Study and evaluation of the TranFuzz method for generating and using black-box adversarial samples

Mars 2025

Thimoté Dupuch
thimote.dupuch@etu.emse.fr

Noé Backert
noe.backert@etu.emse.fr

Timothée Cloup-Martin
timothee.cloup@etu.emse.fr

Mounia Kharbouche-Harrari
mounia.kharbouche-harrari@st.com

Julien Montmasson
julien.montmasson@st.com

Abstract The use of artificial neural networks presents numerous security challenges and issues, such as robustness against attacks through the generation of adversarial examples. For a classifier-type model, white-box attacks exploit knowledge of the internal architecture of a deep neural network and the data on which it was trained to generate images likely to deceive the classification system. This type of attack, although easy to implement and very powerful, is not realistic for a computer system because it is often impossible to know the model's architecture or the initial training data. Black-box attacks aim to design adversarial examples without assuming any particular knowledge of the model beforehand. TranFuzz is a system that enables the creation of a model capable of generating adversarial examples while adhering to the conditions of a black-box attack. In this work, we provide a review and study of the performance of the TranFuzz system.

Contents

I] Introduction

In recent years, neural networks have become integral to various applications, from image recognition and natural language processing to autonomous vehicles and healthcare diagnostics. However, their widespread adoption has also exposed them to potential security threats known as adversarial attacks. These attacks involve subtle manipulations of input data designed to deceive the neural network into making incorrect predictions or decisions. For instance, slight alterations to an image, imperceptible to the human eye, can cause a neural network to misclassify objects, leading to significant errors in critical systems. The importance of addressing these vulnerabilities can't be overstated. In domains such as autonomous driving, adversarial attacks could lead to catastrophic failures, endangering human lives. Similarly, in healthcare, misdiagnoses due to adversarial inputs could have severe consequences for patient treatment. Furthermore, as neural networks are increasingly deployed in sensitive areas like finance and cybersecurity, their susceptibility to attacks poses substantial risks to data integrity and privacy. Therefore, understanding and mitigating adversarial attacks is crucial for ensuring the reliability and safety of neural network-based systems in real-world applications.

The entire body of work on the TranFuzz system is presented in the paper [1]. It provides a framework to train, test and use a new type of adversarial attack, that can also be used to improve the robustness of base models via adversarial retraining.

II] State of the Art

II.1] White-Box Attacks

White-box attacks assume that the attacker has full knowledge of the target model, including its architecture, parameters, and training data. This level of access allows for highly effective and precise attacks. Some of the most notable white-box attack methods include:

Fast Gradient Sign Method (FGSM): Introduced by Goodfellow et al. (2014) [2], FGSM generates adversarial examples by disturbing the input in the direction of the gradient of the loss function with respect to the input. This method is computationally efficient and can be implemented with a single gradient computation.

Basic Iterative Method (BIM): An extension of FGSM, BIM applies smaller perturbations iteratively, making it more effective in generating adversarial examples that are closer to the original input.

Jacobian-based Saliency Map Attack (JSMA): This method uses the Jacobian matrix to identify the most influential pixels in the input and perturb them to maximize the likelihood of misclassification.

DeepFool: Introduced by Moosavi-Dezfooli et al. (2016) [3], DeepFool is an iterative method that aims to find the minimal perturbation required to change the classification of an input. It is particularly effective in generating adversarial examples that are visually indistinguishable from the original input.

Projected Gradient Descent (PGD): Introduced in 2019 [4], this attack generates adversarial examples by iteratively refining perturbations to maximize a model's classification error.

Carlini & Wagner (C&W) Attack: This method, proposed by Carlini and Wagner (2017) [5], formulates the generation of adversarial examples as an optimization problem. It is highly effective in generating adversarial examples with minimal perturbations and has been shown to be effective against various robust models.

II.2] Black-Box Attacks

Black-box attacks, on the other hand, assume that the attacker has limited or no knowledge of the target model's internal workings. These attacks are more challenging to execute but are also more realistic in many practical scenarios. Some prominent black-box attack methods include:

Spatial Tranformation (ST): Introduced in 2018 [6], this attacks uses a different type of perturbation : the spatial transformation, that is able to generate perceptually realistic and more difficult to defend against images.

Zeroth Order Optimization (ZOO): Proposed by Chen et al. (2017) [7], ZOO uses zeroth-order optimization techniques to estimate the gradients of the target model. It can generate adversarial examples with limited queries to the model.

One-Pixel Attack: This method, introduced by Su et al. (2019) [8], demonstrates that changing a single pixel in an image can be sufficient to fool a classifier. It uses differential evolution to find the optimal pixel to perturb.

Boundary Attack: Proposed by Brendel et al. (2018) [9], the boundary attack starts with a large perturbation and iteratively reduces it while maintaining the adversarial nature of the example. It does not require gradient information and is effective against various models.

Transfer-Based Attacks: These attacks leverage the transferability of adversarial examples across different models. An adversarial example generated for one model can often fool another model with a similar architecture or training data. This property allows attackers to generate adversarial examples using a surrogate model and apply them to the target model.

Query-Efficient Black-Box Attacks: Methods like the Simple Black-Box Attack (SimBA) by Guo et al. (2019) [10] aim to minimize the number of queries to the target model. SimBA uses random perturbations and queries the model to iteratively refine the adversarial example.

