CIS 6261: Trustworthy Machine Learning

Final Project Report [Option 1]: Recommended Project

Ammar Amjad
(Point of Contact)
ammar.amjad@ufl.edu

Mohammad Anas
mo.anas@ufl.edu

Mohammad Uzair Fasih
mfasih@ufl.edu

April 27, 2023

1 Introduction

Machine learning models can be susceptible to all kinds of attacks, posing risks to their reliability and trustworthiness. These attacks come in different forms, such as adversarial attacks (like FGSM or PGD) and data privacy attacks (like Membership Inference Attacks) which aim to exploit vulnerabilities within the model and compromise the privacy of individuals whose data was used during training. These attacks can undermine the accuracy, fairness, and integrity of machine learning models, negatively impacting user trust and stakeholder confidence. Therefore, it is crucial to develop and deploy machine learning models with robust defenses against these attacks.

In the scope of this project, our objective is to identify and evaluate potential defenses against different types of attacks. To adhere to the specified requirements and ground rules, we conducted a survey of defenses that do not rely on training with adversarial examples. As a result, we identified three categories of defenses to be the subject of our study:

1. **Input data filtering or Denoising techniques** that aim to either remove perturbations from the input data or reduce the effectiveness of adversarial perturbations.
2. **Runtime Adversarial training** which can improve the model's robustness without involving training of the model from scratch.
3. **Certified Robustness** approaches that provide provable guarantees of model robustness against adversarial examples. We focus on Randomized Smoothing.

While there are other approaches for defending against adversarial attacks, we decided to limit our survey to the aforementioned categories due to time constraints and the inherent trade-offs associated with these approaches.

2 Background & Related Work

For our project, we started our analysis with a review of recent findings on adversarial attacks and a detailed understanding of the attack models and methodologies.

We start with a technique proposed by **Certified Adversarial Robustness via Randomized Smoothing** [2]. The authors propose a technique called *"randomized smoothing"* that can turn any classifier that performs well under *Gaussian noise* into a new classifier that is robust to adversarial perturbations with l_2 norm. The paper has been cited extensively in subsequent research papers and has

received attention from the machine learning community for its contribution to the field of adversarial robustness. We implement a defense using this approach and analyse our findings.

We then follow up our analysis with **Runtime Adversarial training**. There are several papers [4] and [9]. The paper [4] proposes the *Fast Gradient Sign Method (FGSM)*, a simple yet effective method for generating adversarial examples and it demonstrates how adversarial examples can be used to improve the robustness of machine learning models through adversarial training.

The paper [9] proposes a method called *Projected Gradient Descent (PGD)* for generating adversarial examples during adversarial training. It shows that adversarial training with PGD-generated examples can significantly improve the robustness of deep neural networks against various types of adversarial attacks. It demonstrates how adversarial training can result in models that are more robust to adversarial examples compared to other defense methods. We leverage these techniques to implement a defense using adversarial training without saving the model.

We conclude our survey by analysing several input pre-processing techniques. We review several techniques including **Median Filtering** [7], **Bilateral Filtering and Rotation** [6] [10] [12], **3 Bit Depth Reduction** [7] [13], **Non-local Mean** [7] [13] [1], **Rotation by 10%** [7] [13], **JPEG Compression by 90%** [8] [5] [3] and **TVM** [8] [5]. An exhaustive list of all techniques subject to our survey is presented later in table 3.

3 Approach: Dataset(s) & Technique(s)

3.1 Defenses

In this section, we summarize our used approaches with emphasis on the defense and our analysis.

3.1.1 Certified Robustness

We began our survey with **Certified Robustness** which is known for its generality and applicability to various types of adversarial attacks. Furthermore, this approach comes with provable guarantees and verifiable assurances that the model can withstand perturbations up a certain level making it a effective baseline for our investigation.

