©Copyright 2022

Dat Tien Le

Emulated Autoencoder: A Time-Efficient Image Denoiser for Defense of Convolutional Neural Networks against Evasion Attacks

Dat Tien Le

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science in Computer Science & Software Engineering

University of Washington

2022

Committee:

Brent Lagesse

Clark Olson

Afra Mashhadi

Program Authorized to Offer Degree:
Computing and Software Systems

University of Washington

Abstract

Emulated Autoencoder: A Time-Efficient Image Denoiser for Defense of Convolutional Neural Networks against Evasion Attacks

Dat Tien Le

Chair of the Supervisory Committee:
Associate Professor Brent Lagesse
Computing and Software Systems

As Convolutional Neural Networks (CNN) have become essential to modern applications such as image classification on social networks or self-driving vehicles, evasion attacks targeting CNNs can lead to damage for users. Therefore, there has been a rising amount of research focusing on defending against evasion attacks. Image denoisers have been used to mitigate the impact of evasion attacks; however, there is not a sufficiently broad view of the use of image denoisers as adversarial defenses in image classification due to a lack of trade-off analysis. Thus, image denoisers' costs, including training time, image reconstruction time, and loss of benign F1 scores of CNN classifiers, are explored in this thesis. Additionally, Emulated Autoencoder (EAE), which is the proposed method of this thesis to optimize trade-offs for high volume classification tasks, is evaluated alongside state-of-the-art image denoisers in the gray-box and white-box threat models. EAE outperforms most image denoisers in both the gray-box and white-box threat models while drastically reducing training and image

reconstruction time compared to the state-of-the-art denoisers. As a result, EAE is more appropriate for securing high-volume classification applications of images.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

CONVOLUTIONAL NEURAL NETWORK(CNN): A Neural Networks with convolution layers extracting spatial features of a given dataset and classifies future data from seen examples.

AUTOENCODER: a Neural Networks, which is comprised of an encoder and a decoder, creates data from learned features.

SUPERVISED LEARNING: An classification function is constructed from data and labels of a given dataset

UNSUPERVISED LEARNING: An classification function is constructed from data without labels of a given dataset

MODEL: A trained architecture that is for classifying data.

EVASION ATTACKS: Adversarial Attacks generate examples that are misclassified by classifiers.

ADVERSARIAL PERTURBATION: Modifications introduced by evasion attackers to benign examples.

ADVERSARIAL EXAMPLES: Data contains adversarial perturbation.

ADVERSARIAL DEFENSE: Defense algorithms that are used to mitigate the impacts of evasion attacks

IMAGE DENOISER: An algorithm that removes noises from given images

UNTARGETED ATTACKS: An evasion attacks against classifiers that focus on no specific classes.

ACKNOWLEDGMENTS

I would like to thank my thesis advisor Professor Brent Lagesse of the School of Science, Technology, Engineering, and Mathematics at the University of Washington. He always makes time to answer my questions as well as provide thoughtful ideas and insights to guide me toward success. Additionally, I want to thank my entire thesis committee for their unlimited support and flexibility to help me with my thesis.

Sincerely,

Dat Tien Le

Chapter 1

INTRODUCTION

In recent years, machine learning has been at the center of increasing the productivity of the modern world. For instance, machine-learning techniques have been playing a major part in advancing medical image classification, digital check processing, and autonomous vehicles. Specifically, machine-learning applications are essential to the self-driving cars developed by the company NVIDIA or Waymo [1, 2]. Furthermore, Facebook is also among the companies utilizing machine learning (ML) extensively for object detection and content classification on its social media platform [3]. The Convolutional Neural Networks (CNN), which also draws attention from the ML research community, helped achieve high performance in image classification tasks [4]. These successes have been fueling the investment into hardware such as Graphical Processing Units (GPU) and software such as PyTorch or TensorFlow to further improve the performance of CNN as well as its accessibility to the public.

However, as consumers have increasingly become dependent on the services provided by CNN, any mistakes made by CNN can impact people's lives to a certain extent. Therefore, adversaries can launch attacks against CNN to force it to make incorrect decisions. Images generated by adversaries are also known as adversarial examples. Thus, there have been several papers discussing the robustness of CNN against adversarial attacks [5]. The impact

caused by adversarial attackers against these CNNs can be severe. For example, inappropriate pictures to young viewers, crafted and posted by adversarial attackers on social media platforms, may not be classified by CNN as inappropriate [6]; therefore, they are not removed from the platforms at a large scale. This issue can cause severe mental health issues to many minors. Furthermore, traffic signs such as stop signs or speed limit signs, tampered with by adversarial attackers, can be used to manipulate self-driving vehicles to cause traffic casualties [7]. These incidents not only reduce the productivity of society but also costs people's lives. Therefore, there has been a lot of attention drawn to developing adversarial defenses for image classifiers as well as to object detection.

Image denoising was shown to be an effective defense against evasion attacks [8, 9, 10, 11]. However, the trade-offs of image denoisers were not well mentioned. This paper will explore the costs of using a group of image denoisers as adversarial defense against evasion attacks. This paper hypothesizes that there are trade-offs, including loss of benign classification performance, image reconstruction, and training time, for using several image denoisers as adversarial defense. As a result, these trade-offs may affect ML engineers' decisions when it comes to choosing appropriate defenders against evasion attacks. These trade-offs are not adequate, but necessary when it comes to selecting adversarial defenses. This work also offers engineers a broader perspective when it comes to choosing adversarial defenses as well as defender comparison in the future. This paper also proposes a time-efficient adversarial defense against evasion attacks that target image classifiers. The contributions of this thesis are:

- explores the trade-offs, including loss of benign classification performance, training and image reconstruction time, of a group of image denoisers.

- proposes a time-efficient image denoiser by emulating the structure of Autoencoder via the bilinear interpolation method. The proposed method, which is Emulated Autoencoder (EAE), was shown to be a time-efficient and effective adversarial defense in both the gray-box and white-box threat model. Except for the Variational Autoencoder (VAE), EAE outperformed all other image denoisers in terms of denoising adversarial examples with minimum trade-offs. EAE averaged 85% of the adversarial performance of VAE at only 0.8% of the image reconstruction time cost.

- improves the performance of an image denoiser by chaining it with EAE. Chained denoiser between EAE and JPEG compression consistently outperformed standalone EAE and JPEG compression against a set of adversarial attacks in the white-box threat model. Also, chaining EAE and JPEG compression helped achieve roughly 94% performance of VAE, which is the most effective against adversarial attacks in the white-box threat model, at only 75% of the image reconstruction time cost. Meanwhile, VAE-EAE chained denoiser consistently outperformed both standalone EAE and VAE on the CIFAR-100 dataset in the white-box threat model at only a 0.8% additional time cost over standalone VAE.

1.1 Motivation

Since Goodfellow et al. [5] explored that Neural Networks are prone to evasion attacks, especially on image classification tasks, there has been growing research conducting practical evasion attacks. For example, according to Yuan et al. [6], only two-thirds of sexually explicit images embedded with adversarial perturbations were caught by explicit content detection such as Google Cloud Vision API or Yahoo "Not Suitable for Work" (NSFW) Image detector. As a result, it showed that adversarial examples potentially cause damage to minors on the internet. Additionally, based on Deng et al. [12], white-box evasion attacks achieved high success in deceiving the NVIDIA DAVE-2 self-driving car model. Furthermore, practical evasion attacks conducted by having an adversarial sticker installed in front of a camera could also fool image classifiers [13]. Overall, these studies showed that evasion attacks could pose threats to real-world ML-based applications and consumers.

One of the modern adversarial defenses is image denoising, which transforms images before they are sent to the CNN for training or inference tasks. This defense mechanism not only removes random noise but also adversarial perturbation [8, 9, 10, 11]. Adversarial perturbations are typically unrecognizable by human eyes; however, this noise can control the classification decision of target Neural Networks models. However, the image denoising process may consume a tremendous amount of time in the inference phase. As a result, slow inference can have a significantly negative impact on classification tasks such as facial recognition or object identification, which commonly require fast inference performance from

modern neural architectures. For example, self-driving cars require low reaction time to ensure safety for passengers at any moment while in traffic. Similarly, inappropriate images on social media need to be removed promptly to prevent severe mental damage to vulnerable victims. Additionally, image denoising may distort the images extensively so that Neural Networks classifiers are not able to classify them correctly. Thus, the goals of this paper include understanding the trade-offs of using image denoising, comprised of drops in classification performance on normal images, training, and image reconstruction time. Moreover, finding the most prominent image denoising techniques which do not possess significant drawbacks was also part of this paper's objectives. Alternatively, a new adversarial defense, which utilizes a resizing method, was suggested and evaluated under the same testing parameters. NVIDIA RTX-6000 GPU and Intel Xeon Gold 6230 CPU were used to run experiments in this thesis. Unlike this thesis, NVIDIA DRIVE ORIN GPU is used on self-driving vehicles at NVIDIA [14]. NVIDIA RTX-6000 GPU is only capable of conducting 130.5 trillion floating-point operations per second (TFLOPS) [15]. Meanwhile, NVIDIA DRIVE ORIN GPU can achieve 200 TFLOPS [16]. The relative performance among classifiers and denoisers found in this thesis ought to be held on faster hardware.

1.2 Structure of This Thesis

A brief introduction to ML and Neural Networks are presented, which are then followed by a summary of CNN as well as transfer learning. Then, summaries of Autoencoder and adversarial attack are introduced. Next, an understanding of the related works to this paper

is established. Sequentially, the methodology is presented including this thesis' proposed method. Finally, the results along with discussions are the last two sections.

Chapter 2
BACKGROUND

In this chapter, a brief introduction to ML, Neural Networks, transfer learning, Autoencoder, and adversarial attack are presented.

2.1 Machine Learning

As society advances, a tremendous amount of data, which cannot be processed by humans alone, is being created and stored at every moment. As computer hardware has become cheaper in recent decades, it paved the way for computer software to become a crucial part of processing a large amount of data to extract insights. To do so, computer programmers can develop a set of hard-coded instructions or rules that can apply to a group of testing samples. This approach was eventually replaced by machine learning, which observes and extracts patterns from a set of input data.

In some scenarios, it is less economical to hire a human than to utilize machine learning. For instance, computers can process hundreds or even thousands of faces in a short amount of time, which is an impossible task for a human. It is more effective to filter spam emails by using machine learning to prevent significant financial damage for consumers without compromising their productivity [17]. Facial recognition or spam filtering can be known as part of the classification problem, which involves classifying images, texts, or audio. As more

applications and systems can be powered by machine learning, the world has witnessed an explosion in the number of machine-learning-driven commercial projects as well as research papers in this field in recent years.

Classification problems can be divided into two categories. The first one is *unsupervised learning* where the input data is not labeled. For example, unsupervised learning can be used for anomaly or outlier detection. The second one is *supervised learning* where the input data is fully labeled. The classification function is generated in supervised learning to extract features among known classes of given datasets. For instance, testing images of cats and dogs can be classified based on a set of labels. Traditional ML algorithms include but are not limited to Naïve Bayes, K-nearest neighbors, or decision trees that can be used to complete classification tasks [18]. However, traditional ML techniques such as Support Vector Machine are outperformed by modern machine learning algorithms including Neural Networks on large datasets [19]. Thus, Neural Networks were chosen as image classifiers in this paper.

2.2 Neural Networks History

Mark I Perceptron, introduced by Rosenblatt, received a matrix of input and then multiplied it with another matrix of weights before adding up the results. The result was then sent through a gate where it returned 1 if it was above a value and 0 otherwise [20]. The minimization of the dissimilarity between the predicted labels and the true outputs was used to adjust the weights of the perceptron. By doing so, the model extracted the pattern of the

problem. However, this model was proved to be ineffective in solving exclusive-or problems [21]. The binary output gateway in each neuron was dropped by Widrow & Hoff to disperse the weights to the adjacent or all neurons. Thus, this made the outcome of each neuron differentiable [22, 18]. To quantify the correctness of the current trained parameters, the total mean squared difference between the current outcome of the network and the true outcome could be known as the loss function of the network or how close the current output is to the true outcome. The average loss of all samples was referred as the cost function of the network. As a result, the weights of the networks could be adjusted based on the negative gradient of the cost function to minimize the difference between the predicted outcome and the true outcome [18]. Then, multiple layers of neurons were stacked between the input and output layer to create a multi-layer perceptron (MLP) by Werbos to handle non-linear problems including exclusive-or problems [21]. The gradient of the loss function regarding the individual network's weight was found to adjust the weights and bias of the networks via the method of *backpropagation*. Specifically, at a given neuron layer, the chain rule was used to find the loss gradient of the current layer based on the layer before it [23, 18]. Eventually, all the weights of a network were modified till they were optimal to extract insights from training datasets. In supervised learning, the performance of a Neural Networks in classifying unseen data was improved via training it with labeled data. The process of a Neural Networks classifying unseen data or testing data is referred to as the inference phase.

2.3 Convolutional Neural Networks

One of the problems with using MLP is that its performance is sub-optimal when the number of instances in a dataset is low compared to the size of the feature space. As a result, it requires a tremendously larger number of images to train for all variables in the feature space. Furthermore, MLP could not capture the spatial connections among attributes [24]. These issues are well mitigated by a Convolutional Neural Networks (CNN).