II.3] Defenses Against Adversarial Attacks

In response to these attacks, various defense mechanisms have been proposed. Some notable defenses include:

Adversarial Training: This method involves training the model on both clean and adversarial examples to improve its robustness. It has been shown to be effective against various attacks but can be computationally expensive.

Defensive Distillation: Defensive distillation involves training the model to produce soft labels and then using these labels to train a second model. This process can make the model more robust against adversarial examples.

Gradient Masking: This technique aims to hide the gradient information from the attacker, making it more difficult to generate adversarial examples. However, it has been shown to be vulnerable to attacks that doesn't use any gradient method such as Carlini & Wagner's attack.

Input Transformation: Methods like JPEG compression, random resizing, and padding can be used to transform the input and reduce the effectiveness of adversarial perturbations.

III] TranFuzz system

TranFuzz is an advanced framework designed to generate highly transferable adversarial examples. It operates through two key components:

- Domain Adaptation with Deep Subdomain Adaptation Network (DSAN)

TranFuzz leverages DSAN to align the source and target data distributions. This process enables the creation of a local substitute model that closely mimics the target model's behavior, even without direct access to the target data.

- Adversarial Example Generation via Fuzzing Techniques

By utilizing neuron coverage metrics, TranFuzz identifies model vulnerabilities and applies targeted mutations to maximize attack transferability. Additionally, it introduces a novel ensemble-based seed mutation strategy, enhancing the diversity and effectiveness of generated adversarial samples.

III.1] DSAN

The Deep Subdomain Adaptation Network (DSAN), as proposed by [11] Zhu et al. (2020), addresses domain adaptation challenges by reducing the distribution gap between source and target data domains. In TranFuzz, DSAN plays a crucial role in building a local substitute model that learns to replicate the target model's behavior.

The objective of the DSAN methodology is to train a local model capable of generalizing to the target model's decision boundaries. DSAN employs an adaptive loss function composed of two parts:

- **Classification loss** (cross-entropy), ensuring the model learns accurate decision boundaries on available data.

- **Local Maximum Mean Discrepancy (LMMD)** loss, which minimizes distributional discrepancies between source and target data across different feature levels.

This combination makes the local model more representative of the target model and significantly enhances the transferability of adversarial examples.

An effective adversarial example is one that triggers new neuron activations that regular inputs do not, pushing the model into less explored and more vulnerable states.

On the other hand, TranFuzz uses two key objective functions to guide adversarial generation:

- **Misclassification Objective** :

$$\text{obj}_{1\text{TF}} : f(x_{\text{adv}}) \neq \text{true label of } \text{x}_{\text{adv}}$$

The adversarial example is successful if the local substitute model misclassifies it.

- **Similarity Preservation Objective**

Here, Average Structural Similarity (ASS) ensures the adversarial input remains visually indistinguishable from the original input. A high similarity threshold (set to 0.96) ensures imperceptibility to human observers.

A generated adversarial example must satisfy both conditions — it must induce misclassification while remaining visually close to the original input.

In the code provided by the research document, the model used for the DSAN is a ResNet, which remains widely used today. The `DSAN.py` code calls the source dataset as well as the target model's test dataset to evaluate adaptation. The target model's training dataset is also used, but only its size is retrieved in order to train the ResNet with exactly the same number of images from the source dataset. Thus, it can be clearly observed that the model generalizes properly.

III.2] Fuzzer

The fuzzing module in TranFuzz is inspired by software fuzz testing, but applied to neural networks. It explores the unusual behavioral spaces of the local substitute model to produce adversarial examples that not only fool this model but are also likely to succeed against the target model.

Core Components of the Fuzzing Process:

- Neuron Coverage (NC): Guides the selection of inputs and mutations by focusing on maximizing neuron activation diversity, with DeepXplore being used to achieve this goal. In `CoverageUpdate.py`, each neuron is checked by taking the average of its activations. If its value exceeds the threshold, it is marked as "covered."

```
if torch.mean(scaled[..., num_neuron]) > threshold and not model_layer_dict[(list(layer_names.keys())
[idx], num_neuron)]:
                model_layer_dict[(list(layer_names.keys())[idx], num_neuron)] = True
```

- Coverage Analyzer: Dynamically identifies neurons whose activation patterns are under-explored and prioritizes them for mutation.

```
if not_covered:
        layer_name, index, v = random.choice(not_covered)
```

- Objective Functions: The two conditions outlined above (misclassification and similarity preservation) define the success criteria for each adversarial sample.

```
condition_max_sim = torch.mean(dist).item() <0.995 * torch.mean(ssim(init_image, init_image)).item()
condition_min_sim = torch.mean(dist).item() < 0.9 * torch.mean(ssim(init_image, init_image)).item()
```

- Mutator: Applies various input transformations and perturbations to discover adversarial variants with maximal transferability. In the `mutators.py` file, several mutations can be seen, such as gradient modification, Gaussian blur, and salt-and-pepper noise.

```
gauss_noise = torch.tensor([gaussian_gradient_magnitude(grads, sigma=2)])
blur = cv2.GaussianBlur(img, (3, 3), 0)
```

The notable innovation introduced by TranFuzz is the ensemble-based seed mutation strategy, which maximizes the difference between benign and adversarial examples by using a population-based approach rather than relying on single-instance mutations. This seems to significantly improve the diversity and success rate of adversarial attacks across different target models.

