Machine Learning Security Project

https://github.com/APruner-23/ML_Sec_Project.git

Progetto_MLSec_Leandri_Pruner.ipynb

Milena Leandri (70/90/00519) - Alessandro Pruner (70/90/00502)

01

Introduction

Objective of the Project

Objective of our project

Adversarial **Generation**

Generation of adversarial examples on a 3 **RobustBench** model **ensemble**

Transferability **Evaluation**

Evaluate Transferability on **7 different** RobustBench models

Adversarial Crafting *characteristics*

Universal

Each adversarial example is crafted against the combined ensemble of three models

Untargeted

We do not specify any **target class**
We only want the models to **misclassify** the input sample

Model **Ensemble**

- Our **ensemble**:

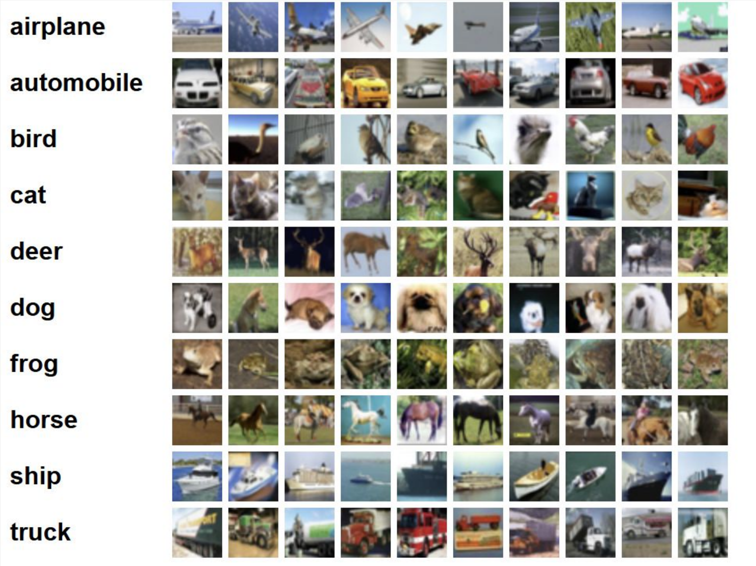
	Clean Accuracy	Robust Accuracy	Architecture
Zhang2019You	87.20%	44.83%	WideResNet-34-10
Xu2023Exploring_WRN-28-10	**93.69%**	**63.89%**	WideResNet-28-10
Gowal2021Improving_28_10_ddpm_100m	87.50%	63.38%	WideResNet-28-10

Transferability Models

	Clean Accuracy	Robust Accuracy	Architecture
Huang2022Revisiting_WRN-A4	91.58%	65.79%	WideResNet-A4
Peng2023Robust	**93.27%**	**71.07%**	RaWideResNet-70-16
Amini2024MeanSparse_Ra_WRN_70_16	93.24%	68.94%	MeanSparse RaWideResNet-70-16
Sehwag2021Proxy_ResNest152	87.30%	62.79%	ResNet152
Debenedetti2022Light_XCiT-L12	91.73%	57.58%	XCiT-L12
Cui2023Decoupled_WRN-28-10	92.16%	67.73%	WideResNet-28-10
Rebuffi2021Fixing_28_10_cutmix_ddpm	87.33%	60.73%	WideResNet-28-10

CIFAR-10 **Dataset**

- **60000** 32x32 *colour* images
- **10** Classes
- **6000** images per class
- All *models* used have been **trained** on this dataset



02

FGSM Attack

Fast Gradient Sign Method (FGSM)

- Introduced by Goodfellow et al. in *Explaining and Harnessing Adversarial Examples* (2015)
- Adversarial attack intended to reveal weaknesses in machine learning models
- Perturbs the input in the direction of the gradient of the loss with respect to the input