We implemented this defense as follows:

Input : Training data D, pretrained neural network f, smoothing radius σ
Output: Smoothed classifier g
`//Train` f `with Gaussian data augmentation`
Train f with D using Gaussian data augmentation at variance σ^2;
`//Create smoothed classifier` g
for x *in training set* D **do**
  `//Generate samples for` x
  $samples \leftarrow SampleGaussian(x, \sigma, k)$;
  `//Predictions from` f `on samples`
  $preds \leftarrow Predict(f, samples)$;
  `//Most likely class for` x `in` g
  $g(x) \leftarrow ClassWithMaxCount(preds)$;
end
return g

Algorithm 1: Smoothed Classifier Creation via Randomized Smoothing

3.1.2 Runtime Adversarial Training

We implement runtime adversarial training using the Fast Gradient Sign Method (FGSM). The FGSM-based adversarial training process is integrated during runtime in adherence with the ground rules of the project, generating adversarial examples from a subset of the training data. We carefully tuned the epsilon value, which determines the magnitude of the perturbations, to strike a balance between adversarial robustness and accuracy on clean data. We evaluate the performance by measuring its accuracy and robustness against adversarial examples.

We implemented our defense as follows:

Input: Pre-trained model f, training data X, labels Y, learning rate α, number of epochs N
Output: Adversarially trained model f_{adv}
Hyperparameter: Perturbation strength ϵ
Initialize model parameters of f_{adv} with those of f; **for** $n = 1$ *to* N **do**
  Randomly select a mini-batch of training samples $(x, y) \in (X, Y)$;
  Generate adversarial examples $x_{adv} = x + \epsilon \cdot sign(\nabla_x J(f_{adv}(x), y))$, where J is the loss function;
  Update model parameters using gradient descent: $f_{adv} \leftarrow f_{adv} - \alpha \nabla_\theta J(f_{adv}(x_{adv}), y)$, where θ denotes the parameters of f_{adv};
end

Algorithm 2: Adversarial Training with FGSM

3.1.3 Input data filtering or Denoising techniques

Preprocessing-based defenses, such as image filtering, help defend against these attacks but often result in the loss of valuable features and reduced classification accuracy. We conducted a series of experiments to evaluate the effectiveness of different filters in mitigating adversarial and membership inference attacks. To achieve this, we tested the pre-trained model in conjunction with a custom predict function, applying various filters in each iteration and evaluating the accuracy of membership inference and adversarial attacks. Then, we refined our experiment further using the result from the last step. The results of our experiments are presented later in table 3.

One paper we identified with promise and evaluated is Feng et. al.'s paper on adversarial defense [6]. This paper proposes a new defense method that uses different filter parameters and randomly rotated filtered images. By averaging the output classification probabilities, the method is claimed to maintain accuracy while removing perturbation.

The algorithm is shown in the figure below:

