v2	0	0.367376566	10.3	67.2	32.8
	DenseNet121	0	0.378295541	3.75	76.7	23.3
ResNet101	AlexNet	1147.067	0.422928572	7.3	60.05	39.95
	ResNet50	1176.4121	0.381159544	6	96.7	3.3
	ResNet101*	1165.7941	0.383220553	5.3	85.35	14.65
	MobileNet_v2	1131.2896	0.373892903	9.65	70.15	29.85
	DenseNet121	1155.4196	0.380117893	3.1	77.6	22.4
MobileNet_V2	AlexNet	0	0.434672475	8.3	58.35	41.65
	ResNet50	0	0.392568231	5.75	95.45	4.55
	ResNet101	0	0.406144023	4.8	75.35	24.65
	MobileNet_V2*	0	0.387378454	11.15	67.05	32.95
	DenseNet121	0	0.391313195	3.6	77.85	22.15
DenseNet121	AlexNet	2604.8662	0.421614766	9.5	59	41
	ResNet50	2622.8987	0.371679664	6.15	94.65	5.35
	ResNet101	2583.0205	0.374443054	5.2	70.35	29.65
	MobileNet_v2	2582.7012	0.367733002	10.15	72.2	27.8
	DenseNet121*	2649.0557	0.378239989	3.8	96.45	3.55

❖ UC Merced Land Use Dataset

❑ White-Box Attacks

- Epsilon=0.0005

Attack No.	Attack Name	AlexNet	Resnet50	Resnet101	MobileNet_v2	DensetNet
	Original Accuracy	0.653937947	0.723150358	0.754176611	0.813842482	0.801909308
	Best Attack Accuracy	0.627684951	0.706443906	0.732696891	0.799522668	0.785202861
	Percentage of Performance Drop	**4.01460059**	2.310232169	2.84810214	1.759531398	2.083333731
1	**L2 Contrast Reduction Attack**	0.642004758	0.708830535	0.766109779	0.809069201	0.811455846
2	**Virtual Adversarial Attack**	0.670644373	0.708830535	0.751789972	0.818615749	0.794749394
3	**DDNA Attack**	0.634844869	0.715990454	0.747016698	0.809069201	0.799522668
4	**L2 Projected Gradient Descent Attack**	**0.627684951**	0.720763713	0.756563231	0.825775653	0.801909298
5	**Linf Projected Gradient Descent Attack**	0.649164677	0.715990454	0.763723135	0.818615749	0.806682572
6	**L2 Basic Iterative Attack**	0.644391388	0.725536972	0.761336505	0.825775653	0.801909298
7	**Linf Basic Iterative Attack**	0.646778017	0.708830535	0.773269683	0.813842475	0.806682572
8	**L2 Fast Gradient Attack**	0.651551306	0.713603795	0.756563231	0.830548912	0.801909298
9	**Linf Fast Gradient Attack**	0.656324565	0.708830535	0.766109779	0.811455846	0.797136024
10	**Attack**	0.651551306	0.723150343	0.758949876	**0.799522668**	0.799522668
11	**L2 Repeated Additive Uniform Noise Attack**	0.649164677	0.706443906	0.761336505	0.806682572	0.799522668
12	**L2 Clipping Aware Repeated Additive Gaussian Noise Attack**	0.651551306	0.725536972	0.754176602	0.823389009	0.809069201
13	**L2 Clipping Aware Repeated Additive Uniform Noise Attack**	0.644391388	0.718377084	0.758949876	0.804295942	0.801909298
14	**Linf Repeated Additive Uniform Noise Attack**	0.646778017	0.723150343	0.756563231	0.818615749	0.797136024
15	**Newton Fool Attack**	0.649164677	0.711217165	0.732696891	0.816229105	0.794749394
16	**Linf Deep Fool Attack**	0.663484484	0.730310261	0.766109779	0.809069201	0.801909298
17	**Salt And Pepper Noise Attack**	0.646778017	0.715990454	0.763723135	0.818615749	0.797136024
18	**L2 Deep Fool Attack**	0.649164677	0.723150343	0.768496409	0.816229105	0.801909298
19	**L2 Additive Gaussian Noise Attack**	0.639618129	0.727923632	0.780429587	0.801909298	0.804295942
20	**L2 Additive Gaussian Noise Attack**	0.653937936	0.708830535	0.761336505	0.809069201	0.794749394
21	**L2 Clipping Aware Additive Gaussian**	0.634844869	0.727923632	0.758949876	0.806682572	0.804295942
22	**L2 Clipping Aware Additive Uniform Noise**	0.639618129	0.720763713	0.763723135	0.804295942	0.78758949
23	**Linf Additive Uniform Noise Attack**	0.658711195	**0.706443906**	0.751789972	0.811455846	0.801909298
24	**L2 Carlini Wagner Attack**	0.651551306	0.715990454	0.766109779	0.821002379	0.801909298
25	**FGM**	0.658711195	0.718377084	0.758949876	0.813842475	0.804295942
26	**FGSM**	0.63245821	0.713603795	0.773269683	0.813842475	0.797136024
27	**L2 PGD**	0.63245821	0.730310261	0.768496409	0.816229105	0.801909298
28	**Linf PGD**	0.637231499	0.708830535	0.758949876	0.809069201	**0.785202861**
29	**PGD**	0.642004758	0.725536972	**0.732696891**	0.816229105	0.806682572