original input

model's parameter

$$adv\_x = x + \epsilon * \text{sign}(\nabla_x J(\theta, x, y))$$

adversarial example

original label

multiplier to ensure perturbation is small

FGSM Ensamble

1. Prepare the adversarial input
2. Ensamble prediction
3. Loss and gradient calculation
4. FGSM perturbation
5. Clipping
6. Output

```
def fgsm_ensemble(models, x, true_labels, epsilon):
    x_adv = x.clone().detach()
    x_adv.requires_grad = True
    logits_list = [model(x_adv) for model in models]
    stacked_logits = torch.stack(logits_list, dim=0)
    ensemble_logits = torch.mean(stacked_logits, dim=0)
    loss = margin_loss(ensemble_logits, true_labels)
    grad = torch.autograd.grad(loss, x_adv)[0]
    x_adv = x_adv.detach() + epsilon
* torch.sign(grad.detach())
    x_adv = torch.clamp(x_adv, 0, 1).detach()
    y_pred = predict_with_ensemble(models, x_adv)
    y_pred_0 = y_pred[0]
    return x_adv, y_pred_0
```

Attacking an ensemble of models makes the adversarial perturbation more effective and robust, as it considers the weaknesses of multiple networks at once.

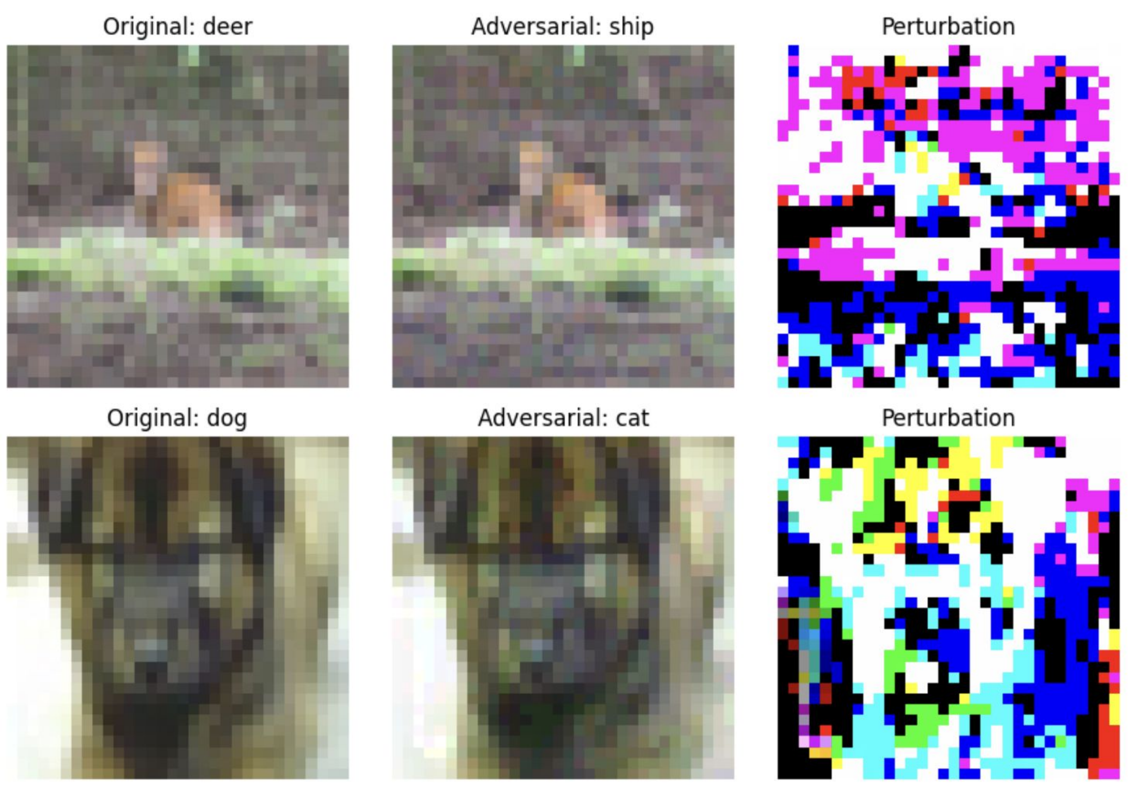
Attack

```
for (x, y_true) in data_loader:
   x, y_true = x.to(device), y_true.to(device)
   y_pred = predict_with_ensemble(models, x)
   y_pred_0 = y_pred[0]
   if y_pred_0 != y_true:
       excluded_adv_examples += 1
       continue
   x_adv, y_pred_adv = fgsm_ensemble(models, x, y_true,
8/255)
   if y_pred_adv != y_true:
       final_adversarials.append((x_adv, y_true.item()))
       total_adv_examples += 1
```

For each sample:

- **Predicts** the label using the *ensemble* of models
- If the ensemble already misclassifies the original input, that **sample is excluded** from the attack process
- Otherwise, *FGSM* adversarial attack is applied using an *epsilon* of *8/255* to **generate an adversarial example**

Generated Perturbation



Out of 1000 samples ...