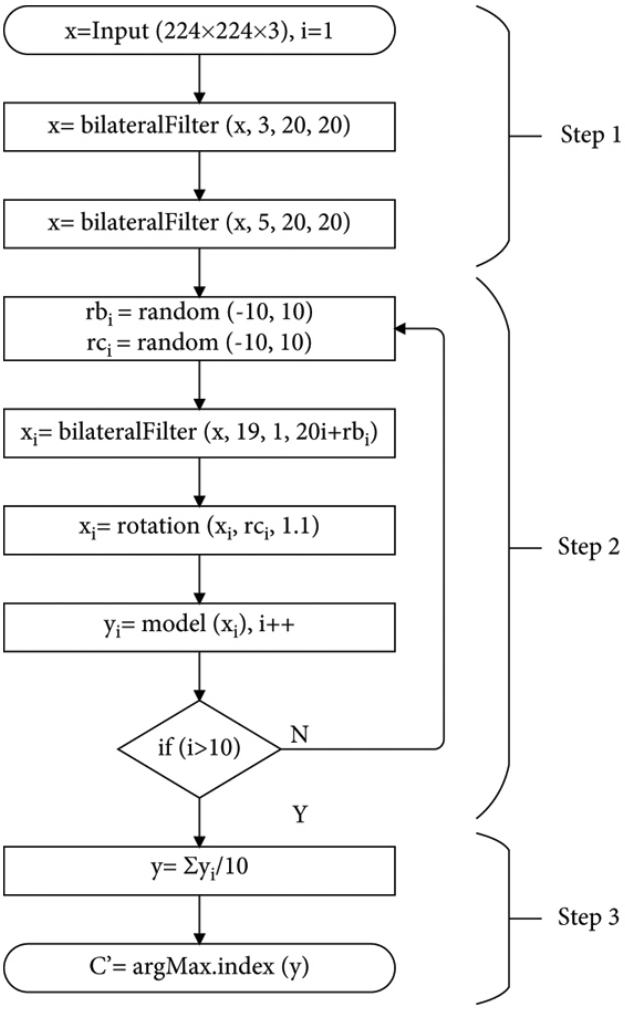

Figure 1: Filtering-based Defense Algorithm

The basic concept is to apply bilateral filtering with rotation and average the results of predictions over 10 rounds.

Despite the paper claiming to achieve great results, in practice, the experimental results indicate no improvement in defense against various adversarial attacks. The results can be seen in Table 3. It's evident that the MIA results are satisfactory but the compromise on benign and adversarial accuracy makes this method impractical.

Secondly, we evaluated Shokri et al.'s defenses or mitigation strategies[11]:

1. Restrict the prediction vector to the top k classes where k = 3.
2. Applying Rounding d = 3.
3. Applying Temperature t = 5.

The Implementation of defenses can be found in provided Python file.

Nonetheless, the methods gave unsatisfactory results and were ultimately not chosen as our final defense.

Some other filtering techniques applied which were inspired from Liu et. al's paper [7] and other works: TVM [8] [5] , JPEG Compression [8] [5] [3] , Image Rotation [7] [13] , 3 Bit Depth Reduction [7] [13] and Non-local Mean(11-3-4) [7] [13] [1].

The exhaustive list of different defenses we implemented is mentioned in Table 3 along with the accuracies and effectiveness of each defense.

3.2 Dataset

For Part 2 of our project, we used the Fashion-MNIST . Fashion-MNIST is a publicly available dataset widely used in machine learning research. It consists of grayscale images of fashion items with a size of 28x28 pixels. Similar to the original MNIST dataset, it includes a total of 60,000 training images and 10,000 test images. Each image in Fashion-MNIST belongs to one of 10 classes, representing different fashion categories such as dresses, shoes, bags, and more, making it a drop in replacement for MNIST. Additionally, Fashion-MNIST shares the same number of channels (i.e., one channel for grayscale) as the original MNIST dataset. A preview of our dataset is as follows:

Figure 2: Fashion-MNIST Dataset Preview

3.3 Model

For Part 2 of the project, we implemented a convolutional neural network (CNN) model for classifying images from the Fashion MNIST dataset. The outline of the architecture is outlined as follows:

- `Conv2D` layer with 64 filters, a kernel size of $(3, 3)$, ReLU activation function, and L2 regularization with a regularization strength of 0.01.
- `BatchNormalization` layer, which normalizes the activations of the previous layer.
- `Conv2D` layer with 64 filters, a kernel size of $(3, 3)$, ReLU activation function, and L2 regularization with a regularization strength of 0.01.
- `BatchNormalization` layer.

- `Conv2D` layer with 128 filters, a kernel size of $(3, 3)$, ReLU activation function, and L2 regularization with a regularization strength of 0.01.
- `BatchNormalization` layer.
- `MaxPooling2D` layer with a pool size of $(2, 2)$, which performs max pooling on the previous layer's output.
- `Dropout` layer with a dropout rate of 0.5, which helps to regularize the model by randomly dropping out input units during training.
- `Flatten` layer, which flattens the previous layer's output to a 1D tensor.
- `Dense` layer with 128 units, ReLU activation function, and L2 regularization with a regularization strength of 0.01.
- `Dropout` layer with a dropout rate of 0.5.

Our apporach for implementing the defense involves leveraging the insights gleaned from the first part of our project and applying them to the specified dataset and model. This approach will enable us to rigorously assess the effectiveness of the identified defense mechanisms within a diverse context, contributing to a more robust evaluation of its potential impact on mitigating adversarial attacks.