2.3.1 Convolution Layer

CNN extracts the spatial relationships among features via convolution layers. The extraction involves a filter or kernel, which is a matrix of weights, moving over the input image a distance or stride to conduct dot product with the input pixels. The result when the filter finishes is known as a feature map. As the convolution layer only uses one filter to extract the patterns of the data, the number of weights of CNN is significantly smaller than MLP where a unique weight is assigned for each input feature on each neuron [24, 18].

2.3.2 Pooling Layer

One of the common problems that happen during the Neural Networks training process is over-fitting. This issue refers to when the model focuses only on learning patterns and features to separate observed data rather than both observed and unobserved instances [25]. This issue can be mitigated by cutting the number of dimensions of data via pooling layers. This layer maintains the invariance of spatial transformation with lower computational

resources due to a lower number of dimensionalities of the feature map [26]. The general structure of a CNN can be found in the figure 2.1.

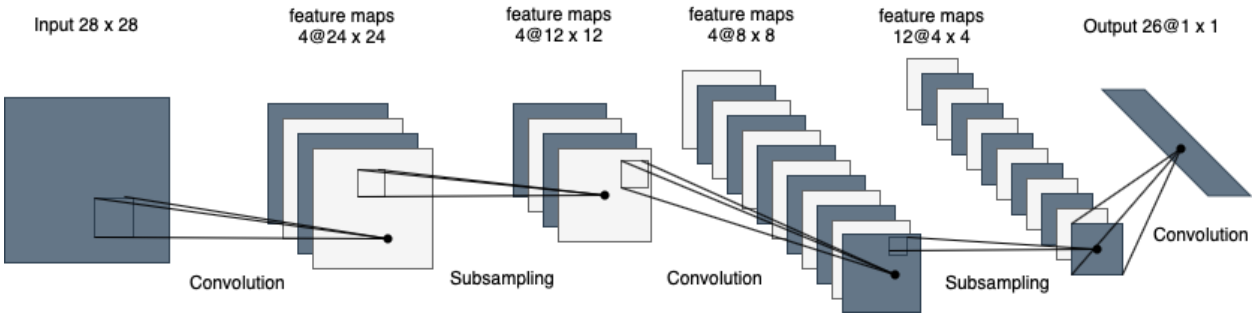

Figure 2.1: Structure of CNN for image classification

2.3.3 Transfer Learning

In some cases where the distribution of the training dataset and testing dataset are not perfectly aligned or the number of training instances is low, the performance of classifiers is drastically diminished. Transfer learning, which transfers features or insights learned from one domain to another affiliated one, is used to alleviate the data distribution dissimilarity between training and testing datasets. An example provided by Weiss et al. [27] is that sentiment classification in food reviews could be improved by information learned from digital camera reviews as both reviews share a mutual domain, which in this case was a product review. From a lower-level perspective, transfer learning also means using a set of pre-trained weights, collected from training a Neural Networks on a dataset, to train the same networks on associated datasets. Furthermore, utilizing transfer learning reduces the training time

and preserves high classification performance for learners.

2.4 Autoencoder

Neural network models are categorized into two sub-categories which are *discriminative* and *generative* models. The discriminative model classifies unseen data based on a certain set of features of observed data. For example, a discriminative learner trained on a handwritten digit training set is used to classify a testing handwritten dataset. Meanwhile, the generative model generates data that possess a set of features observed from training sets. For instance, a trained image generator generates fake images given a set of testing data.

Autoencoder (AE) is a generative network where it is comprised of an encoder and a decoder. The function $z = f(W_e, b_e; x)$ represents the encoder where x is an instance of the input dataset. The encoder is mainly used for constructing the latent space z from input data x. Meanwhile, the decoder can be written in a form of $r = g(W_d, b_d; z)$ where it generates image r from latent space z. Additionally, W_e and b_e are the weights and a bias of the encoder while W_d and b_d are the weights and a bias of the decoder [28]. Thus, the loss function of AE can be defined as:

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - r_i\|_2^2 \quad \text{where} \quad \theta = (W_e b_e; W_d b_d) \tag{2.1}$$

$\|x_i - r_i\|_2^2$ is squared Euclidean norm of $x_i - r_i$ and Euclidean norm is the distance between a vector to the space origin. The optimization problem of the AE's loss function can be found by a gradient descent algorithm. The structure of AE can be found in the figure 2.2.

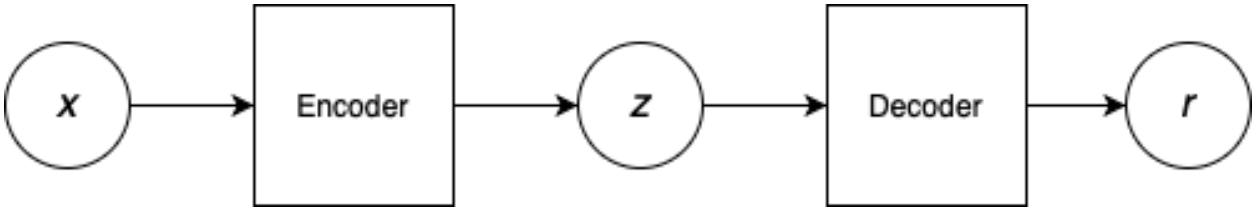

Figure 2.2: Structure of Autoencoder

2.5 Adversarial Attacks

The learner's performance on the testing dataset can be determined by how the distribution of the testing dataset aligns with the same figure of the training dataset. As a result, the performance of learners on the testing dataset can be greatly degraded if the difference, in terms of distribution of training and the testing dataset, is large. A small feature difference in a data instance may not be correctly classified, which can be because it has never been observed previously by the learner. Due to this issue, the security or robustness of a machine learning model was mentioned and shown that it can be attacked with tampered data [5, 18].

There are two types of attacks, which are *causative* and *exploratory*. Causative attack aims to perturb training examples in a way that adjusts the extracted features of the classifier which eventually causes incorrect classification of testing data. On the other hand, the exploratory (evasion) attack aims to discover the network by querying the target model with data samples. This attack attempts to explore the target network to generate data that the network has never seen before and incorrectly label it. The evasion attacks are conducted at the inference phase [29]. In this paper, the evasion attacks were chosen to

conduct experiments.

Imperceptibly modified images were found to be efficient in forcing classifiers to misclassify data in the inference phase [5]. The perturbation, which is often found imperceptible to human eyes, combined with normal examples can result in *adversarial examples*. As these examples may be found in a different data distribution, which has not been observed by the learner, they may cause misclassification.

Chapter 3
RELATED WORK

This chapter summarises the previous works in terms of adversarial defense. Additionally, a comparison regarding the experimental setup between this paper and previous works is conducted.

3.1 Adversarial Training

To improve the robustness of the Neural Networks, Madry et al. [30] suggested adversarial training where adversarial examples were used to train CNN classifiers. By training on adversarial examples, the Neural Networks were more robust against various adversarial attackers. To demonstrate the benefits of adversarial training, experiments were conducted on the Handwritten digit MNIST and CIFAR-10 datasets. Additionally, ResNet classifiers were used to classify images on the CIFAR-10 dataset. The two main adversarial attackers used in this experiment were the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD), which had full access to the target models. The adversarial training method helped boost the robustness of CNN classifiers. For example, on the CIFAR-10 dataset, the ResNet base module achieved a natural accuracy score of 92.7%. Meanwhile, the classifier only achieved robust accuracy scores of 27.5% and 0.8% against adversarial examples generated by FGSM and PGD attackers. By training on adversarial examples crafted by

the FGSM attacker, the simple ResNet classifier achieved a natural accuracy score of 87.4% and robust accuracy scores of 90.9% and 0% against FGSM and PGD attacks respectively. Additionally, the classifier, trained on adversarial examples created by the PGD attacker, accomplished a natural accuracy score of 79.4% while its robust accuracy scores against FGSM and PGD attackers were 51.7% and 43.7%. This result demonstrated that adversarial training on adversarial examples produced by PGD attackers was superior to the ones created by the FGSM. However, the natural accuracy score dropped significantly from 92.7% to 79.4% by using adversarial examples crafted by PGD attackers.

Overall, the adversarial training suggested by Madry et al. [30] boosted the robustness of the ResNet classifier on the mentioned datasets. However, the trade-off of a significant decrease in natural accuracy score could play a negative role in choosing this method as an adversarial defense. Moreover, the total increased time for utilizing adversarial training was not mentioned, which might negatively impact the decision if this method was included as an adversarial defense in a commercial ML pipeline.

3.2 Autoencoder

Kim et al. [8] contributed to the ML research community by showing that the autoencoder could be used as an adversarial defense. The autoencoder used in the study conducted by Kim et al. was comprised of convolution and pooling layers. This framework, where pretrained AE was the sole preprocessor of a CNN classifier, could be referred to as AE-C. The experiment was conducted on the Handwritten digit MNIST and CIFAR-10 dataset. The

two main attacks used as white-box attackers were FGSM and PGD where both attackers substantially reduce the classification performance. As a result, AE-C achieved higher robust accuracy than the base CNN classifier model, which could be referred to as C. However, the improvement of the robust accuracy via the use of AE was not significant. For example, according to table 1 of the paper written by Kim et al. [8], the robust accuracy scores of C and AE-C were 2.58% and 5.14% respectively when encountering PGD attacks in the white-box threat model on the CIFAR-10 dataset. Additionally, the natural accuracy scores of AE-C were barely lower than C, which were 88.32% and 89.67% respectively. The experiment was also conducted in a black-box threat model where substitute models were used for the white-box attackers, which were FGSM and PGD, to generate adversarial examples. The improvement in robust accuracy in the black-box threat model was more significant than the same figures in the white-box. For instance, ResNet-110 was used as the substitute classifier for PGD attackers on the CIFAR-10 dataset, the robust accuracy score of C was 18.28% while the same figure for AE-C was 44.91% [8]. Besides the evaluation of the benign performance of AE-C, the image reconstruction time of AE in this experiment was not mentioned, which did not provide a full understanding of the trade-offs of AE when it was used as an image denoiser.

3.3 Variational Autoencoder

3.3.1 Clean Training

Luo et al. [9] discovered that Variational Autoencoder (VAE), which is mainly used for compressing data or image denoising, is also an efficient adversarial defense for CNN. VAE was also proved to be more effective against adversarial attacks than JPEG compression via several experiments. VAE is an autoencoder where it first maps input data into a latent space via an encoder and then generates data from the latent space via a decoder. VAE differs from an Autoencoder in that it randomly samples latent vectors. Particularly, the mean and the standard deviation of the output of the last layer of the encoder are utilized to generate the latent space. Then, the random sampling of the latent space is conducted. Moreover, figure 3.1 shows the general structure of the VAE [9].

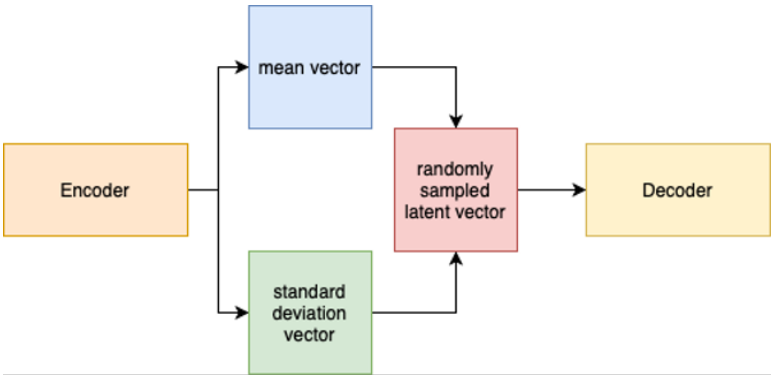

Figure 3.1: Structure of VAE

The image reconstruction loss and the loss of Kullback-Leibler-divergence (KL-divergence) are the building blocks of the loss function of VAE. Specifically, KL-divergence measures the

difference between two probability distributions; therefore, by minimizing KL-divergence loss, the chosen mean/variance pair has the largest distance from the standard Gaussian distribution. Besides KL-divergence, the reconstruction error calculation is to quantify the difference between the output and input images. Thus, by minimizing the reconstruction loss, it encourages the model to adjust its weights and biases to generate images that resemble the input validation images [9]. The loss function can be found in the following formula:

$$l_i(\theta, \phi) = -E_{z\ q_\theta(z|x_i)} log(p_\phi(x_i|z)) + \beta D_{KL}(q_\theta((z|x_i)\|(p(z))) \tag{3.1}$$

In the above formula, $q_\theta(z|x_i)$ is the function of an encoder, which generates sampled latent vector z from image x. Meanwhile, $p_\phi(x_i|z))$ is the function of a decoder generating image x from sampled latent vector z. Additionally, on $\mathbb{R}^d$, a standard Gaussian distribution can be denoted as $p(z)$. Furthermore, by adding a hyperparameter, every feature of the input data can be uniquely described by the latent vector components. This can be referred to as disentangled VAE [9].