- Epsilon=1.0

Attack No.	Attack Name	AlexNet	Resnet50	Resnet101	MobileNet_v2	DensetNet
	Original Accuracy	0.653937947	0.723150358	0.754176611	0.813842482	0.801909308
	Best Attack Accuracy	0.119331717	0.095465362	0.095465362	0.014319777	0.028639555
	Percentage of Performance Drop	81.7518286	86.79868425	87.34177636	**98.24047309**	96.4285793
1	L2 Contrast Reduction Attack	0.656324565	0.723150343	0.766109779	0.556085914	0.968973747
2	Virtual Adversarial Attack	0.649164677	0.720763713	0.756563231	0.584725529	0.973747015
3	DDNA Attack	0.625298321	0.696897358	0.751789972	0.556085914	0.973747015
4	L2 Projected Gradient Descent Attack	0.651551306	0.725536972	0.766109779	0.572792351	0.973747015
5	Linf Projected Gradient Descent Attack	**0.119331717**	**0.095465362**	0.097851992	0.019093037	0.040572762
6	L2 Basic Iterative Attack	0.637231499	0.715990454	0.756563231	0.558472544	0.973747015
7	Linf Basic Iterative Attack	0.121718347	0.102625251	0.102625251	0.016706407	**0.028639555**
8	L2 Fast Gradient Attack	0.646778017	0.713603795	0.758949876	0.572792351	0.971360382
9	Linf Fast Gradient Attack	0.138424814	0.136038125	0.138424814	0.100238621	0.045346022
10	Attack	0.651551306	0.708830535	0.768496409	0.563245803	0.973747015
11	Attack	0.653937936	0.711217165	0.751789972	0.565632433	0.968973747
12	L2 Clipping Aware Repeated Additive Gaussian Noise Attack	0.642004758	0.720763713	0.751789972	0.560859174	0.971360382
13	L2 Clipping Aware Repeated Additive Uniform Noise Attack	0.658711195	0.715990454	0.766109779	0.568019062	0.971360382
14	Attack	0.36992836	0.288782775	0.403341293	0.124104977	0.97613365
15	Newton Fool Attack	0.644391388	0.723150343	0.727923632	0.520286381	0.26491642
16	Linf Deep Fool Attack	0.403341293	0.300715983	0.408114552	0.164677799	0.957040571
17	Salt And Pepper Noise Attack	0.651551306	0.720763713	0.756563231	0.565632433	0.245823383
18	L2 Deep Fool Attack	0.646778017	0.723150343	0.763723135	0.558472544	0.966587111
19	L2 Additive Gaussian Noise Attack	0.649164677	0.718377084	0.756563231	0.563245803	0.971360382
20	L2 Additive Gaussian Noise Attack	0.656324565	0.715990454	0.761336505	0.556085914	0.971360382
21	Noise Attack	0.649164677	0.725536972	0.744630069	0.572792351	0.968973747
22	Attack	0.642004758	0.713603795	0.766109779	0.553699255	0.973747015
23	Linf Additive Uniform Noise Attack	0.415274441	0.331742227	0.453460574	0.152744591	0.284009516
24	L2 Carlini Wagner Attack	0.651551306	0.720763713	0.761336505	0.551312625	0.973747015
25	FGM	0.644391388	0.715990454	0.768496409	0.57756561	0.968973747
26	FGSM	0.143198073	0.140811443	0.133651495	0.10501188	0.047732651
27	L2 PGD	0.653937936	0.718377084	0.770883054	0.572792351	0.973747015
28	Linf PGD	0.136038125	0.097851992	**0.095465362**	0.026252925	0.038186133
29	PGD	0.124104977	0.102625251	0.100238621	**0.014319777**	0.033412874