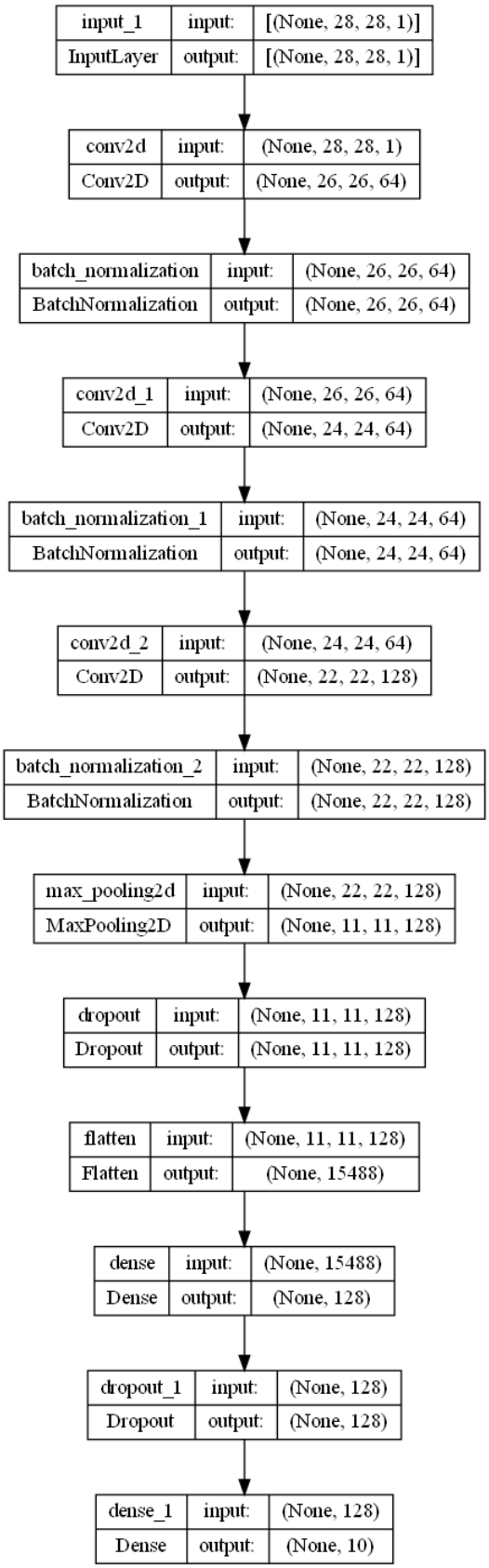

Figure 3: Model Architecture

4 Results

In this section, we provide a brief discussion about our results with our chosen defense strategies.

4.1 Part 1

Using the **Certified Robustness** approach, we were able to obtain somewhat decent results by varying the values of sigma, the hyperparameter that controls the magnitude of perturbations applied to the input data during the robustness certification process. By carefully tuning the sigma values, we achieved promising outcomes in terms of model performance and robustness against adversarial examples.

Our results with this approach are as presented in Table 1.

sigma	Noise Type	num_samples	lambd	Benign Accuracy 0	Benign Accuracy 1	Adversarial Accuracy 0	Adversarial Accuracy 1
0.01	Laplace	10	1.0	95.00%	97.00%	0.00%	6.50%
0.10	Laplace	100	1.0	50.00%	53.50%	30.00%	30.50%
0.10	Gaussian	10	1.0	80.00%	25.00%	79.00%	47.50%
0.10	Gaussian	100	1.0	85.00%	20.00%	79.00%	47.50%

Table 1: Certified Robustness using Randomized Smoothing.

Some of our observations are as follows:

- Higher sigma generally leads to lower Benign Accuracy 0 and Benign Accuracy 1.
- Adversarial Accuracy 0 tends to increase with higher sigma.
- Adversarial Accuracy 1 generally increases with higher sigma.
- Gaussian noise type with 100 num_samples shows consistent Benign Accuracy 0 at 85.00%, but Benign Accuracy 1 decreases from 20.00% to 47.50% with higher sigma.