The experiment was conducted on the Handwritten digit MNIST, CIFAR-10, and NIPS 2017 Defense Against Adversarial attacks development datasets. CNN classifiers were first trained and validated before being attacked by FGSM and Iterative-FGSM (I-FGSM) attackers. Then, the adversarial examples were denoised by VAE, which was trained on the same training and validation datasets as CNN classifiers. VAE image denoiser successfully defended CNN classifier against FGSM attackers on all datasets in the gray-box threat model. For instance, for the CIFAR-10 dataset, VAE helped the CNN classifier achieve 70% in robust accuracy score while the base CNN classifier only acquired 22% in robust accuracy score

against FGSM attacker. Meanwhile, JPEG compression, which had a quality of 50, only accomplished a robust accuracy of 68% for the same testing scenario. Additionally, the CNN classifier only achieved roughly 3% against the I-FGSM attacker on the ImageNet dataset. Meanwhile, VAE helped the CNN classifier achieve a score of 63% on the same figure [9]. Even though it was demonstrated that VAE and JPEG compression were effective against adversarial attacks in the gray-box threat model, their performance was not evaluated on benign images and their image reconstruction time was not mentioned. Therefore, their trade-offs were not evaluated.

Furthermore, VAE was used as the core component in spherical sampling-based VAE (SB-VAE), which achieved state-of-the-art defending performance in the ImageNet dataset. For instance, SB-VAE achieved a robust accuracy score of 68% against FGSM attackers, which reduced the base robust accuracy score down to around 3% on the ImageNet dataset [31]. Thus, the performance of SB-VAE and standalone VAE achieved similar results.

3.3.2 Adversarial Training

Similar to using VAE as an adversarial defense suggested by Luo et al., Li et al. [32] also suggested the use of VAE as an image denoiser to eliminate adversarial noises. The experiment was conducted on the Handwritten digit MNIST, Fashion-MNIST, CIFAR-10, and CelebA datasets. There were 4 different CNN classifiers trained on each dataset before they were all attacked by FGSM, Random-FGSM, and Carlini-Wagner (CW) attackers. The adversarial examples were then denoised by pre-trained VAE denoisers (Defense-VAE)

before it was fed back to the classifiers for classification. Unlike the training process of VAE suggested by Luo et al., Li et al. [32] suggested that adversarial examples generated by multiple configurations of adversarial attackers be used as the training images, while the original images were used as validation images. The defender helped all CNN classifiers achieve high robust accuracy scores against all selected adversarial attackers on all mentioned datasets. For instance, Defense-VAE helped a CNN classifier model accomplish a robust accuracy score of 44.86% against FGSM attackers on the CIFAR-10 dataset. Meanwhile, the same base model, which acquired a natural accuracy score of 86.52%, only achieved 2.44% as its robust accuracy score against the same attacker on the same dataset. The effectiveness of Defense-VAE was also evaluated when it was only trained on adversarial examples generated by only two attackers instead of all attackers. As a result, Defense-VAE achieved similar results as when it was trained on all mentioned attackers. Furthermore, Defense-VAE took around 9 seconds to denoise 1000 images with the support of NVIDIA Titan-Xp GPUs [32].

The defense system suggested by Li et al. [32], which did not require retraining of the CNN classifier, was shown to be effective against adversarial attackers in the gray-box threat model, where the attackers were not aware of the defense. Additionally, the image reconstruction time of Defense-VAE was collected to show it took a significantly smaller time than other defenders; however, it did not show the total increased time for the entire ML pipeline. Moreover, the training process of the Defense-VAE makes it dependent on the adversarial attackers, which may evolve in the future. Furthermore, the classification performance of benign images was not conducted when applying this defender; thus, it did

not give a full understanding of the mentioned defender's trade-offs in a general ML pipeline.

3.4 U-Net

Besides VAE, a different type of Autoencoder was suggested by Zhang et al. [10] to be an image denoiser to defend CNN classifiers against adversarial attacks. U-Net is a variant of Autoencoder generating images from a set of input images. The overall structure of U-Net used in the study conducted by Zhang et al. can be found in figure 3.2. There are two differences between the AE and U-Net structures used by Zhang et al. [10]. The first difference is that a dilated convolution layer is used instead of a regular convolution layer. The receptive field, which is the area scanned by the kernel of the convolution layer, is broadened by the dilated convolution layer by ignoring pixels during the convolution operation. The pixel jumping can be calculated as $(L-1)$ where L is the dilation rate of a given convolution layer [33]. As a result, the expansion of the receptive field helps the classifier extract more features from input images. The second difference is that the output of dilated convolution layers of the encoder is concatenated to the input of the convolution layers in the decoders. This was illustrated as function F in figure 3.2. The training process of the U-Net involved the use of adversarial examples generated from various adversarial attackers, including FGSM, BIM, CW, JSMA, and DeepFool, as training data. Meanwhile, the benign images were used as validation data. The experiment was conducted on the ImageNet dataset, which is one of the most widely used datasets in image processing. Additionally, a pre-trained Inception V3 model was used as the CNN classifier. The adversarial examples were generated against

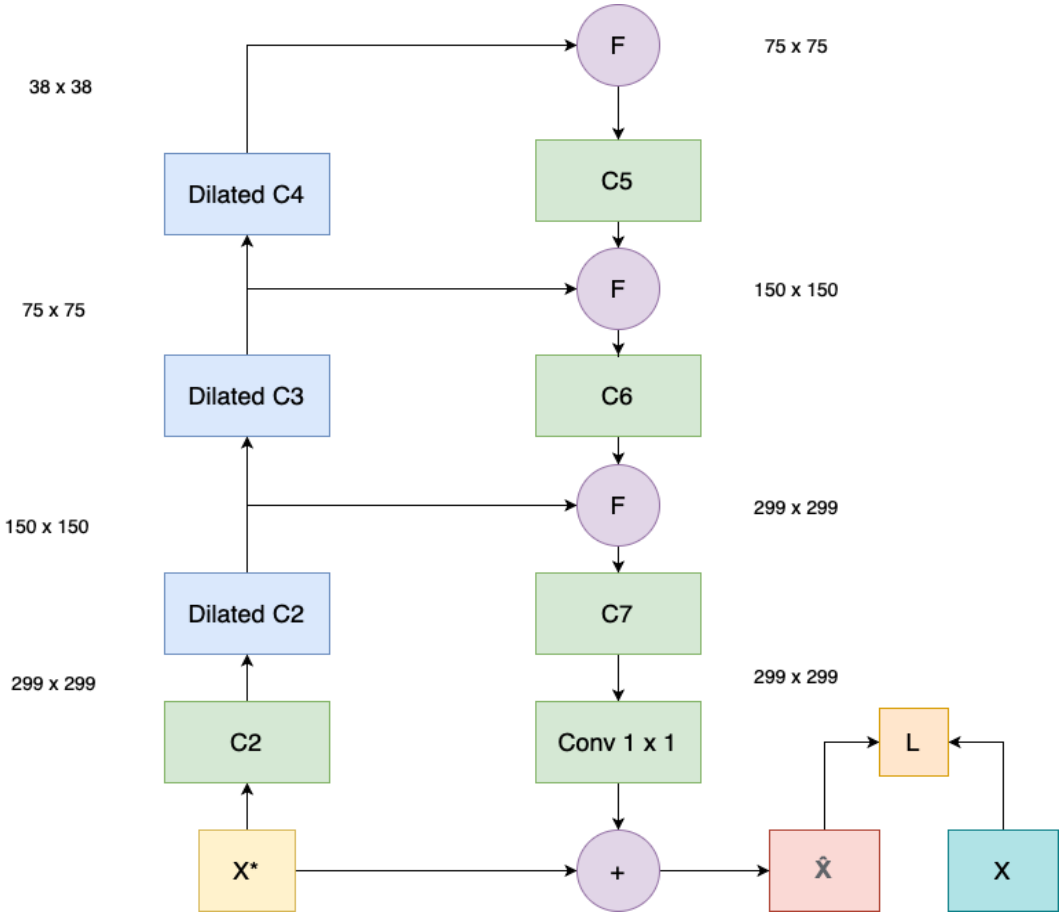

Figure 3.2: Structure of U-Net

target classifiers before they were denoised by the trained U-Net denoiser. As a result, U-Net helped the CNN classifier achieve 75.9%, 72%, 73.9%, 70%, and 77.3% of robust accuracy scores against FGSM, BIM, CW, JSMA, and DeepFool respectively on the ImageNet dataset. Moreover, the U-Net defender took 2.10 seconds to reconstruct one image. Additionally, the U-Net defender helped the classifier achieve an accuracy score of 81.6% on benign images [10].

Besides the exceptional performance against the selected adversarial attackers on the ImageNet dataset, the robustness of the base classifier was neither included nor discussed, which makes it difficult to quantify the impact of the U-Net denoiser on improving the robustness of the given ML pipeline. Moreover, the configuration of the hardware used in this experiment was not included in the paper; therefore, it is difficult to know if U-Net can be a competitive adversarial defender in commercial ML pipelines given the image reconstruction time of 2.10 seconds per image. Furthermore, as the training process of U-Net depends on adversarial examples, this makes it inflexible against future attackers.

3.5 Traditional Image Denoisers

3.5.1 Total Variance Minimization and JPEG compression

Similar to neural-networks-based image denoisers, traditional image denoisers such as JPEG compression [34], Total Variance Minimization, etc. were proved by Guo et al. [11] to be a robust defense against adversarial attacks. A random set of pixels $X(i;j;k)$ from the Bernoulli distribution is first chosen by the Total Variance Minimization (TVM). Finding the convex of the formula below is the goal of TVM and image z is the output of the TVM. Element-wise multiplication is referred to as $\odot$ [11]. TVM could be denoted in the below formulas:

$$min_z \|(1 - X) \odot (z - x) + \lambda_{TV} * TV_p(z) \tag{3.2}$$

$$TV_p(z) = \sum_{k=1}^{K} \left[\sum_{i=2}^{N} \|z(i,:,k) - z(i-1,:,k)\|_p + \sum_{j=2}^{N} \|(z(:,j,k) - z(:,j-1,k)\|_p \right] \tag{3.3}$$

Evasion attackers used in this experiment were FGSM, CW, and DeepFool. The ex-

periment was conducted in the gray-box threat model where the adversary only has full knowledge of the target model but not the defense. Therefore, the adversarial examples were generated against target classifiers and then purified by image denoisers before they were sent to the classifier for classification. Additionally, a ResNet-50 model was used to train on the ImageNet dataset in the paper written by Guo et al. [11]. The classifier's robust accuracy scores were 0% when it was attacked by the DeepFool and CW adversarial attackers. Similarly, the classifier only acquired a robust accuracy score of 23% when it was attacked by the FGSM algorithm. However, TVM helped the classifier achieve robust accuracy scores of around 63% against all three attackers. Similarly, JPEG compression helped the classifier increase robust accuracy scores to around 65% against the FGSM, DeepFool, and CW attackers respectively [11].

Overall, traditional image denoisers such as JPEG compression and TVM achieved significant improvements in robust accuracy scores for classifiers against FGSM, CW, and DeepFool attacks on the ImageNet dataset. However, the evaluation of how those denoisers affect the classification performance of benign images was not mentioned. Additionally, similar to other papers suggesting image denoisers as an adversarial defense, Guo et al. [11] did not provide the image reconstruction time of mentioned traditional image denoisers, which could be a major factor for time-sensitive commercial ML applications.

3.5.2 Re-sampling Interpolation

Similar to re-sampling methods suggested by Guo et al., Jiang et al. [35] proposed the use of interpolation algorithms such as nearest neighbor and bilinear interpolation as adversarial defenses. The nearest neighbor algorithm, which could be referred to as proximal interpolation, can be depicted in the following formula [35]:

$$f(i+u, j+v) = f(i, j) \tag{3.4}$$

In the formula of nearest neighbor interpolation, $u, v \in (-0.5, 0.5)$ and the target pixels require re-sampled are $f(i+u, j+v)$.

Similarly, the associated values of four directly adjacent neighbor pixels were used in the bilinear interpolation method. The formula of this bilinear interpolation algorithm can be found in the formula below [35]:

$$\begin{aligned} f(i+u, j+v) = (1-u)(1-v)f(i, j) + (1-u)vf(i, j+1) \\ +u(1-v)f(i+1, j) + uvf(i+1, j+1) \end{aligned} \tag{3.5}$$

Similarly, $u, v \in [0, 1]$ and the re-sampled pixels are $f(i+u, j+v)$. Furthermore, Jiang et al. [35] used 1105 images of people with face coverings from the ImageNet dataset for evaluation. The experiment was conducted and evaluated under the gray-box threat model. A pre-trained ResNet50 model was the main target of the attacks. Specifically, ResNet50 achieved a natural accuracy score of 86.33% while the robust accuracy score was 0% under the attack of FGSM or DeepFool. Proximal and bilinear interpolations helped the classifier achieve 68% and 75% respectively in robust accuracy scores against FGSM attacks. Similarly, the

ResNet50 classifier achieved 71% and 77% of robust accuracy scores against DeepFool attacks with the help of proximal and bilinear interpolations respectively. Additionally, the benign accuracy scores of proximal and bilinear interpolations were 81% and 86% respectively.

Overall, both interpolation methods helped ResNet50 achieve high robustness against several adversarial attacks in the gray-box threat model without significantly reducing the benign classification performance. However, as the sample size was 1105 images of a specific class, the effect of both interpolations on ResNet50's robustness may not be seen in other classes, datasets, or classifiers.

3.6 Comparison

Table 3.1 includes the comparison of the related works and this paper. As can be seen in table 3.1, this paper experimented with both the gray-box and white-box threat models to have a better understanding of the performance of image denoisers as adversarial defenses. Exploring drops of benign classification performance and time trade-offs of several image denoisers were also part of the objectives of this paper.