IV] Evaluation methodology

The methodology involves several key steps to evaluate the robustness of a target model against adversarial examples generated by a Deep Subdomain Adaptation Network (DSAN). The process is as follows:

1. **Dataset Selection**:
   - Choose one of the two datasets: Office31 or OfficeHome (the latter being larger).
2. **Target Model Training**:
   - Select a target model architecture from AlexNet, VGG19, or DenseNet121.
   - Train the target model on one of the subsets of the chosen dataset. The subsets include:
     - Office31/Amazon
     - Office31/RealWorld
     - OfficeHome/Product
     - OfficeHome/Webcam
   - Assume this subset is unknown for the attack.
3. **Source Model Training with DSAN**:
   - Train a ResNet model using the DSAN method on the same dataset but using the complementary subset. This subset is referred to as the source data, and the model trained on it is the source model.
4. **Local Inference and Training**:
   - Perform local inference on the target model using the source data.
   - Use the inference results to further train the ResNet model.
5. **Adversarial Example Generation**:
   - Once trained, the ResNet model generates adversarial examples on the test split of the source data.
6. **Evaluation**:
   - Test the misclassification rate of these adversarial examples on the target model to assess its vulnerability.

By following these steps, the methodology aims to evaluate how well the target model can withstand attacks from adversarial examples generated by a DSAN-trained ResNet model.

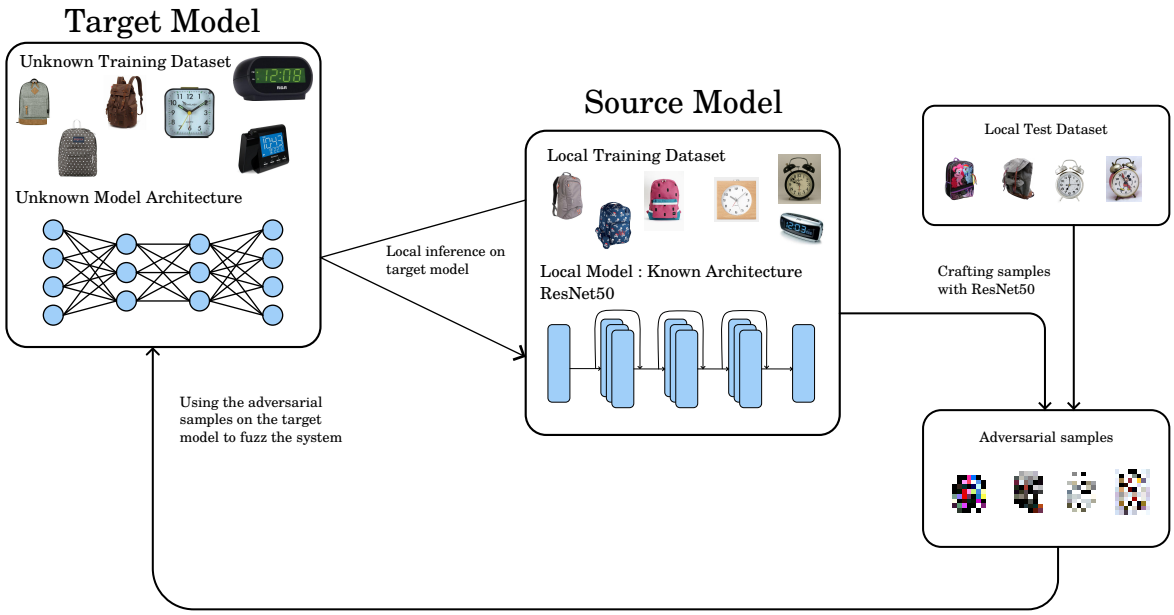

Figure 1: Overall fuzzing methodology diagram

IV.1] Adversarial retraining

To further enhance the robustness of the target model, the dataset can be augmented with the previously generated adversarial examples. This augmented dataset is then used to train from scratch the target model with the same number of epochs as the original non robust training. By incorporating adversarial examples into the training process, the target model can better learn to defend against such attacks, improving its overall resilience and accuracy in real-world scenarios.

By following these steps, the methodology aims to evaluate how well the target model can withstand attacks from adversarial examples generated by a DSAN (of ResNet architecture) model and to improve its robustness through retraining.