Next, we implemented a defense using adversarial training. We trained the model using adversarial examples crafted from a subset of the training dataset using the Fast Gradient Sign Method (FGSM). We evaluated the adversarial training results for varying alpha and epochs, both with and without the filtering defense. Our results are recorded in Table 2.

Finally, we applied different filters to the input data and evaluated the accuracy of the model on test data, adversarial examples, and membership inference attacks using input preprocessing techniques. Table 3 summarizes the results of our experiments. We observed that maximum filtering, median filtering, and gaussian-laplace filtering were effective in defending against membership inference attacks, reducing their accuracy to around 50%. The median filtering with double filter technique[7] was found to be the most effective in defending against adversarial attacks in our study. It reduced the accuracy of the attacks to 54.80%, while the training accuracy of the model was 77.74%, and the test accuracy was 56.22%. The technique replaces each pixel in an image with the median value of its neighboring pixels, which smooths out variations and noise in the image, making it harder for adversarial attacks to manipulate. Applying the filter twice further enhances the effect of the filter.

In the next stage, we tested a novel approach that combined Gaussian filtering with the previously used double median-filter technique. The results showed that combining Gaussian and median filtering

improved the model's ability to defend against adversarial examples. However, in this method, the overall test accuracy of the model was still low. Similarly, other combinations were also evaluated but we did not find them to be effective.

Based on extensive experiments, we determined that the most effective defense using input preprocessing is Median Filtering with a dual filter technique. As shown in the Table 3, this filter achieves an attack accuracy of around 50% while maintaining a high adversarial accuracy that is comparable to benign accuracy. Additionally, it strikes a reasonable balance between reducing the effectiveness of adversarial examples and membership inference attacks and maintaining accuracy on test data.

Filtering Technique	**Train Acc.**	**Test Acc.**	**MIA Attack Acc.**	**Benign Example 0 Acc.**	**Adversarial Example 0 Acc.**	**Benign Example 1 Acc.**	**Adversarial Example 1 Acc.**
No Defense	100.00%	75.18%	74.25%	95.00 %	0.00%	96.00%	0.00%
No perturbations - unfitted	77.74%	56.22%	54.80%	85.00 %	70.00%	72.50%	68.00%
Adversarial Training (alpha=0)	78.76%	57.44%	54.90%	80.00%	65.00%	72.50%	68.50%
Adversarial Training (alpha=0.025)	78.80%	55.02%	53.77%	80.00%	80.00%	73.50%	70.00%
Adversarial Training (alpha=0.035)	75.52%	52.84%	52.80%	80.00%	85.00%	72.00%	69.00%
Adversarial Training (alpha=0.05)	74.50%	51.28%	52.28%	90.00%	90.00%	69.00%	68.00%
Adversarial Training (alpha=0.1)	52.94%	38.90%	50.22%	55.00%	65.00%	50.50%	46.50%
Adversarial Training (alpha=0.2)	20.10%	18.36%	49.98%	10.00%	10.00%	17.00%	17.50%
Adversarial Training (alpha=0.05) - 1 epoch	59.04%	46.86%	50.00%	75.00%	70.00%	53.00%	48.00%
Adversarial Training (alpha=0.05) - 10 epochs	99.58%	64.14%	58.88%	100.00%	95.00%	95.00%	93.50%
Adversarial Training (alpha=0.05) - single median filter - 10 epoch - 32 batch	88.52%	59.28%	55.88%	90.00%	90.00%	79.50%	78.00%
Adversarial Training (alpha=0.075) - 10 epochs	97.56%	61.96%	59.23%	90.00%	95.00%	91.00%	90.50%
Adversarial Training (alpha=0.085) - 10 epochs - batch 16	91.18%	62.50%	55.12%	90.00%	90.00%	89.00%	88.00%

Table 2: Adversarial training results varying alpha and epochs with and without filtering defense.

Based on the data in Table 2, several trends can be observed regarding adversarial training results:

1. As the alpha value increases, both the train accuracy and test accuracy generally decrease. This indicates that as the adversarial perturbation becomes stronger (higher alpha), the model's performance on both train and test sets tends to worsen.