Table 3.1: Comparison among related works in terms of experimental setup

Method	Gray-box threat model	White-box threat model	Performance on benign examples	Time trade-off
Adversarial training	N/A	✓	✓	-
VAE with normal training	✓	-	-	-
VAE with adversarial training	✓	-	-	✓
JPEG and TVM	✓	-	-	-
AE	-	✓	✓	-
U-Net	✓	-	✓	✓
Re-sampling Interpolation	✓	-	✓	-
Ours	✓	✓	✓	✓

Chapter 4
METHODOLOGY

In this section, the experimental design including the proposed method, software version, choices of datasets, attackers, defense models, threat models, and classification performance metric are articulated.

4.1 Software Version

In this section, versions of major software used in this experiment are discussed. The container reader software used in this paper is Singularity [36]. The version of the utilized container was NVIDIA NGC TensorFlow 21.12-tf2-py3. The attackers were loaded via ART 1.9.1 [37].

4.2 Proposed Method: Emulated Autoencoder

As resizing methods may introduce minor distortion into images, this phenomenon can be utilized to distort adversarial noises potentially embedded in testing images. Specifically, images are first resized to a smaller size, which introduces some distortion as a side effect during this process. Next, the same batch of images is resized back into the original size, which introduces another small batch of modifications to the input images. This method can be referred to as Emulated Autoencoder (EAE). As the Autoencoder deflates the size of

images in the encoder and inflates the image size in the decoder, which is similar to the proposed method, hence the name Emulated Autoencoder was chosen. Images can go through this resizing process as many times as possible; however, excessive distortion introduced to images can cause the target classifier to misclassify benign as well as adversarially perturbed images. As a result, the resizing process was conducted once for each image to preserve the original features and distort potential adversarial perturbations. Figure 4.1 demonstrates an example of EAE. As a result, the EAE method reduced minor details of the original image.

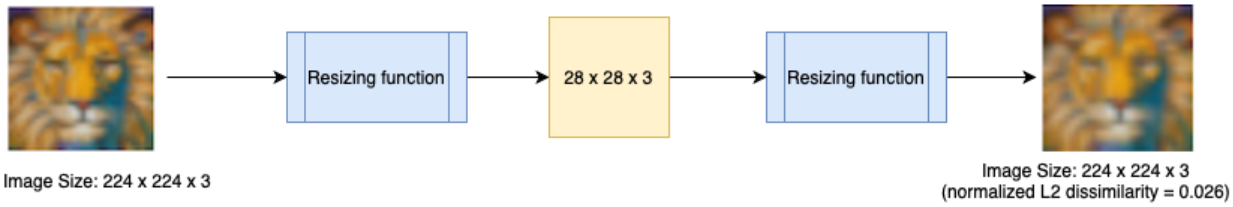

Figure 4.1: Resized CIFAR-100 example

4.2.1 Resizing Methods

Figure 4.2: CIFAR-10 32 x 3 x 3 original

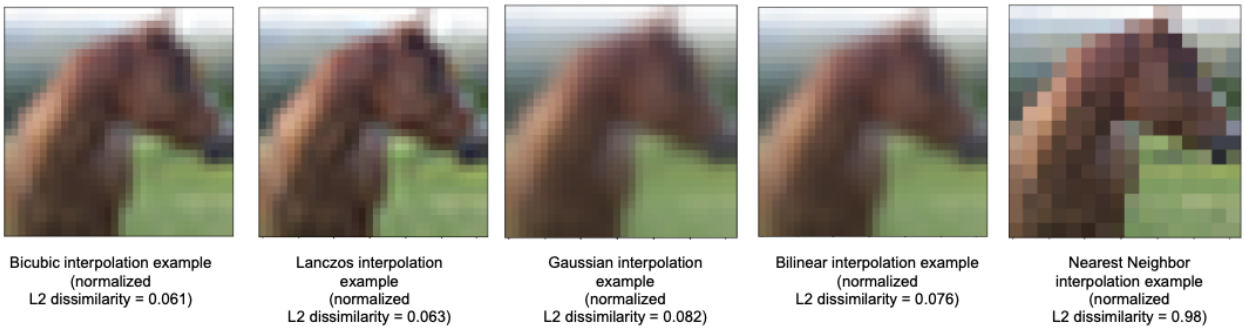

Figure 4.3: 32 x 32 x 3 interpolation examples of the CIFAR-10 dataset

To resize images, there were several resizing methods including lanczos [38], nearest neighbor, gaussian [39], bilinear, and bicubic interpolation [40]. Normalized L2 dissimilarity and the reconstruction time were used to select the most efficient resizing method. Normalized L2 dissimilarity can be found via formula [11]:

$$\frac{\|x_n - x'_n\|_2}{\|x_n\|_2} \tag{4.1}$$

Specifically, x_n and x'_n are original and resized images respectively in the normalized L2 dissimilarity formula. Additionally, an example of CIFAR-10 32 x 32 x 3 images can be found in figure 4.2. To amplify the amount of information changed during the interpolation process, the image size was first reduced by half to 16 x 16 x 3 before it was restored back to the original one in figure 4.3. This process was similar to the EAE. In table 4.1, the reconstruction time of each interpolation method was measured when images were resized from 32 x 32 x 3 to 224 x 224 x 3, which was required by transfer learning.

As shown in table 4.1, the bilinear interpolation was faster than other interpolation methods, except for the nearest neighbor. An interpolation method creating images with a low

Table 4.1: Image reconstruction time of resizing methods

Interpolation Method	Reconstruction time per 1000 images (s)
Bilinear	0.0085
Nearest Neighbor	0.0084
Lanczos	1.33
Gaussian	1.03
Bicubic	0.21

normalized L2 dissimilarity value was preferred to prevent extensive image distortion which could make classifiers label benign images incorrectly. As can be seen in figure 4.3, interpolated example created by bilinear interpolation had a lower normalized L2 dissimilarity value than nearest neighbor method. Thus, the bilinear interpolation method was the main resizing method in this study. Due to its simplicity, EAE theoretically does not consume significant computational resources. Thus, it was later chained with several image denoisers in an attempt to further improve the overall robustness.

4.2.2 Linear Interpolation

To understand the bilinear interpolation, an understanding of linear interpolation needs to be established, which is at the core of the EAE method. In linear interpolation, a weighted average of two associated points is used to interpolate a target point in a one-dimension

array [41]. The problem that linear interpolation attempts to resolve can be depicted in the figure 4.4.

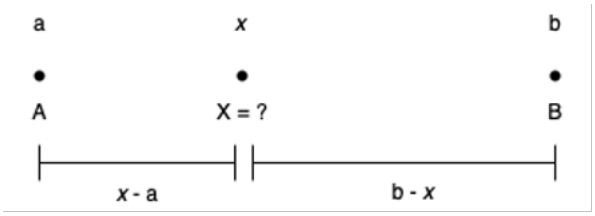

Figure 4.4: Linear interpolation

Furthermore, the linear interpolation algorithm can be found via the following formula [41]:

$$X = A(1 - w) + Bw \tag{4.2}$$

In the formula of linear interpolation, $w = \frac{x-a}{b-x}$ is the weighted average in this case. Furthermore, A and B are the associated values of point a and b respectively while X is the interpolated value of target point x. Moreover, a and b can be found via the following formula [41]:

$$a = (\lfloor ratio * x \rfloor), \quad b = (\lceil ratio * x \rceil) \quad where \;\; ratio = \frac{inputSize - 1}{targetSize - 1} \tag{4.3}$$

4.2.3 Bilinear Interpolation

Bilinear interpolation is comprised of applying linear interpolation at the horizontal and the vertical axes. Associated points and target points can be depicted in the figure 4.5 [41].

Firstly, linear interpolation is used to interpolate the values on the horizontal axis, which

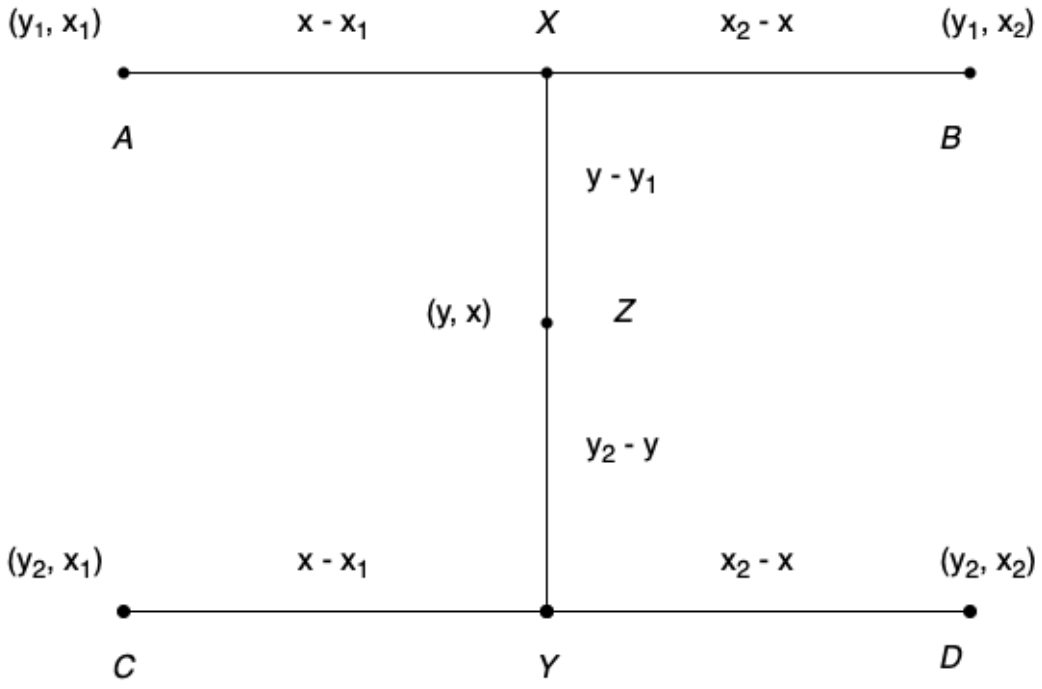

Figure 4.5: Bilinear Interpolation

can be depicted in the following formula [41]:

$$X = A(1 - w_x) + Bw_x, \quad Y = C(1 - w_x) + Dw_x \tag{4.4}$$

According to the formula above, X and Y are the interpolated value on the horizontal axis of the target point (x, y) that has a value Z. Finally, the value Z of point (x, y) can be found in the following formula [41]:

$$Z = X(1 - w_y) + Yw_y = A(1 - w_x)(1 - w_y) + Bw_x(1 - w_y) + C(1 - w_x)w_y + Dw_xw_y \tag{4.5}$$

Where $w_x = \frac{x - x_1}{x_2 - x_1}$, $\quad w_y = \frac{y - y_1}{y_2 - y_1}$ are the weighted average used in this method.

Similar to linear interpolation, associated point A, B, C, and D can be found via the

following formula given that $f(x, y)$ is the function returns the associated value of point (x, y) [41]:

$$\alpha = \frac{currentWidth - 1}{targetWidth - 1} \quad where \quad (targetWidth - 1) > 1 \tag{4.6}$$

$$\beta = \frac{currentHeight - 1}{targetHeight - 1} \quad where \quad (targetHeight - 1) > 1 \tag{4.7}$$

$$A = f(\lfloor \alpha x \rfloor, \lfloor \beta y \rfloor), \quad B = f(\lceil \alpha x \rceil, \lfloor \beta y \rfloor) \tag{4.8}$$

$$C = f(\lfloor \alpha x \rfloor, \lceil \beta y \rceil), \quad D = f(\lceil \alpha x \rceil, \lceil \beta y \rceil) \tag{4.9}$$

Unlike the implementation of bilinear interpolation proposed by Jiang et al. [35], EAE is conceptually more effective at negating the effect of adversarial perturbation. In the implementation, both methods use the values of the image pixels in calculations of bilinear interpolation. However, the number of pixels remains unchanged after applying bilinear interpolation in the method proposed in [35]. The effectiveness of the method proposed by Jiang et al. [35] against adversarial examples could be due to modifying adversarial pixels. Meanwhile, the number of pixels changes during the EAE process. Specifically, the number of pixels first drops when images are down-sampled and then it is restored when they are up-sampled. Therefore, it physically skips or removes some of the potential adversarial perturbation during the down-sampling process. EAE method's effectiveness relies on both modifying and removing adversarial pixels in both up and down-sampling.

4.3 Experimental Setup

In this experimental setup, the CNN classifiers were first trained on a given dataset without defenses or attacks to establish a base performance of benign classification. Unlike adver-

sarial training, image denoiser does not require classifiers to retrain in order to increase the overall robustness. Image denoising also does not depend on evasion attacks to improve the robustness of classifiers; therefore, the effectiveness of denoiser is not affected by future attacks. Thus, image denoising was the main defense mechanism in this paper. Similar to classifiers, neural-networks-based image denoisers were trained before being applied as adversarial defenses. Various threat models, including the gray-box and white-box, were then applied. Each adversarial defense was benchmarked for its performance on benign and adversarial examples. Training and image reconstruction times were recorded to evaluate the speed of each applicable image denoiser.

4.3.1 Datasets

The CIFAR-10 and CIFAR-100 datasets were utilized as they are widely used for image classification evaluation. Each CIFAR dataset contains 60,000 of RGB images of size 32 x 32 x 3 including 50,000 images for training and 10,000 images for testing. Furthermore, all classes of both datasets were equally distributed. The difference between the two datasets is that the CIFAR-10 dataset has 10 classes while the CIFAR-100 dataset has 100 fine-grained classes and 20 coarse-grained or superclasses. For the CIFAR-100 dataset, 20 superclasses were chosen as all classifiers yielded very low benign classification performance when 100 fine-grained classes were used. Moreover, figure 4.6 shows examples of the CIFAR-100 dataset [42].