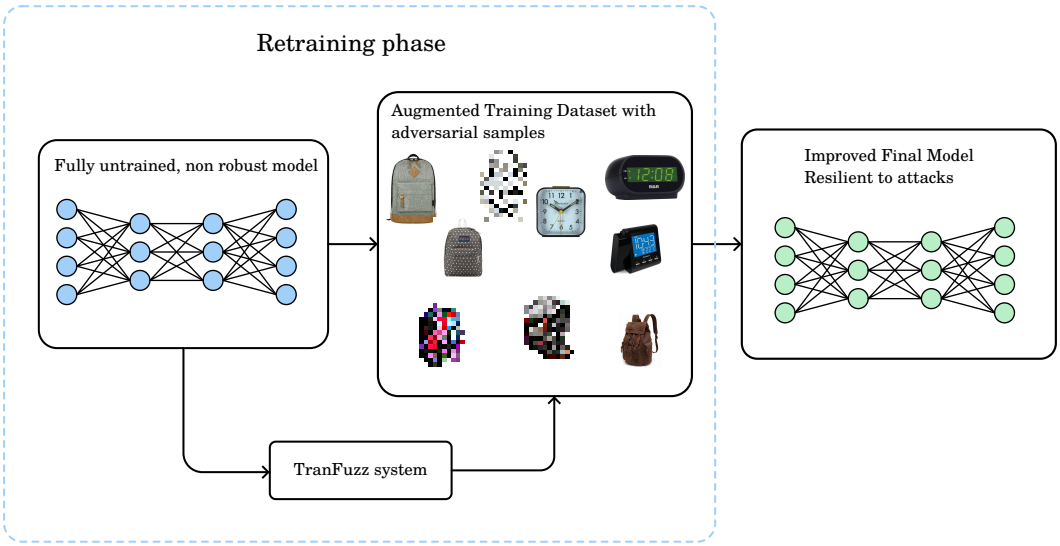

Figure 2: Adversarial retraining with augmented dataset

In order to perform other kind of attacks, and to setup defense strategies with the TranFuzz system, we used Adversarial Robustness Toolbox (ART) [12] that provides a convenient way to craft adversarial images. It also simplifies the implementation of defense strategies such as FBF [13] and Madry's Protocol [4]. However, we created our own implementation of the retraining using TranFuzz, with a handmade dataset containing both base images and adversarial images.

V] Experiences and protocols

1. **Baseline evaluation**

- Train a target model on clean data.
- Evaluate and record the accuracy of the target model.
- Train a source model (ResNet) using DSAN on the target model and a same labelled dataset
- Evaluate the target model against known adversarial attacks (FGSM, PGD, C&W, Spatial Transformations (ST))
- Generate fuzzed adversarial examples using the trained source model.
- Evaluate the adversarial examples generated from the source model on the target model.

2. **Madry robust model evaluation**

- Train a Madry robust target model (adversarial training based on the initial target model) using the ART (Adversarial Robustness Toolbox) framework [12].
- Evaluate and record the accuracy of the Madry robust model.
- Train a new source model using DSAN, this time using the Madry robust model as the target.
- Test the Madry robust model against the same known attacks (FGSM, PGD, C&W, ST).
- Generate fuzzed adversarial examples targeting the Madry robust model.
- Evaluate the Madry robust model against these newly generated fuzzed examples.

3. **TranFuzz robust model evaluation**

- Train a TranFuzz robust model by retraining the target model on fuzzed images generated in the previous stage.

- Evaluate and record the accuracy of the TranFuzz robust model.
- Train a new source model using DSAN on the TranFuzz robust target model.
- Test the TranFuzz robust model against the same known attacks (FGSM, PGD, C&W, ST).
- Generate fuzzed adversarial examples targeting the TranFuzz robust model.
- Evaluate the TranFuzz robust model on these fuzzed samples.

VI] Results

VI.1] Process

In order to determine whether the system works better than other conventional attacks and defense methods, we had to compare them step by step with TranFuzz.

To be able to follow our protocol, we developed an improvement[1] of the code given from the article. [1].

We first trained a target model, measured its accuracy on the test dataset : the percentage of correct label classification against all images on the test dataset.

We then performed FGSM, PGD, C&W and ST attacks on the test dataset of the target model and measured each accuracy value. We also performed the TranFuzz attack to compare results.

We trained a first robust model that is using the Madry's Protocol [4] to improve the target model's robustness and performed the attacks again.

Finally, we retrained from scratch (without any transfer learning) a model that is using a dataset consisting of 90% of the source training dataset images, and 10% of adversarial images previously generated with the TranFuzz system. We performed the attacks on this model as well.

This allows us to understand which attack works effectively on which model, and compare the results with each dataset.

[1] https://github.com/noebackert/PR-ICICS

VI.2] Transferability results

Target Model	**Target Dataset**	**Transfer Accuracy** (Accuracy on the source model)
Densenet (non robust)	Webcam	98.90% (90/91)
Densenet (Madry retrained model)	Webcam	98.90% (90/91)
Densenet (TranFuzz retrained)	Webcam	98.90% (90/91)
Densenet (non robust)	Amazon	76.79% (225/293)
Densenet (Madry retrained model)	Amazon	77.13% (226/293)
Densenet (TranFuzz retrained)	Amazon	79.52% (233/293)
Densenet (non robust)	Product	89.36% (420/470)
Densenet (Madry retrained model)	Product	91.06% (428/470)
Densenet (TranFuzz retrained)	Product	93.40% (439/470)

Table 1: Transfer accuracy of the target model **DenseNet-121** on the source model using different target datasets

First of all, we can see that the transfer accuracy is highly dependent on the dataset used and its size. The Webcam dataset, which is comparatively small and less complex, achieves near-perfect transfer accuracy (98.90%) across all three source models, showing that for simpler domains, even non-robust models are sufficient for effective transfer.

In contrast, for more challenging datasets like Amazon and Product, we observe notable performance gaps. On Amazon, the non-robust model achieves 76.79% accuracy, while adversarially retrained models (both Madry and TranFuzz) improve performance incrementally, with TranFuzz reaching 79.52%. This suggests that adversarial retraining, especially using methods like TranFuzz that incorporate domain adaptation and fuzzing-based perturbation, provides stronger generalization across domains with higher distribution shifts.