2. As the alpha value increases, the MIA attack accuracy decreases up to a certain point, after which it increases again. This suggests that adversarial training is effective in mitigating MIA attacks when the alpha value is chosen appropriately. However, overly strong perturbations (high alpha) can make the defense less effective.

3. The benign and adversarial example accuracies also show a trend related to the alpha value. As alpha increases, the accuracy on adversarial examples tends to improve while the accuracy on benign examples decreases, indicating a trade-off between robustness against adversarial attacks and performance on benign examples.

4. Increasing the number of training epochs generally improves both the benign and adversarial example accuracies, suggesting that more training can lead to better performance and robustness against adversarial attacks.

5. The use of a single median filter in the adversarial training process appears to improve the adversarial example accuracy without significantly affecting the benign example accuracy. This indicates that incorporating a filtering defense can enhance the model's robustness against adversarial attacks while maintaining performance on benign examples

6. The adversarial training takes around 1200 seconds or 20 minutes for the whole run of the code which is a reasonable amount of time.

Filtering Technique	**Train Acc.**	**Test Acc.**	**MIA Attack Acc.**	**Benign Example 0 Acc.**	**Adversarial Example 0 Acc.**	**Benign Example 1 Acc.**	**Adversarial Example 1 Acc.**
No Defense	100.00%	75.18%	74.25%	95.00 %	0.00%	96.00%	0.00%
Fourier Gaussian (sigma=1)	10.46%	10.06%	50.00%	15.00%	15.00%	11.00%	11.00%
Fourier Uniform (size=3)	10.46%	10.06%	50.00%	15.00%	15.00%	11.00%	11.00%
Fourier Ellipsoid (size=3)	10.34%	10.08%	50.00%	15.00%	15.00%	14.50%	14.50%
Maximum (size=3)	59.24%	47.54%	52.23%	75.00%	65.00%	52.50%	53.00%
Uniform (size=3)	73.04%	50.58%	54.97%	65.00%	50.00%	68.50%	60.00%
Median (size=3)	88.54%	60.72%	57.05%	80.00%	65.00%	85.00%	76.50%
Gaussian (sigma=1.0)	62.00%	44.46%	53.25%	65.00%	50.00%	62.00%	50.00%
Minimum (size=3)	61.50%	47.60%	51.62%	60.00%	40.00%	58.50%	32.50%
Spline	92.88%	68.80%	59.35%	80.00%	0.00%	93.00%	36.00%
Gaussian Laplace (sigma=1)	29.72%	27.24%	50.15%	20.00%	15.00%	23.50%	19.50%
Laplace	30.90%	29.42%	49.70%	25.00%	5.00%	26.50%	22.00%
Gaussian + Median (sigma=1.0)	80.66%	54.76%	53.35%	75.00%	60.00%	77.00%	67.50%
Median (shape= (2,2,2))[7]	77.74%	56.22%	54.80%	85.00%	70.00%	72.50%	68.00%
Bilateral Filtering and Rotation[6] [10] [12]	25.48%	23.56%	50.05%	25.00%	30.00%	23.00%	19.50%
3 Bit Depth Reduction [7] [13]	10.60%	10.34%	50.08%	10.00%	10.00%	14.00%	14.00%
Non-local Mean [7] [13] [1]	35.74%	31.02%	49.85%	40.00%	45.00%	23.50%	26.50%
Rotation 10% [7] [13]	36.98%	33.62%	49.48%	50.00%	40.00%	26.00%	27.00%
JPEG Compression 90% [8] [5] [3]	10.46%	10.06%	50.00%	15.00%	15.00%	11.00%	11.00%
TVM [8] [5] (weight=0.1)	67.06%	49.78%	53.45%	60.00%	50.00%	65.00%	57.50%

Table 3: Performance comparison of various filtering techniques on the dataset.

Based on the data in Table 3, several trends can be observed regarding the filtering techniques:

1. Applying no defense provides the highest train and test accuracies, but the worst performance on adversarial examples, which indicates a lack of robustness against adversarial attacks.