Similar to the two CIFAR datasets, the German Traffic Sign Recognition Benchmark

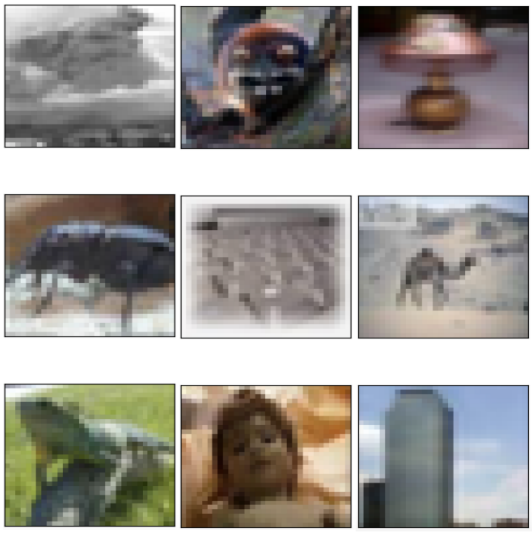

Figure 4.6: Examples of CIFAR-100 dataset

(GTSRB) dataset was selected in this experiment as it is one of the most widely used datasets for traffic sign classification tasks. The GTSRB dataset contains 51,839 RGB images of various sizes of 43 unique classes. Specifically, there are 39,209 images for training and 12,630 images for testing. All classes of the GTSRB dataset are not equally distributed in either the training or testing set. Furthermore, figure 4.7 shows examples of the GTSRB dataset [43].

4.3.2 Classifiers

4.3.2.1 Architectures

For the architecture of CNN classifiers, EfficientNet-B0, ResNet-50, and VGG-19 were used in this experiment. EfficientNet, comprised of Inverted Linear Bottleneck Convolution lay-

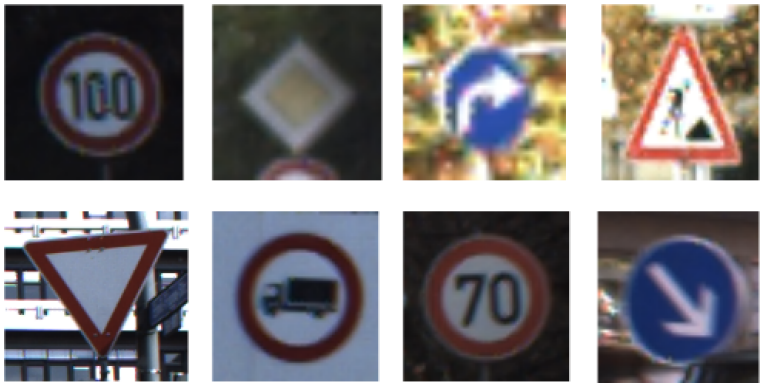

Figure 4.7: Examples of GTSRB dataset

ers, achieved the cutting-edge classification performance on the ImageNet dataset at a high inference speed in recent years [4]. Additionally, EfficientNet-B0, the base model of the EfficientNet architecture, was used in this experiment due to its high classification performance at the lowest computational cost. Meanwhile, ResNet and VGG structures achieved state-of-the-art performance on the CIFAR-10 and ImageNet datasets in the prior years. Specifically, the residual block, which concatenates the input of a layer to the output of the next layer, is the main component of the ResNet architecture [44]. As a result, ResNet-50, one of the base models of the ResNet architecture, was chosen for this experiment due to its high classification performance at a low computational cost. Similar to ResNet, VGG architecture uses a smaller receptive field and has deeper layers to further increase the classification performance [45]. Therefore, VGG-19, one of the most effective versions of VGG architecture, was chosen due to the similar properties that both ResNet and EfficientNet architecture possess.

4.3.2.2 Training

Transfer learning was used to help achieve high classification performance and reduce the training time for all classifier architectures. As the pre-trained models were trained on the 224 x 224 x 3 ImageNet dataset, training images of three selected datasets were resized to 224 x 224 x 3 to match the ones in the ImageNet dataset. Hyperparameter tunings combined with the cross-validation were employed during the training process to create the best model for each CNN architecture. Specifically, the hyperparameter set with the lowest average validation loss score was recorded and used to train models of each CNN architecture. As classification performance may vary among models of the same architecture, five models of each architecture were saved to mitigate the potential issue. The average classification performance was then recorded and reported in this study. Hyperparameter selections for training each architecture can be found in the table A.1, A.2, and A.3.

4.3.3 Adversarial Attacks

For evasion attack algorithms, Fast Gradient Sign Method (FGSM), Projected Gradient Descent (PGD), and DeepFool were selected. Furthermore, untargeted attacks were chosen over the targeted ones as the goal of the attackers in this experiment was to make CNN classifiers misclassify regardless of the classes.

4.3.3.1 Fast Gradient Sign Method

One of the evasion attacks used in this paper was the Fast Gradient Sign Method (FGSM). FGSM is one of the simplest adversarial attacks, which was introduced by Goodfellow et al. [5], and can trick classifiers with small perturbations introduced to input images. The attacking method can be depicted in the following formula:

$$\eta = \epsilon * sign(\nabla x J(\theta, x, y))) \tag{4.10}$$

For the above formula, θ, x, and y indicate the classifier's parameter, input sample, and the label respectively. Thus, $\nabla x J(\theta, x, y)$ is the gradient of the cost function of the classifier given input images. Additionally, ϵ is the minimum scalar controlling the size of the adversarial perturbation [5]. FGSM was chosen as it crafts minimal adversarial perturbations at a significantly fast pace.

4.3.3.2 DeepFool

The DeepFool attack, which was proposed by Moosavi-Dezfooli et al. [46], was chosen to achieve a higher attacking performance than FGSM. From a high-level perspective, the combination of the classifier's loss gradient and the distances among hyperplanes are exploited under the DeepFool attacks to generate adversarial examples. The attacker computes adversarial perturbation by first projecting input image x to the closest hyperplane. Then a minimum distance is used to modify the current examples further to create minimal perturbations to trick the target networks. The following formula can be used to find the closest

hyperplane [46]:

$$\hat{l}(x_0) = argmin \frac{|f_k(x) - f_{\hat{k}(x_0)}(x_0)|}{\|w_k - w_{\hat{k}(x_0)}\|_2} \tag{4.11}$$

The function $f(.)$ returns the class labels given the benign image x_0. Meanwhile, the gradients and the class with the high probability except the correct label can be depicted as w and k. As a result, the minimal perturbations can be found by via the formula [46]:

$$r_*(x_0) = \frac{|f_{\hat{l}(x_0)} - f_{\hat{k}(x_0)}|}{\|w_{\hat{l}(x_0)} - w_{\hat{k}(x_0)}\|_2^2}(w_{\hat{l}(x_0)} - w_{\hat{k}(x_0)}) \tag{4.12}$$

4.3.3.3 Projected Gradient Descent

Unlike the one-step FGSM adversarial attack, Madry et al. [30] suggested Projected Gradient Descent (PGD), which is an iterative attack, to achieve higher attacking performance with minimal perturbations. PGD, which achieved a high success rate with imperceptible perturbation, was a stronger adversarial attack than FGSM against base classifiers [30]. The creation of PGD's adversarial perturbation can be found in the formula below:

$$x^{t+1} = \prod_{x+S}(x^t + \alpha * sign(\nabla x J(\theta, x, y))) \tag{4.13}$$

In the PGD formula, x^{t+1} and x^t are current and previous adversarial examples respectively for a given input x while S are adversarial perturbations. Moreover, α is the step size of the iteration.

4.3.4 Other Adversarial Defenses

The main method used as the adversarial defense was image denoising, which does not require the target classifiers to retrain or modify their parameters. There are two types of image denoising methods. The first one is neural-networks-based image denoising which utilizes generative CNN models. Moreover, the traditional image denoising techniques, which do not rely on Neural Networks for the image denoising process, were the second type of image denoising method used in this thesis.

4.3.4.1 Neural-Network-based Image Denoiser

As previously mentioned in the Related work chapter, neural-networks-based image denoisers such as Autoencoder (AE), U-Net, and Variational Autoencoder (VAE) drew attention from the ML research community due to their ability to defend classifiers. As a result, AE, U-Net, and VAE were used as neural-networks-based image denoiser in this paper. All neural-networks-based image denoisers in this experiment utilized convolution and pooling layers. The training images were benign images added with small random noises while the validation images were benign. By doing so, these neural-networks-based image denoisers learned how to construct the latent space ultimately used to generate clean images from noisy images. Adversarial examples were not chosen as training images as this paper attempts to avoid attacks-dependent adversarial defense, which can be potentially compromised by new adaptive attacks in the future. Similar to CNN classifiers, hyperparameter tunings were conducted. The average macro F1 score of both benign and adversarial examples was

chosen to select the best hyperparameter set for neural-networks-based image denoisers. The adversarial examples were generated in the gray-box threat model during the hyperparameter tunings of these neural-networks-based image denoisers. The number of saved models for each architecture was one in this setup. As the variation in terms of the inference of Neural Networks has been mitigated by saving multiple models of CNN classifiers. The selected values of hyperparameters of AE, U-Net, and VAE can be found in tables: A.14, A.15, and A.16.

4.3.4.2 Traditional Image Denoisers

Unlike neural-networks-based image denoisers, non-neural-networks-based image denoising techniques or traditional image denoisers require no training overhead and can be utilized as adversarial defenses. As a result, the traditional image denoising techniques, including self-supervision blind denoising (Noise2self) [47], JPEG compression [48], Non-local mean [49], and Total Variance Minimization (TVM) [11] were selected for this paper's experiment. Specifically, JPEG compression and TVM were proved to be effective against evasion attacks in the gray-box threat model in Guo et al.'s study [11]. Additionally, Noise2self, which fine-tunes a given image denoising function, was evaluated in defending CNN classifiers against adversarial attacks. Non-local mean, which is one of the first image denoising methods, was also benchmarked as an adversarial defense. The configuration of JPEG compression, Noise2self, Non-local mean, and TVM can be found in tables: A.17, A.20, A.19, and A.18.

Basic bilinear interpolation proposed by Jiang et al. was not selected in this thesis due to

hardware limitations. Basic bilinear interpolation was also not chosen because the proposed EAE method is conceptually more effective due to the ability to skip and modify adversarial noises. Meanwhile, basic bilinear interpolation can only negate adversarial perturbations via modifications.

4.4 Threat models

The experiment was conducted in two different threat models which are gray-box and white-box. In each threat model, the assumptions of what attackers had access to and could do were different.

4.4.1 Gray-Box Threat Model

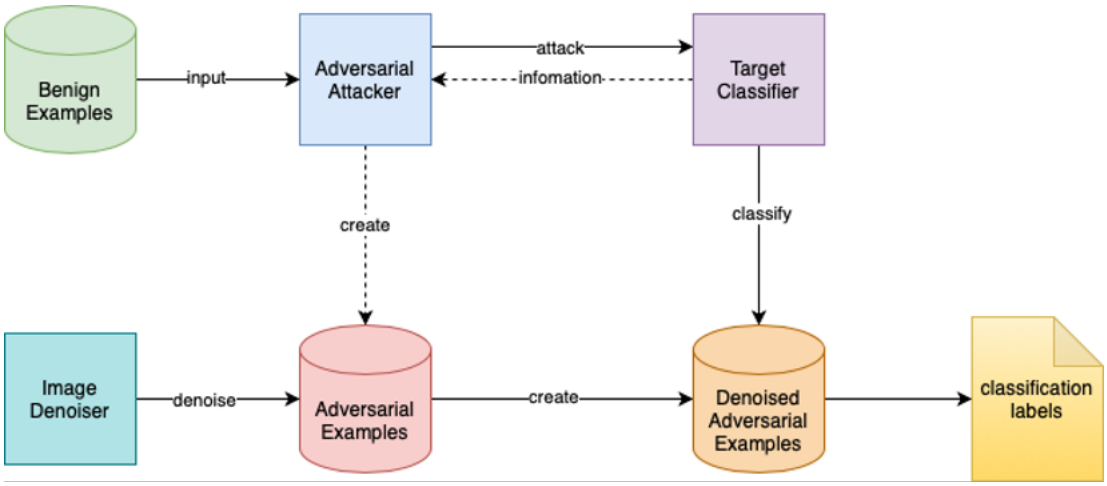

Figure 4.8: Gray-Box Threat Model

In the gray-box setting, the attacker has full access to the target classifiers; therefore, the loss gradient of the classifiers along with the dataset can be used to generate the adversarial

examples. However, the attacker does not have any knowledge of the defense mechanism. Therefore, the adversarial examples generated based on the base target classifier are then purified by an image denoiser. As a result, the denoised images are fed back to the target classifiers for classification. The data flow of this setting can be found in figure 4.8. For the data flow of this threat model, the attacker first obtains all parameters of the target classifier and the testing images. Next the adversarial attacker crafts adversarial examples based on the obtained information of the target classifier and acquired examples of the dataset. Finally, the adversarial examples are denoised by an image denoiser before the purified examples are fed to the target classifiers for classification labels.