For the Product dataset, these improvements are even more significant: the non-robust model reaches 89.36%, but the TranFuzz retrained model attains 93.40%. This large margin (+4.04%) demonstrates that adversarial retraining does not just defend against attacks but also enhances the model's ability to generalize on challenging and more complex data distributions.

Another key observation is that even though Madry retraining provides improvements (e.g., from 89.36% to 91.06% on Product), TranFuzz outperforms it. This may be due to the ensemble mutation strategy and domain adaptation mechanisms described in the paper, which seem to address both black-box model and data distribution challenges simultaneously.

Finally, these results suggest that the benefit of adversarial retraining methods scales with dataset complexity. For simple domains (Webcam), improvement ceilings are quickly reached,

while for more complex domains (Amazon and Product), these methods unlock substantial performance gains.

VI.3] Attack results

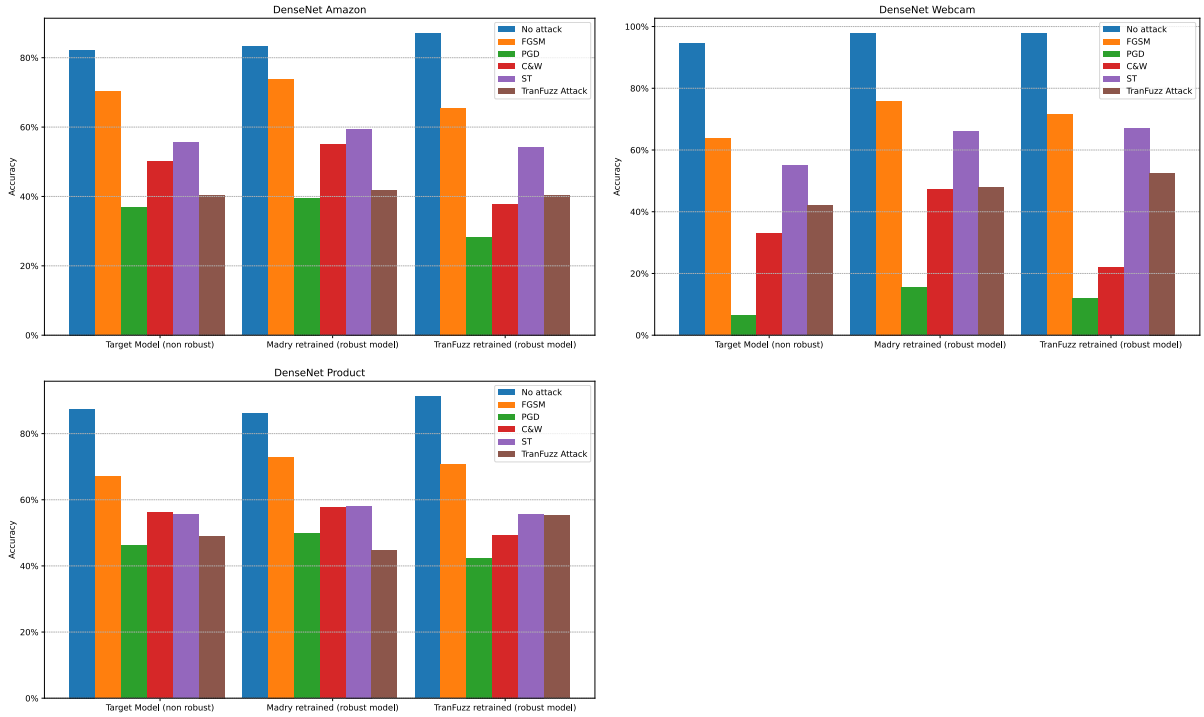

Figure 3: DensetNet-121 result charts on different datasets

First of all, we can see that the TranFuzz attack is highly promising and demonstrates its ability to effectively attack black-box models. For instance, on the DenseNet model, the accuracy drops from 94.51% to 42.03% under a TranFuzz attack, highlighting the strength and efficiency of this method in degrading model performance.

From the Figure 3, we can also observe a strong dependency on the chosen datasets. For example, the Webcam dataset, which is the smallest with only 91 test images, exhibits extremely low accuracy under the PGD attack. This might suggest that smaller datasets with less variance could make models more vulnerable to gradient-based perturbations.