Transferable Sparse Adversarial Attack

- Epsilon=0.0005

Source(Generator)	Model	L0-Norm	Time (ms)	Adversarial Accuracy (%)	Fooling Rate (%)	Robust Accuracy (%)
AlexNet	AlexNet*	9626.8281	0.693851553	36.277	48.926	51.07398568
	ResNet50	9661.6709	0.862086872	19.093	61.098	38.90214797
	ResNet101	9586.6016	1.251717456	23.866	78.998	**21.00238663**
	MobileNet_v2	9645.3301	1.358397536	11.456	16.706	83.29355609
	DenseNet121	9592.4707	1.187203892	31.981	69.69	30.31026253
ResNet50	AlexNet	2228.3818	1.555173095	36.516	47.494	52.50596659
	ResNet50*	2222.7686	1.353261579	19.332	58.95	41.05011933
	ResNet101	2241.4011	1.165145906	22.912	77.088	**22.91169451**
	MobileNet_v2	2234.9023	1.548439904	11.933	17.184	82.81622912
	DenseNet121	2228.0718	1.223687056	31.265	69.212	30.7875895
ResNet101	AlexNet	0	1.687926154	35.8	47.971	52.02863962
	ResNet50	0	1.32609381	19.809	58.473	41.5274463
	ResNet101*	0	1.067505247	23.628	78.043	**21.95704057**
	MobileNet_v2	0	1.533233465	11.933	14.797	85.20286396
	DenseNet121	0	1.185852474	31.265	69.928	30.07159905
MobileNet_V2	AlexNet	45019.4922	32.73339431	41.527	50.119	49.88066826
	ResNet50	45008.9688	32.4927025	17.9	63.246	36.75417661
	ResNet101	45020.0703	32.33842008	21.48	84.726	**15.27446301**
	MobileNet_V2*	45011.9414	32.47196635	11.695	25.776	74.22434368
	DenseNet121	45013.6094	32.65738886	29.594	67.542	32.45823389
DenseNet121	AlexNet	0	1.631933067	34.845	47.494	52.50596659
	ResNet50	0	1.28238241	19.093	61.098	38.90214797
	ResNet101	0	1.044955629	23.866	78.52	**21.4797136**
	MobileNet_v2	0	1.469535304	10.97	17.422	82.57756563
	DenseNet121*	0	1.244934792	5.31	69.215	30.78591291