4.4.2 White-Box Threat Model

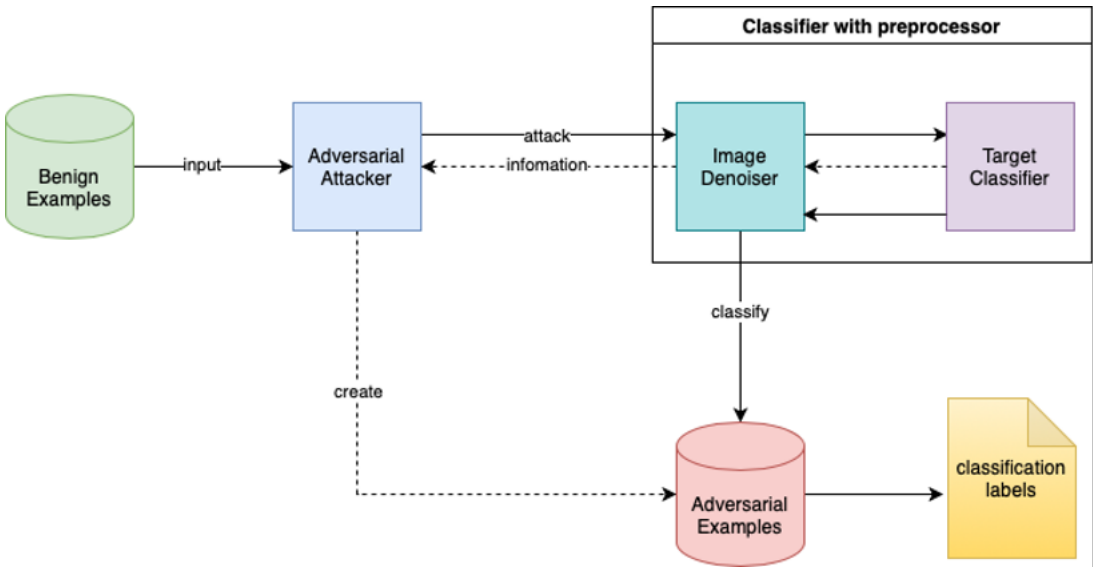

Figure 4.9: White-Box threat model

Similar to the gray-box setting, the attacker in the white-box setting has full access to the target classifier's parameters as well as the entire dataset. However, the attacker is aware of the defense mechanism in this case, which can be used to generate more targeted adversarial examples. Therefore, adversarial attackers in this setting are expected to generate more powerful adversarial examples which further force the target classifier to misclassify adversarial examples. The data flow of this threat model can be depicted in figure 4.9. For the data flow of this threat model, the information about the target classifier, including its parameters, and the image denoiser are obtained. Next, the attacker generates adversarial examples based on the acquired testing samples and knowledge of the target classifier along with its defense. Finally, the adversarial examples are fed to the combination of the image denoiser and the target classifier for classification labels.

According to Athalye et al. [50], when $g(x) \approx x$ is true where $g(.)$ is a preprocessor and x is a benign image, the Straight-Through Estimator using an identity function was able to compromise the defense strategy at a lower cost than Backward Pass Differentiable Approximation. Thus, the Straight-Through Estimator was used in this experiment. For each input image, the loss gradient of the combined denoiser-classifier was approximated to generate adversarial examples after the denoising process was applied [50].

4.5 Classification Performance Metric

As this study had no emphasis on either false negatives or false positives, the performance classification metric was a harmony of precision and recall score. Thus, the F1 score was

used in this thesis to quantify the classification performance of CNN classifiers when they were coupled with or without image denoisers. The formula for precision, recall, and F1 score can be found in the following formulas:

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{4.14}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{4.15}$$

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4.16}$$

Additionally, the macro-average F1 score, which calculates the F1 score of each class before taking the average of those, was used in this study. The macro-average F1 score weighs all classes equally by averaging all F1 scores of all classes in case a dataset has an unevenly distributed number of classes, which happens to the GTSRB dataset.

Chapter 5

RESULTS

In this chapter, the trade-offs of image denoisers, which include benign classification performance, training, and image reconstruction time, are analyzed. Furthermore, the robust classification performance of image denoisers in both of the threat models are evaluated. Moreover, the hyperparameter selections of Emulated Autoencoder in three datasets are mentioned. Additionally, attacker parameters can be found in the table A.4, which includes the limit of the adversarial perturbation added to each pixel, for reproducibility purposes. The practical implications of these results are discussed in the chapter 6.

5.1 Trade-offs: Training and Image Reconstruction Time

5.1.1 Training Time of Neural-Networks-Based Image Denoisers

The AE, U-Net, and VAE are neural-networks-based image denoisers that require training to extract features of a given dataset. Thus, the training time of these denoisers is considered a cost or a trade-off to the overall ML system. According to table 5.1, all neural-networks-based image denoisers took hours to finish training to extract insights of datasets. On the other hand, similar to traditional image denoisers, EAE does not require training to be deployed to defend classifiers. As a result, the overall training time of the ML system remains unchanged

Table 5.1: Training time in **hours** of neural-networks-based image denoisers on selected datasets

	CIFAR-100	CIFAR-10	GTSRB	Average training time (h)
AE	6.7	5.2	12.4	8.1
U-Net	7.8	4.3	12.8	8.3
VAE	21.7	20.3	24.3	22.1
EAE (ours)	**None**	**None**	**None**	**None**

when EAE is utilized as an adversarial defense.

5.1.2 Image Reconstruction Time

The inference speed of classifiers and the image reconstruction time of image denoisers were discussed in this section. Additionally, as there was no official GPU support for traditional image denoisers, the image reconstruction time of image denoisers and inference speed of classifiers were benchmarked under both CPU and GPU.

According to table 5.2, under GPU support, most of the neural-networks-based image denoisers took a similar amount of time to reconstruct images as classifiers took to classify 1000 images, which was at least 3 seconds. However, the Emulated Autoencoder took significantly less amount of time, which was 0.023 seconds under GPU, to reconstruct the same number of images. Thus, the total time of using neural-networks-based image denoisers to

Table 5.2: Image reconstruction time of image denoisers and inference speed of classifiers per 1000 images

Function type	Function name	Time under CPU (s)	Time under GPU (s)
Classifier	EfficientNet-B0	39.3	4.43
	ResNet-50	91.5	3.72
	VGG-19	313.4	3.02
Denoiser	**EAE (ours)**	**0.449**	**0.023**
	AE	108.8	3.23
	U-Net	471.4	7.7
	VAE	8388.27	3.02
	JPEG compression	2.25	N/A
	TVM	328.7	N/A
	Non-local mean	433.2	N/A
	Noise2self	99.4	N/A

defend classifiers was roughly doubled. Meanwhile, the use of EAE as an adversarial defense barely increased the overall time of the system.

Similarly, under CPU support, neural-networks-based image denoisers generally required more time to reconstruct 1000 images than classifiers required to classify the same number of images. For example, roughly 471 seconds were required by the U-Net denoiser to reconstruct 1000 images while EfficientNet-B0 only took around 40 seconds to classify 1000 images. Noticeably, JPEG compression took around 2 seconds to reconstruct 1000 images, while other traditional image denoisers took at least 100 seconds to reconstruct the same amount of examples. On the other hand, EAE only required around half a second to reconstruct 1000 images, which makes EAE significantly faster than any other image denoisers. As a result, the total time of the system barely increased with the help of the EAE denoiser as an adversarial defense under CPU.

As JPEG compression required lower image reconstruction time than other traditional image denoisers do under CPU, it was chained with EAE to create JPEG-EAE and EAE-JPEG chained denoisers. For JPEG-EAE chained denoiser, images denoised by JPEG compression are then denoised by EAE. Similarly, EAE-JPEG denoiser has the images denoised by EAE before they are then denoised by JPEG compression. Similar to JPEG compression, EAE was also chained with VAE, which took the least amount of time out of all three neural-networks-based image denoisers under GPU, to create EAE-VAE and VAE-EAE chained image denoisers.

5.2 *Trade-off: Benign Classification Performance*

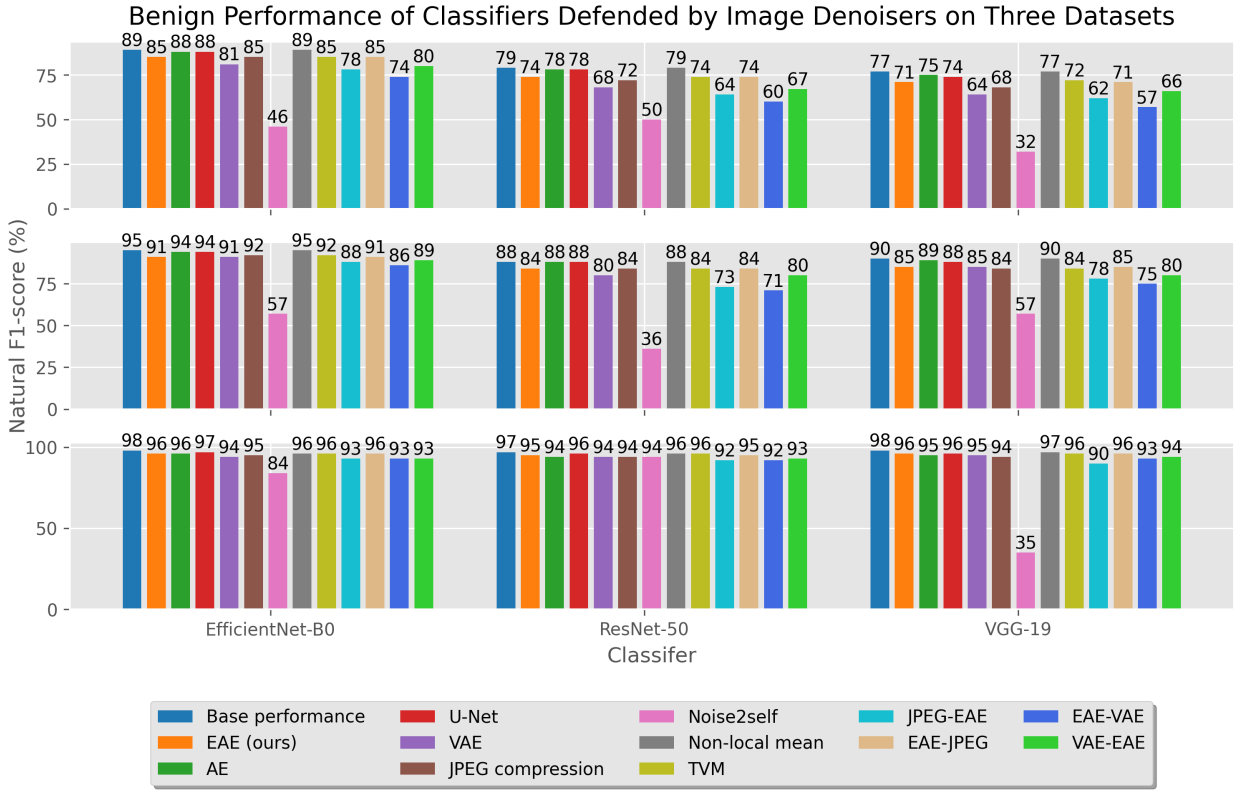

Figure 5.1: Benign classification performance of classifiers under various image denoisers on three datasets. **The top, middle, and bottom subplots are for the CIFAR-100, CIFAR-10, and GTSRB datasets respectively.**

Based on figure 5.1, AE, U-Net, or Non-local mean barely reduced the natural F1 scores of all three classifiers on all three datasets. Meanwhile, EAE, JPEG compression, and TVM reduced about 4% of the natural F1 scores of three classifiers on the CIFAR-100 and CIFAR-10 datasets. On the other hand, VAE dropped around 10% and 6% of natural F1 scores of all

classifiers on the CIFAR-100 and CIFAR-10 datasets respectively. Moreover, VAE-EAE and VAE had the same effect on the benign classification performance of all three classifiers on all three datasets. However, EAE-VAE saw further drops of at least 5% in benign classification performance compared to VAE in both CIFAR-100 and CIFAR-10 datasets. Furthermore, on all three datasets, EAE-JPEG saw similar drops in natural F1 scores of all three classifiers as EAE did; meanwhile, JPEG-EAE caused reductions of roughly 10% in natural F1 scores of all classifiers. Lastly, Noise2self, which is also the worst image denoiser in this evaluation criteria, plummeted at least 20%, 35%, and 10% of the natural F1 scores of all classifiers on the CIFAR-100, CIFAR-10, and GTSRB dataset respectively. Unlike the CIFAR-100 and CIFAR-10 datasets, all image denoisers, except for Noise2self, JPEG-EAE, EAE-VAE, and VAE-EAE, did not cause any drops in benign classification performance of all classifiers on the GTSRB dataset. Particularly, JPEG-EAE, EAE-VAE, and VAE-EAE chained denoiser caused a roughly 5% drop in natural F1 scores of all three classifiers on the GTSRB dataset. Overall, there were no significant trade-offs in utilizing AE, U-Net, EAE, EAE-JPEG chained denoisers, and most traditional denoisers as an adversarial defense on all three datasets.

5.3 Adversarial Examples

The adversarial examples crafted by adversarial attackers on the CIFAR-100 dataset can be found in figure 5.2. The adversarial examples were collected when the base EfficientNet-B0 classifier was the target one. As a result, the adversarial examples crafted by the DeepFool attacker were slightly less imperceptible than the ones made by FGSM and PGD attackers.