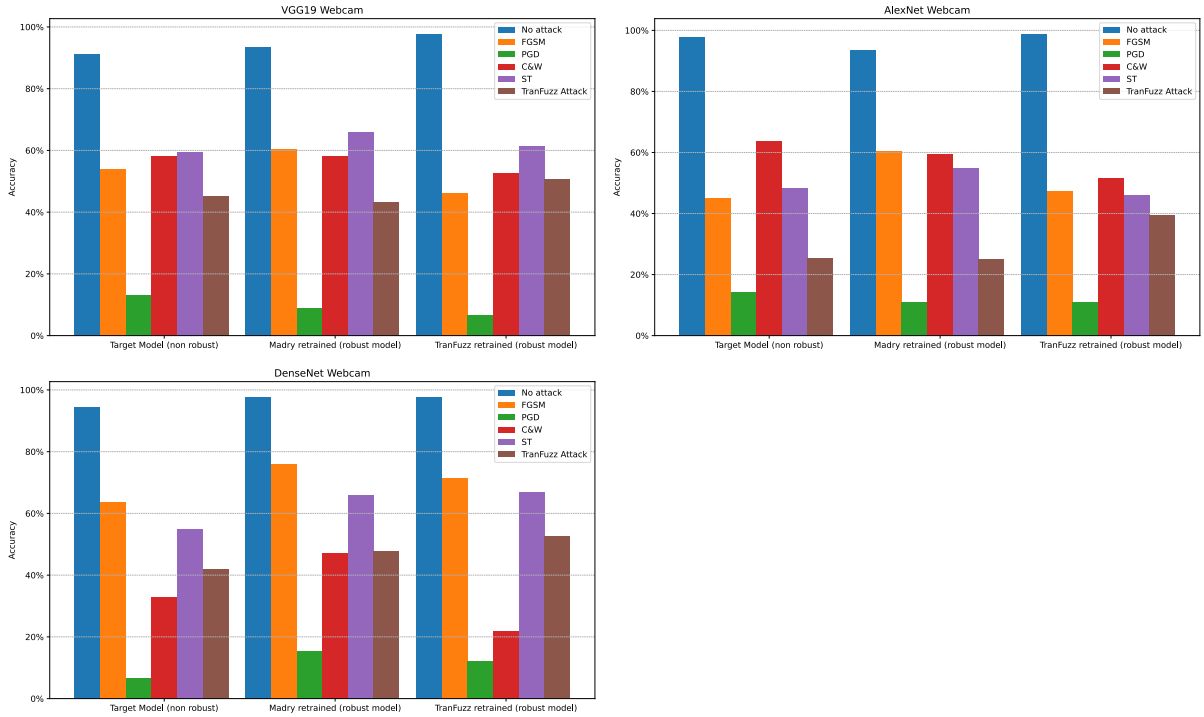

Figure 4: Office31/Webcam result charts for different model architectures

As a general observation across all models, PGD attacks consistently prove to be the most effective (as confirmed in Figure 3), drastically reducing accuracy regardless of whether the models are robust or non-robust. This reaffirms PGD's position as one of the most powerful gradient-based attack methods.

TranFuzz attacks are also highly competitive, often achieving results comparable to PGD and, in some cases, even surpassing classical white-box attacks such as C&W or ST.

While FGSM remains weaker compared to PGD and TranFuzz, it still causes significant degradation in accuracy, particularly on non-robust models, demonstrating that even simple attacks can expose vulnerabilities in poorly defended architectures.

It's important to note that these results can be interpreted in various ways, as they may depend on the target model's architecture, the complexity of the dataset, and the training process.

However, one consistent observation is that models retrained with TranFuzz-generated adversarial images demonstrate some increased robustness to certain attacks — notably against FGSM and, to some extent, TranFuzz attacks themselves. That said, this improvement is not universal. In particular, attacks like C&W still manage to degrade accuracy from 5% to 20% on these supposedly robust models, indicating that the defense provided by TranFuzz-based retraining is partial and does not generalize well against all attack vectors.

This suggests that while TranFuzz is a powerful tool for generating adversarial examples and can enhance resilience in certain scenarios, it does not yet provide a fully reliable defense. In fact, its role as a defensive tool seems limited, especially when facing sophisticated white-box attacks. It may still hold value as a complementary technique for robustness training, but relying on it alone for defense is insufficient.

The results of our experiments suggest that we should prioritize the use of a model like PGD to attack a known target model (white box), and choose TranFuzz rather than ST in the case

of an attack on an unknown target model (black box), as is generally the case in real-world conditions.

VI.4] Limitations

The main limitation of our experiments comes from the datasets used, particularly their size. With only 704 images, the Office-31 Webcam dataset is the smallest. Splitting the training and validation data using a 90/10 ratio makes the accuracy highly volatile, making it difficult to reproduce the results. Moreover, in an effort to make the study more accessible, the research paper omitted some points that are nevertheless essential for reproducing the experiments. Specifically, information about key parameters such as the learning rate or the optimizer was not provided, forcing us to rely on the values already implemented in the code. We also limited our research to a small set of attacks and defense strategies. We also largely limited our study to rather old architectures of image classification models : AlexNet (2012), VGG (2014) and DenseNet (2017), while the source model is a simple ResNet. We got some numbers to confidently give a conclusion about the results from [1].

VII] Conclusion

Throughout this project, we used the open-source code for the TranFuzz system, which provides a way to perform black-box adversarial attacks. We were able to effectively reproduce the results from the original paper [1], with some slight variations in some configurations. Since these variations resulted in performance that was sometimes slightly better and sometimes slightly worse, we believe it is unlikely the original results were hand-picked to be good. The variations can be due to a lot of potential reasons : improvements in the code after the initial release of the paper, changes in the system seeds for the pseudo-random number generation (especially for the stochastic processes happening for the deep learning algorithms).

Numerous avenues exist for future exploration. Firstly, expanding the scope of our analysis to encompass a more comprehensive range of adversarial attacks and defense mechanisms is crucial. Secondly, evaluating the robustness of domain adaptation techniques across a wider variety of datasets beyond Office31 and OfficeHome would provide a more generalizable understanding of their effectiveness. Also, the dataset splits were done only once using a random algorithm with a fixed seed. This results in a reproducible work but that is far from exhaustive, and also we decided to use a repartition of 90%/10% for the splits, and that causes the test datasets to have only dozens of images, especially for Office31/Webcam dataset.