2. Fourier-based filtering techniques (Fourier Gaussian, Fourier Uniform, and Fourier Ellipsoid) result in low train and test accuracies, as well as poor performance on both benign and adversarial examples. This suggests that these techniques might not be effective defenses against adversarial attacks while maintaining the performance on benign examples.

3. Maximum, Uniform, and Median filtering techniques show better performance on both benign and adversarial examples compared to Fourier-based filtering techniques, indicating that they provide a better balance between performance and robustness against adversarial attacks.

4. The Median filtering technique seems to provide a good trade-off between performance on benign examples and robustness against adversarial attacks, as it has relatively high accuracies on both types of examples.

5. Other filtering techniques, such as Gaussian, Minimum, Spline, and Gaussian + Median, show varying degrees of effectiveness in terms of train and test accuracies, as well as performance on benign and adversarial examples. The effectiveness of these techniques depends on their specific parameters and implementation.

6. Some techniques, such as 3 Bit Depth Reduction, Non-local Mean, and JPEG Compression, result in low train and test accuracies, as well as poor performance on both benign and adversarial examples, suggesting that they might not be effective defenses against adversarial attacks while maintaining performance on benign examples.

7. The best technique is to use a 2x2 median filter twice as it gives decent accuracies while providing the most robustness against adversarial and MIA attacks.

4.2 Part 2

In part 2 of the project, we trained a model from scratch on a brand new dataset and evaluated the model against adversarial examples. We generated adversarial examples using FGSM (Fast Gradient Sign Method) and evaluated the model's performance. To mitigate the effects of adversarial examples, we applied a defense technique using median filtering with dual mode technique and adversarial training that we identified to be particularly effective in part 1 of the project. We compared the results of the model with and without the defense to assess its effectiveness in improving the model's robustness against adversarial attacks.

In our methodology, we adopted a robust defense strategy by initially conducting adversarial training, followed by the application of a 2x2 median filter twice, both individually and in combination. Subsequently, these processed images were utilized for making predictions and conducting comprehensive analyses.

We present our findings in Table 4 as follows.

Configuration	**Train Acc.**	**Test Acc.**	**FGSM Benign Acc.**	**FGSM Adv. Acc.**	**MIA Attack Acc.**	**MIA F1**
Median filter only	70.56%	67.41%	75.0%	60.0%	51.10%	0.210
Adversarial Training only (alpha=0.085 epochs=20, batch_size=16)	86.98%	83.24%	100.0%	45.0%	50.12%	0.222
2x2x2 Median + Adversarial Training $\alpha = 0.085$, $epochs = 20$, $batch\_size = 16$	65.70%	64.38%	75.0%	65.0%	50.48%	0.059
2x2x2 Median + Adversarial Training $\alpha = 0.085$, $epochs = 50$, $batch\_size = 16$	71.74%	69.23%	75.0%	45.0%	50.30%	0.039
2x2x2 Median + Adversarial Training $\alpha = 0.085$**,** $epochs = 25$**,** $batch\_size = 16$	78.16%	74.44%	80.0%	70.0%	50.08%	0.012
2x2x2 Median + Adversarial Training $\alpha = 0.085$, $epochs = 25$, $batch\_size = 32$	75.52%	72.97%	85.0%	45.0%	49.93%	0.020

Table 4: Results of different configurations

From the above table, it can be inferred that the defense that combines both adversarial training and median filtering results in higher accuracy, less overfitting and better robustness to adversarial inputs while keeping MIA attacks accuracy close to 50%. Combining adversarial training and median filtering leverages the strengths of both techniques to achieve higher accuracy against adversarial input. Adversarial training makes the model more robust by learning from adversarial examples during training, while median filtering acts as a preprocessing step, reducing the impact of adversarial noise by smoothing the input. Together, these techniques complement each other, creating a model that is more robust against adversarial attacks than using either method alone.