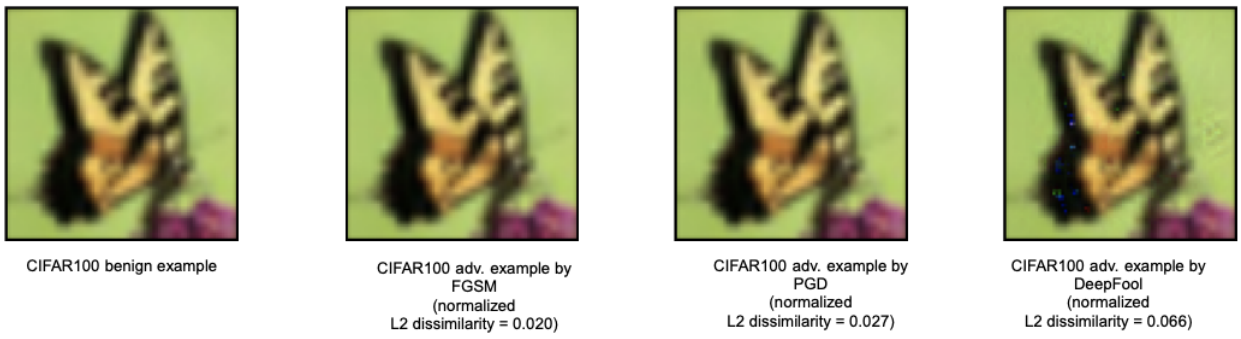

Figure 5.2: Adversarial examples of CIFAR-100

5.4 Gray-Box Threat Model Results

The performance of all image denoisers in the gray-box threat model are evaluated in this section.

5.4.1 CIFAR-100 Dataset

5.4.1.1 EAE Intermediary Size Selection

The EAE method requires users to select an intermediary size as the sole hyperparameter of this adversarial defense. To select an optimal intermediary size, both natural and robust F1 scores of various attack algorithms in the gray-box threat model were considered. According to figure 5.3, an intermediary size of 28 was selected as it helped all three classifiers achieve high F1 scores. The data illustrated in the figure 5.3 can be found in tables: A.5, A.6, and A.7.

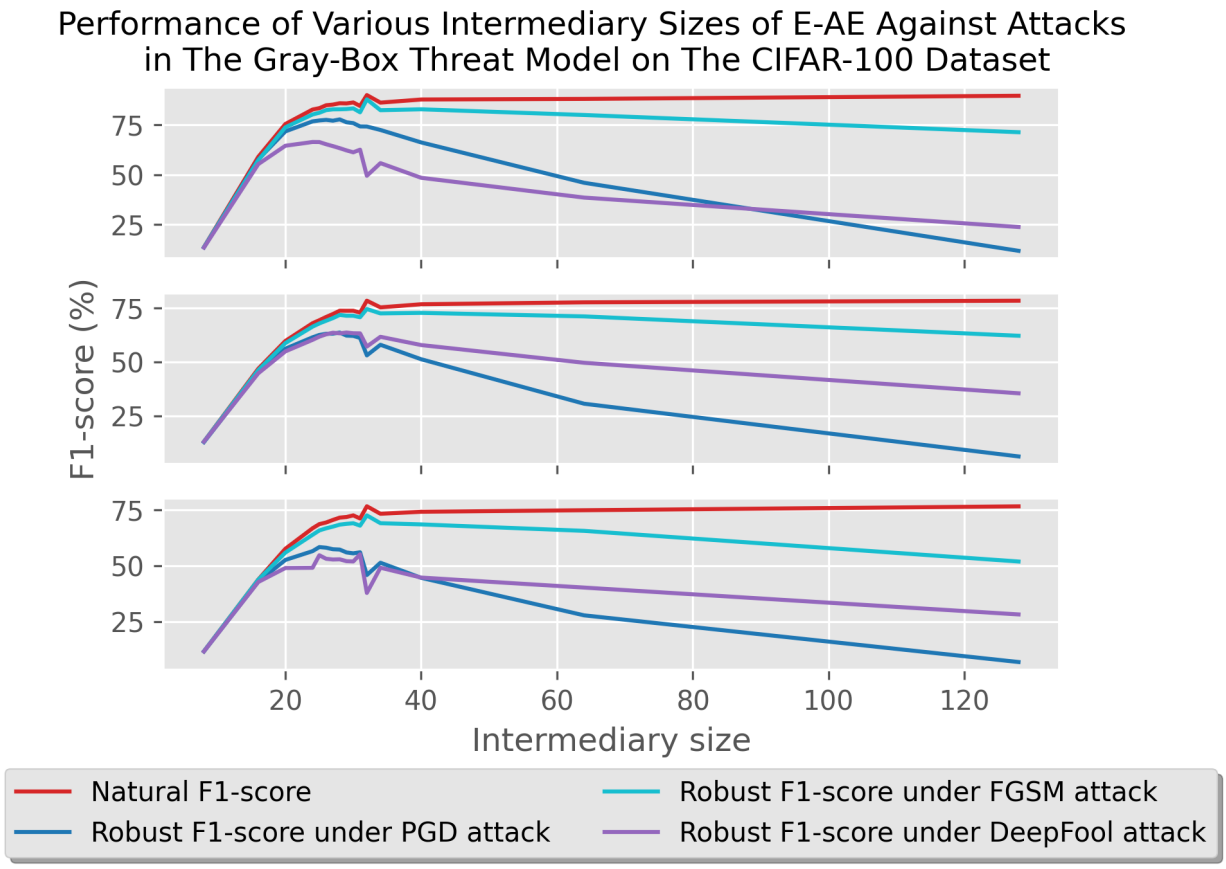

Figure 5.3: Performance of various intermediary sizes of EAE against all attacks in the gray-box threat model on the CIFAR-100 dataset. **Besides, subplots of EfficientNet-B0, ResNet-50, and VGG-19 are from top to bottom respectively.**

5.4.1.2 Robust Performance Results

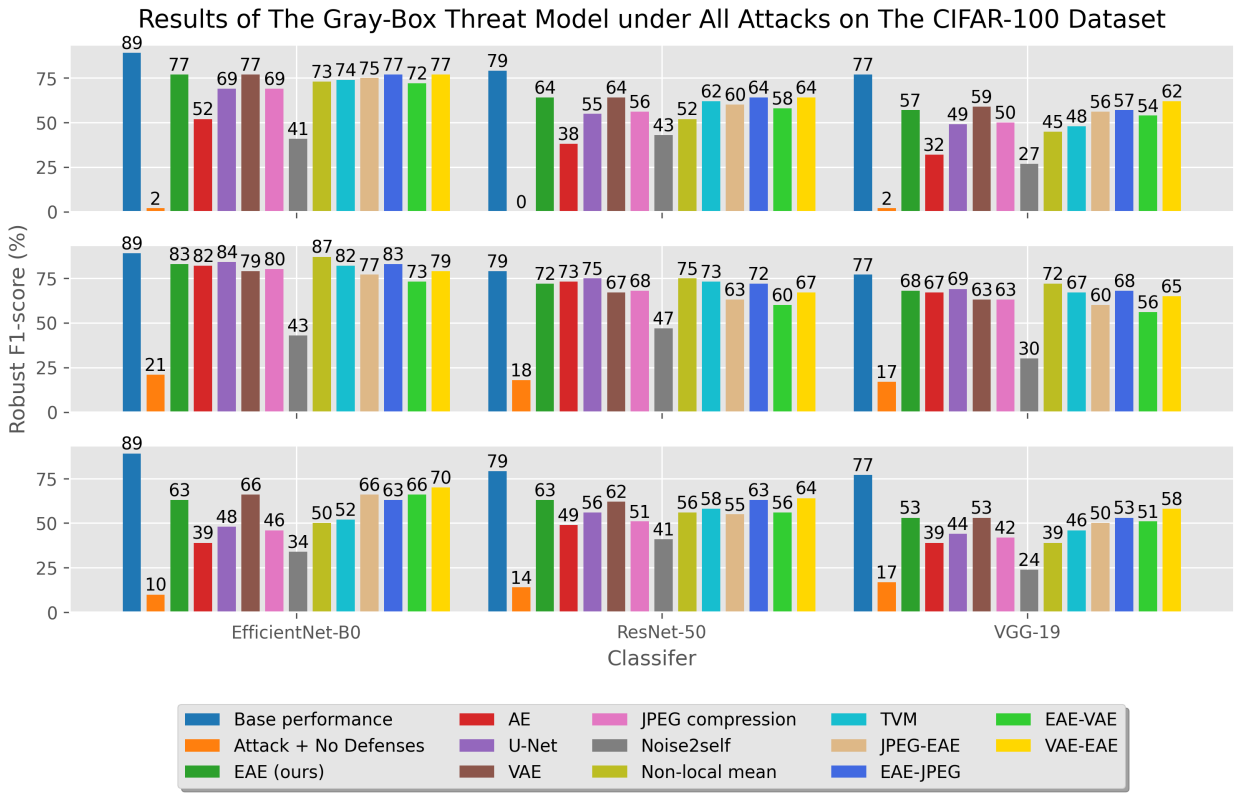

Figure 5.4: Robust classification performance of image denoisers in the gray-box threat model against all attackers on the CIFAR-100 dataset. **The top, middle, and bottom subplots are the results of the PGD, FGSM, and DeepFool attacks respectively.**

According to figure 5.4, all three classifiers achieved high benign classification performance. For example, the natural F1 scores of the EfficientNet-B0, ResNet-50, and VGG-19 classifiers were 89.22%, 79.26%, and 76.70% respectively. However, PGD, FGSM, and DeepFool attacks successfully forced all classifiers to misclassify the majority of adversarial

examples. Specifically, under the PGD, FGSM, and DeepFool attacks, the robust F1 scores were below 2%, 21%, and 17% respectively for all classifiers. On the other hand, all image denoisers substantially neutralized the effects of the adversarial perturbations crafted by all attackers. Noise2self was the least effective image denoiser on this dataset. For example, Noise2self assisted all classifiers to achieve around 45% in robust F1 scores against all three attacks on the CIFAR-100 datasets. Additionally, EAE and VAE outperformed other neural-networks-based-image denoisers and traditional image denoisers in defending all three classifiers against PGD attacks. EAE and VAE aided EfficientNet-B0 classifier to achieve a robust F1 score of around 77%, which was at least 3% higher than any other traditional or neural-networks-based image denoisers, against PGD attack. Similarly, EAE surpassed AE and U-Net by a difference of at least 7% in robust F1 scores against PGD and DeepFool attacks. Meanwhile, EAE, AE, and U-Net performed indistinguishably with regard to nullifying the impact of the FGSM attacks. Furthermore, the robust F1 scores of classifiers protected by EAE were around 3% higher than when they were defended by either JPEG compression or TVM against PGD and DeepFool attacks. Additionally, the Non-local mean outperformed all other image denoisers, including VAE and EAE, in terms of denoising FGSM-crafted adversarial examples. However, the robust F1 scores of all classifiers protected by the Non-local mean only exceeded around 3% compared to when they were defended by EAE. Nonetheless, EAE helped all three classifiers accomplish high robust F1 scores, which were at least 68%, under the FGSM attacks. As a result EAE substantially outmatched all traditional denoisers and neural-networks-based image denoisers, except VAE, with regard

to reversing the effect of adversarial perturbation crafted by PGD and DeepFool attackers.

Besides, the JPEG-EAE and EAE-JPEG chained denoisers had similar performance to the Emulated Autoencoder. Similarly, the EAE-VAE and VAE-EAE denoisers hardly outstripped either EAE or VAE regarding defending classifiers against any attacks on this dataset. For example, EfficientNet-B0 and VGG-19 protected by VAE-EAE chained denoiser accomplished the robust F1 scores of roughly 4.5% higher than VAE against DeepFool attack. However, the improvements of chained denoisers were barely noticeable in all other cases. As a result, the performance improvement of chaining image denoisers was inconclusive in the gray-box threat model on the CIFAR-100 dataset.

5.4.2 CIFAR-10 Dataset

5.4.2.1 EAE Intermediary Size Selection

Similar to the CIFAR-100 dataset, the intermediary size of 28 was the selected hyperparameter set for the EAE denoiser. The reason was that it helped all three classifiers achieve one of the highest benign and robust F1 scores against all adversarial attacks in the gray-box threat model on the CIFAR-10 dataset. The performance of various intermediary sizes of the EAE denoiser against all attacks in this threat model can be found in figure 5.5. The data illustrated in the figure 5.5 can be found in tables: A.8, A.9, and A.10.