Finally, investigating the performance of these techniques on more modern and complex architectures, such as Transformers or EfficientNets, is essential to assess their applicability to contemporary image classification models. Such investigations could reveal novel vulnerabilities or uncover more effective defense strategies tailored to these advanced architectures.

VIII] Bibliography

[1] H. Li, S. Guo, P. Tang, C. Hu, and Z. Chen, "TranFuzz: An Ensemble Black-Box Attack Framework Based on Domain Adaptation and Fuzzing," vol. 12918, pp. 260–275, 2021, doi: 10.1007/978-3-030-86890-1_15.

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014, doi: 10.48550/arXiv.1412.6572.

[3] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: a simple and accurate method to fool deep neural networks," *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, doi: 10.1109/CVPR.2016.282.

[4] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks." [Online]. Available: https://arxiv.org/abs/1706.06083

[5] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, doi: 10.1109/SP.2017.49.

[6] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, and D. Song, "Spatially Transformed Adversarial Examples." [Online]. Available: https://arxiv.org/abs/1801.02612

[7] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. L. Hsieh, "ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017.

[8] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, 2019.

[9] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *arXiv preprint arXiv:1712.04248*, 2018.

[10] C. Guo, C. Chen, T. Suya, Z. Wang, and W. He, "Simple black-box adversarial attacks," *arXiv preprint arXiv:1905.00441*, 2019.

[11] Y. Zhu *et al.*, "Deep Subdomain Adaptation Network for Image Classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1713–1722, Apr. 2021, doi: 10.1109/tnnls.2020.2988928.

[12] M.-I. Nicolae *et al.*, "Adversarial Robustness Toolbox v1.2.0," *CoRR*, 2018, [Online]. Available: https://arxiv.org/pdf/1807.01069

[13] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=BJx040EFvH

IX] Appendix

Dataset	Office31/ Amazon	Office31/ Webcam	OfficeHome/ RealWorld	OfficeHome/ Product
Number of classes	31	31	65	65
Train split (90%)	2524	704	3892	3969
Test split (10%)	293	91	465	470
Total	2817	795	4357	4439

Table 2: Number of images per dataset

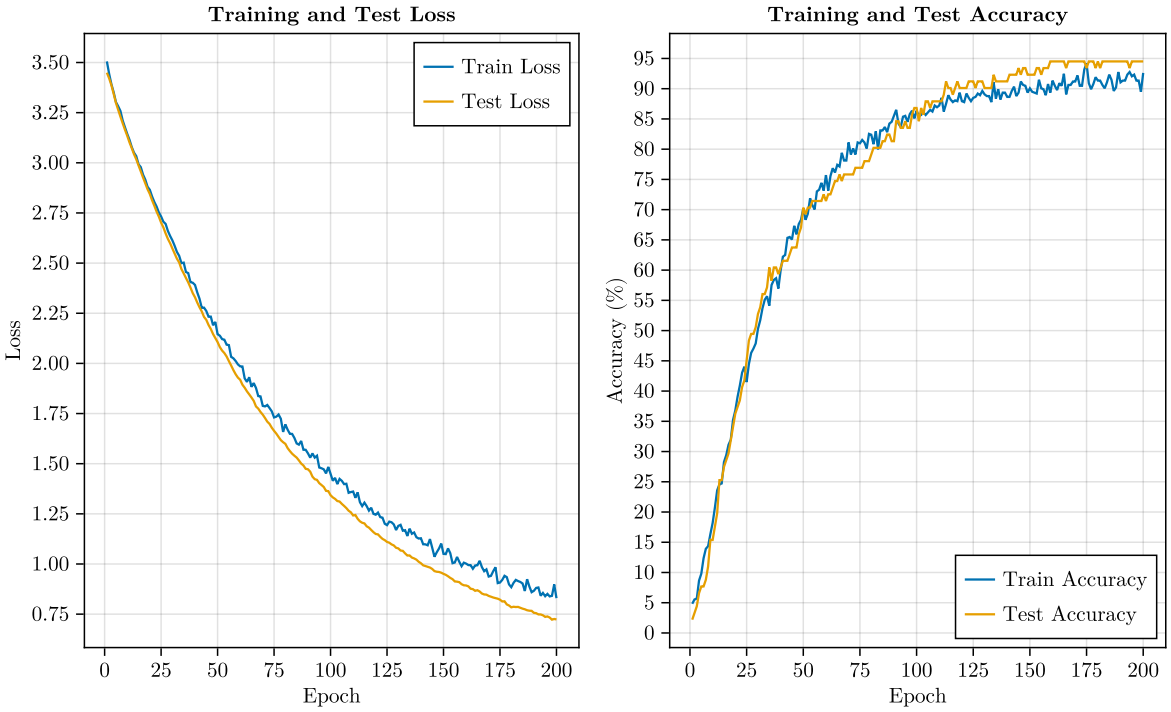

Figure 5: Training graph of the target model (DenseNet-121 on Office31/Amazon)