149

adversarial examples created

75

excluded examples

03

Transferability Evaluation

What’s **Transferability**?

Transferability is useful to understand if an attack created to fool a model, also fools *other* models.
It is the ability of an attack developed against a surrogate model to succeed also against a different target model

Model Ensemble →

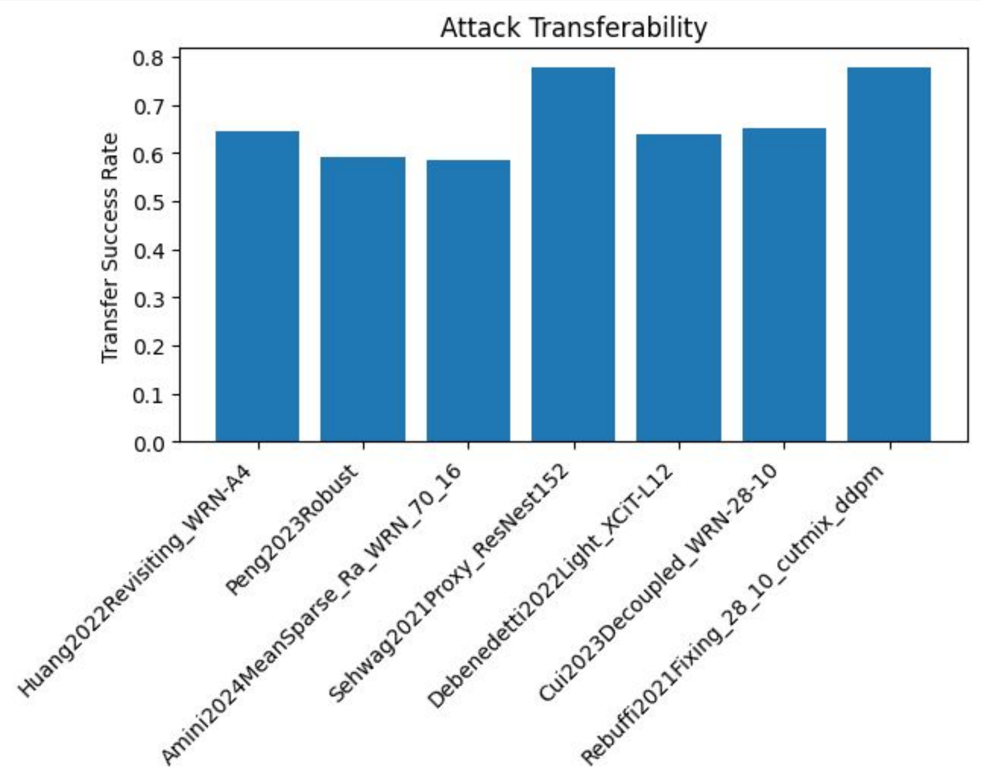
Adversarials

- Model 1 5/100 fool the model → Bad Transferability
- Model 2 70/100 fool the model → Good Transferability
- Model 3 40/100 fool the model → Medium Transferability
- Model 4 12/100 fool the model → Bad Transferability

Results

	Number of Adversarials that fooled the model	Attack Success Rate
Huang2022Revisiting_WRN-A4	96	64.4%
Peng2023Robust	88	59.1%
Amini2024MeanSparse_Ra_WRN_70_16	87	58.4%
Sehwag2021Proxy_ResNest152	116	77.9%
Debenedetti2022Light_XCiT-L12	95	63.8%
Cui2023Decoupled_WRN-28-10	97	65.1%
Rebuffi2021Fixing_28_10_cutmix_ddpm	116	77.9%

Attack Success Rate Plot



Thanks for your attention!

Milena Leandri (70/90/00519)
Alessandro Pruner (70/90/00502)