- Epsilon=1.0

Source(Generator)	Model	L0-Norm	Time (ms)	Adversarial Accuracy (%)	Fooling Rate (%)	Robust Accuracy (%)
AlexNet	AlexNet*	9662.1025	0.285544088	15.752	67.542	32.45823389
	ResNet50	9664.9951	0.27224329	12.411	56.563	43.43675418
	ResNet101	9618.9404	0.389571406	19.809	75.179	**24.82100239**
	MobileNet_v2	9673.0264	0.372455911	33.174	52.506	47.49403341
	DenseNet121	9670.1074	0.375579251	14.32	46.778	53.22195704
ResNet50	AlexNet	2219.1479	0.245995055	35.561	53.938	46.06205251
	ResNet50*	2228.4966	1.061151022	18.616	93.556	**6.443914081**
	ResNet101	2235.4153	0.461475377	29.117	83.771	16.22911695
	MobileNet_v2	2226.2747	0.390163754	48.21	48.926	51.07398568
	DenseNet121	2238.2029	0.365764827	18.377	78.282	21.71837709
ResNet101	AlexNet	0	0.232030782	35.322	51.313	48.68735084
	ResNet50	0	0.262025433	19.809	61.098	38.90214797
	ResNet101*	0	2.595364905	22.912	79.47	**20.52505967**
	MobileNet_v2	0	0.244205493	45.823	46.778	53.22195704
	DenseNet121	0	0.502975573	24.344	65.155	34.84486874
MobileNet_V2	AlexNet	0	1.132587829	35.561	46.778	53.22195704
	ResNet50	0	1.151256174	20.048	59.666	40.33412888
	ResNet101	0	1.171488182	24.105	78.998	**21.00238663**
	MobileNet_V2*	0	1.260259555	46.53	46.301	53.69928401
	DenseNet121	0	1.124749605	23.866	64.439	35.56085919
DenseNet121	AlexNet	0	0.4171244	35.561	48.926	51.07398568
	ResNet50	0	0.404797192	19.332	58.95	41.05011933
	ResNet101	0	0.422845308	24.105	80.191	**19.80906921**
	MobileNet_v2	0	0.412999019	45.585	42.959	57.04057279
	DenseNet121*	0	0.425844033	24.582	65.394	34.60620525

Observations

- On NWPU-RESIC45, the highest drop in accuracy was observed when Virtual Adversarial Attack **(75.23%)** and Newton Fool Attack **(44.67%)** were deployed under eps= 0.0005 and eps=1.0 respectively. to create adversarial examples on the AlexNet and DenseNet respectively.
- When TSAA was executed on NWPU dataset, highest fooling rate was **9.30%** when the generator was trained on ResNet101 architecture and the adversarial examples were created using the ResNet50 model under eps=0.0005 and **3.3%** when the generator was trained on ResNet101 architecture and the adversarial examples were created using the ResNet50 model under eps=1.0
- On UC Merced Dataset, the highest drop in accuracy was observed when L_2 Projected Gradient Descent Attack **(4.01%)** and PGD Attack **(98.24%)** were deployed under eps= 0.0005 and eps=1.0 respectively. to create adversarial examples on the AlexNet and $MobileNetV_2$ respectively.
- When TSAA was executed on UC Merced dataset, highest fooling rate was **15.27%** when the generator was trained on $MobileNetV_2$ architecture and the adversarial examples were created using the ResNet101 model under eps=0.0005 and **6.44%** when the generator was trained on ResNet50 architecture and the adversarial examples were created using the ResNet50 model under eps=1.0

References

- https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9146660
- https://ieeexplore.ieee.org/document/9339955
- https://www.hindawi.com/journals/scn/2021/6663028/
- https://ieeexplore.ieee.org/document/9119167
- https://arxiv.org/pdf/1805.10997
- **NWPU-RESIC45**: https://www.tensorflow.org/datasets/catalog/resisc45*(dataset)*; https://ieeexplore.ieee.org/document/7891544*(research paper)*
- **UC Merced Land-Use Dataset**: https://www.tensorflow.org/datasets/catalog/uc_merced *(dataset)*;https://faculty.ucmerced.edu/snewsam/papers/Zhu_SIGSPATIAL15_LandUse Classification.pdf*(research paper)*
- **FoolBox**: https://github.com/bethgelab/foolbox*(GitHub)*; https://foolbox.jonasrauber.de*(official guide)*; https://arxiv.org/abs/1707.04131*(research paper)*
- **Transferable Sparse Adversarial Attack(*TSAA*)**: https://arxiv.org/abs/2105.14727*(research paper)*; https://github.com/shaguopohuaizhe/TSAA*(GitHub)*

Thank You!

Questions?