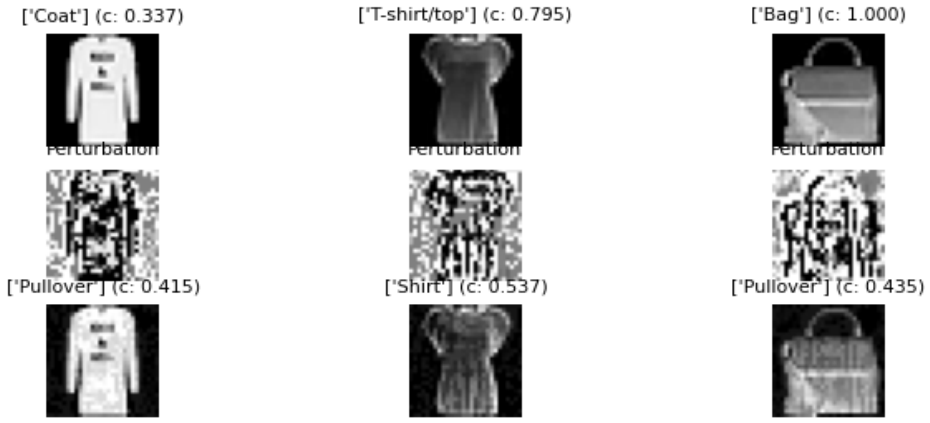

Figure 4: Raw Model - Untargeted FSGM

Figure 5: With Defense - Untargeted FSGM

5 Conclusions

Based on the data the effectiveness of filtering techniques and adversarial training methods in defending against adversarial attacks varies significantly. Some methods, like Median filtering and certain Adversarial Training configurations (e.g., alpha=0.05 and 10 epochs), provide a better trade-off between performance on benign examples and robustness against adversarial attacks.

In Part 1 of our experiments, we systematically investigated various defense mechanisms and configurations to identify the optimal approach for enhancing the performance of our model when applied to the given dataset. Through a rigorous evaluation process, we determined that adversarial training with the Fast Gradient Sign Method (FGSM) and median filtering were the most effective defenses.

Our findings enabled us to eliminate certified robustness as a viable option, primarily due to its higher time requirements for achieving superior accuracy levels. This characteristic is not aligned with the time

constraints specified in the project's ground rules.

Using the defenses identified in part 1, we tried to come up with a defense in part 2 of the project. We employed a Convolutional Neural Network (CNN) model and the Fashion-MNIST dataset to develop a robust defense strategy against adversarial attacks. Upon extensive analysis, we discovered that a combination of adversarial training and a median filter (2x2x2 Median + Adversarial Training $\alpha = 0.085$, $epochs = 25$, $batch\_size = 16$), yielded the most optimal results. This may be attributed to the nature of our model and dataset, therefore it is not enough to generalize the defense.

In conclusion, no single technique or approach appears to be universally effective. The choice of a defense method should be based on the specific problem, desired balance between performance and robustness, and the nature of the adversarial attacks expected. The best approach may involve combining multiple defense techniques or fine-tuning the defense parameters to achieve the desired level of security while maintaining acceptable performance on benign examples.

In future research endeavors, we would like to conduct a comprehensive analysis of ensemble models and certified robustness techniques to enhance our understanding of their efficacy in defending against adversarial attacks. Additionally, we would like to extend our investigations to encompass tabular and text-based datasets, in order to gain a more holistic perspective on the applicability and effectiveness of our proposed defense strategies across various domains.

References

[1] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65 vol. 2, 2005.

[2] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019.

[3] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[5] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[6] Feng Li, Xuehui Du, and Liu Zhang. Adversarial attacks defense method based on multiple filtering and image rotation. *Discrete Dynamics in Nature and Society*, 2022, 2022.

[7] Hui Liu, Bo Zhao, Minzhi Ji, Yuefeng Peng, Jiabao Guo, and Peng Liu. Feature-filter: Detecting adversarial examples by filtering out recessive features. *Applied Soft Computing*, 124:109027, 2022.

[8] Zihao Liu, Qi Liu, Tao Liu, Nuo Xu, Xue Lin, Yanzhi Wang, and Wujie Wen. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 860–868. IEEE, 2019.

[9] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[10] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pages 582–597. IEEE, 2016.

[11] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[12] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 839–846, 1998.

[13] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.