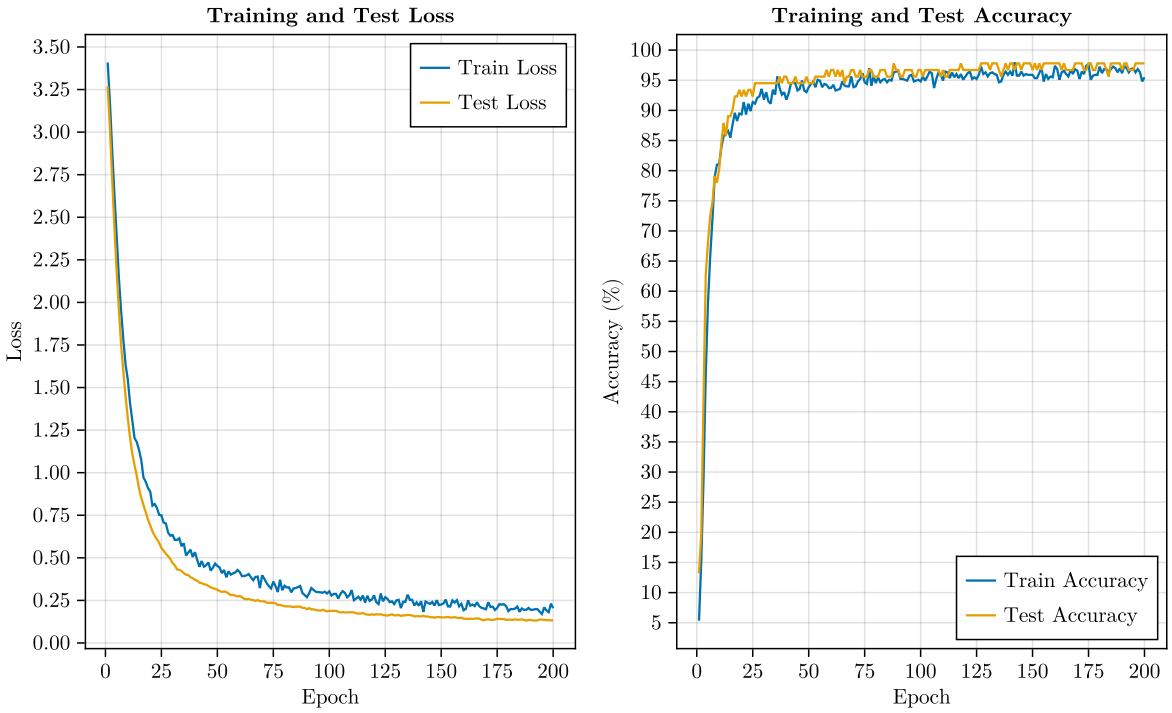

Figure 6: Training graph of the TranFuzz robust target model (DenseNet-121 on Office31/ Amazon)

Model/Accuracy	No attack	FGSM	PGD	C&W	ST	TranFuzz Attack
Target Model (non robust)	94.51%	63.74%	6.59%	32.97%	54.95%	42.03%
Madry retrained (robust model)	97.80%	75.82%	15.38%	47.25%	65.93%	47.83%
TranFuzz retrained (robust model)	97.80%	71.43%	12.09%	21.98%	67.03%	52.54%

Table 3: Target Model **DenseNet-121** on **Office31/Webcam** target dataset

Model/Accuracy	No attack	FGSM	PGD	C&W	ST	TranFuzz Attack
Target Model (non robust)	82.25%	70.31%	36.86%	50.17%	55.63%	40.48%
Madry retrained (robust model)	83.28%	73.72%	39.59%	54.95%	59.39%	41.67%
TranFuzz retrained (robust model)	87.03%	65.53%	28.33%	37.88%	54.27%	40.48%

Table 4: Target Model **DenseNet-121** on **Office31/Amazon** target dataset

Model/Accuracy	No attack	FGSM	PGD	C&W	ST	TranFuzz Attack
Target Model (non robust)	87.45%	67.02%	46.17%	56.17%	55.53%	49.09%
Madry retrained (robust model)	86.17%	72.98%	49.79%	57.66%	58.09%	44.75%
TranFuzz retrained (robust model)	91.28%	70.64%	42.34%	49.36%	55.74%	55.25 %

Table 5: Target Model **DenseNet-121** on **OfficeHome/Product** target dataset

Model/Accuracy	No attack	FGSM	PGD	C&W	ST	TranFuzz Attack
Target Model (non robust)	97.80%	45.05%	14.29%	63.74%	48.35%	25.36%
Madry retrained (robust model)	93.41%	60.44%	10.99%	59.34%	54.95%	25.00%
TranFuzz retrained (robust model)	98.90%	47.25%	10.99%	51.65%	46.15%	39.49%

Table 6: Target Model **AlexNet** on **Office31/Webcam** target dataset

Model/Accuracy	No attack	FGSM	PGD	C&W	ST	TranFuzz Attack
Target Model (non robust)	91.21%	53.85%	13.19%	58.24%	59.34%	45.29%
Madry retrained (robust model)	93.41%	60.44%	8.79%	58.24%	65.93%	43.12%
TranFuzz retrained (robust model)	97.80%	46.15%	6.59%	52.75%	61.54%	50.72%

Table 7: Target Model **VGG-19** on **Office31/Webcam** target dataset