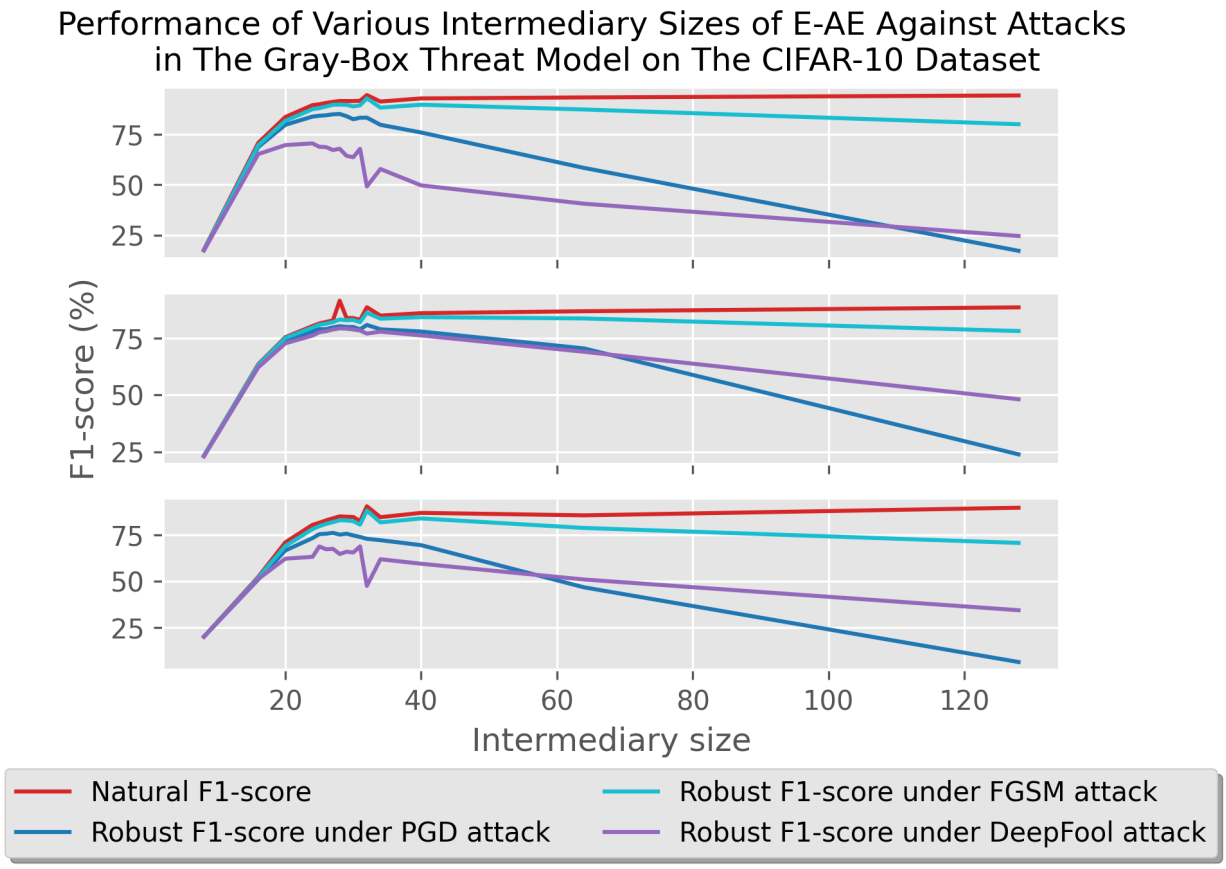

Figure 5.5: Performance of various intermediary sizes of EAE against attacks in the gray-box threat model on the CIFAR-10 dataset. **Besides, subplots of EfficientNet-B0, ResNet-50, and VGG-19 are from top to bottom respectively.**

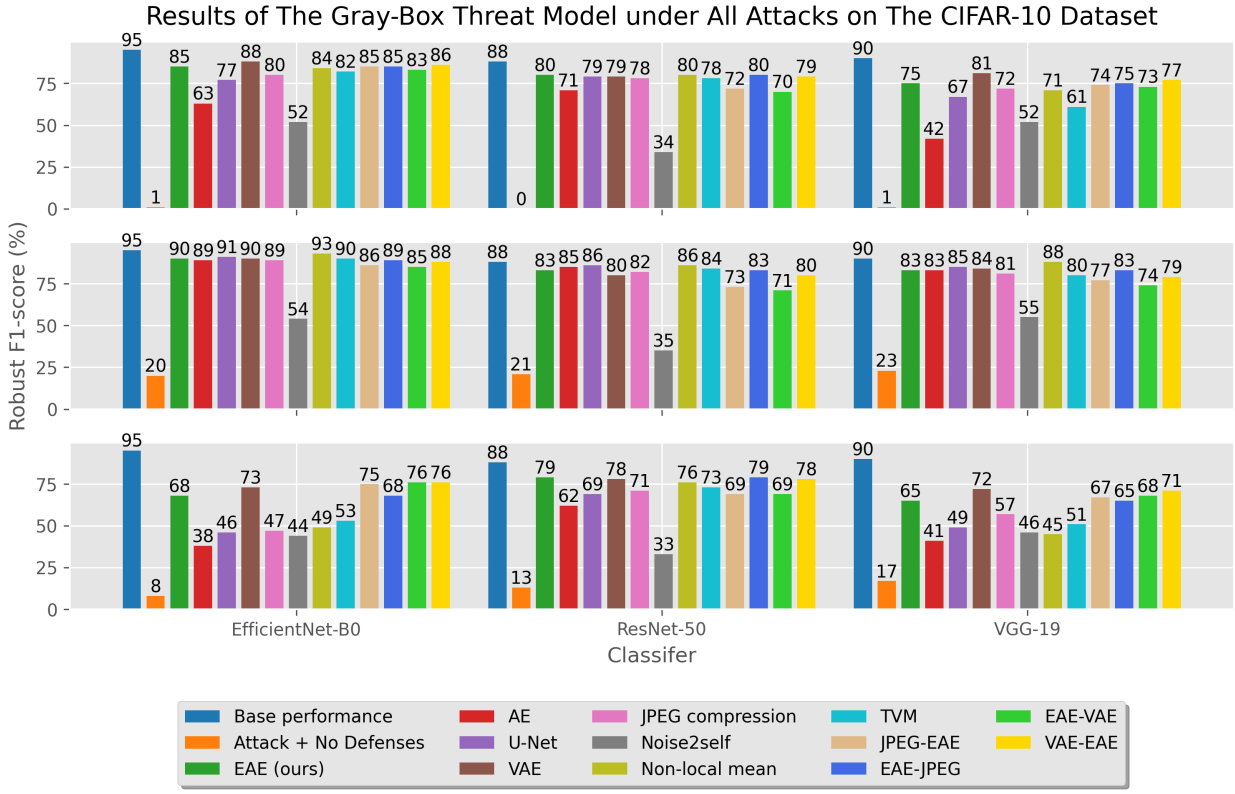

Figure 5.6: Robust classification performance of image denoisers in the gray-box threat model against all attackers on the CIFAR-10 dataset. **The top, middle, and bottom subplots are the results of the PGD, FGSM, and DeepFool attacks respectively.**

5.4.2.2 Robust Performance Results

Similar to the CIFAR-100 dataset, according to figure 5.6, EfficientNet-B0, ResNet-50, and VGG-19 classifiers accomplished high natural F1 scores, which were 95%, 88%, and 90% correspondingly. However, the robust F1 scores of classifiers under PGD, FGSM, and DeepFool attack were significantly reduced to under 1.5%, 23%, and 17% respectively. On the

other hand, the negative impacts of adversarial examples created by three attackers were substantially invalidated by all image denoisers. Specifically, the performance of EAE denoiser exceeded the performance of AE and U-Net, in terms of defending classifiers against all attacks in this dataset. In fact, the robust performance of classifiers helped by EAE denoiser was at least 10% higher than AE and U-Net against PGD and DeepFool attacks. Likewise, the denoising performance against most attacks of all traditional denoisers were surpassed by EAE. To be more specific, under the PGD and DeepFool attacks, the robust F1 scores of all classifiers aided by EAE denoiser were at least 5% and 7% respectively higher than all traditional denoisers. Meanwhile, the denoising performance of EAE, neural-networks-based image denoisers, and traditional image denoisers, except Noise2self, were similar against FGSM attack since the robust F1 scores of all classifiers aided by the mentioned denoisers were over 80%. Furthermore, the robust F1 scores of most classifiers were higher when they were protected by either EAE or VAE denoiser than other image denoisers against most attacks. In fact, the difference in terms of denoising performance of both EAE and VAE was hardly noticeable. For example, with the help of EAE denoiser, the robust F1 scores of the ResNet-50 classifier under the attack of PGD and DeepFool were 2% higher than the same figures accomplished by the VAE. On the hand, under the attacks of PGD, FGSM, and DeepFool, VAE denoiser helped the EfficientNet-B0 classifier achieve around 5% higher than EAE. Last but not least, Noise2self was the least effective image denoiser on this dataset as it only helped classifiers achieve slightly over 35% in robust F1 score against all attacks. Overall, even though all denoisers, except Noise2self, performed similarly against FGSM

attacks, EAE and VAE outstripped all other neural-networks-based and traditional image denoisers with regard to defending classifiers against PGD and DeepFool attacks.

Besides neural-networks-based and traditional image denoisers, the robust F1 scores of EfficientNet-B0 and VGG-19 classifiers aided by JPEG-EAE and EAE-JPEG chained denoiser were at least 10% higher than the same figures of JPEG compression. However, most of the robust F1 scores of all classifiers aided by either JPEG-EAE or EAE-JPEG chained denoiser were indistinguishable from the same figures of EAE denoiser. Furthermore, the performance improvement provided by chaining EAE and VAE was inconclusive according to the figure 5.6. Particularly, the EfficientNet-B0 classifier, protected by VAE-EAE chained denoiser, achieved robust F1 scores of 10% and 5% higher than EAE and VAE respectively against DeepFool attack on the CIFAR-10 dataset. Nevertheless, the robust F1 scores of classifiers aided by either EAE-VAE or VAE-EAE against the PGD and FGSM attacks were not noticeably higher than the same figures of EAE and VAE denoiser.

5.4.3 GTSRB Dataset

5.4.3.1 EAE Intermediary Size Selection

According to figure 5.7, the intermediary size of 28 was selected for hyperparameter value as it enabled EAE denoiser to help all three classifiers achieve high robust F1 scores against all three attacks, meanwhile maintaining high natural F1 scores. The data demonstrated in the figure 5.7 can be found in tables: A.11, A.12, and A.13.

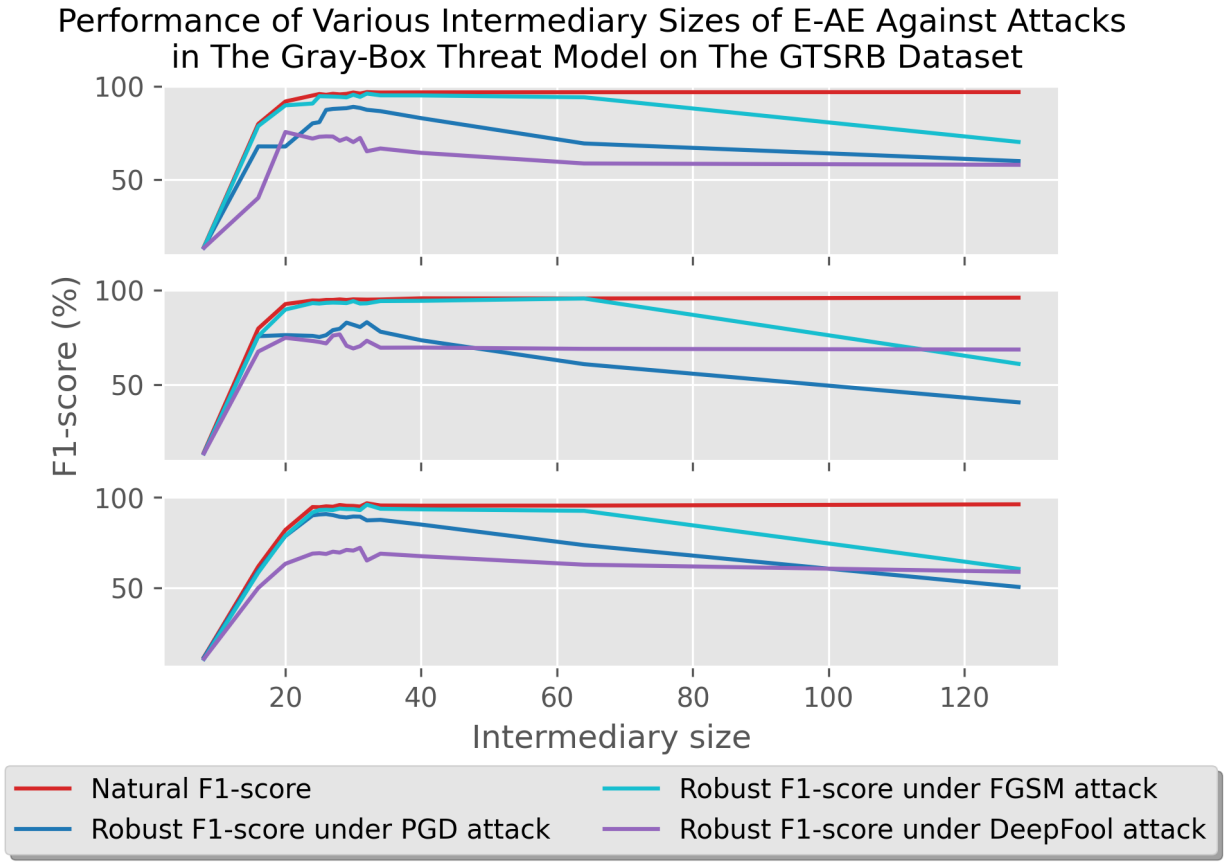

Figure 5.7: Performance of various intermediary sizes of EAE against attacks in the gray-box threat model on the GTSRB dataset. **Besides, subplots of EfficientNet-B0, ResNet-50, and VGG-19 are from top to bottom respectively.**

5.4.3.2 Robust Performance Results

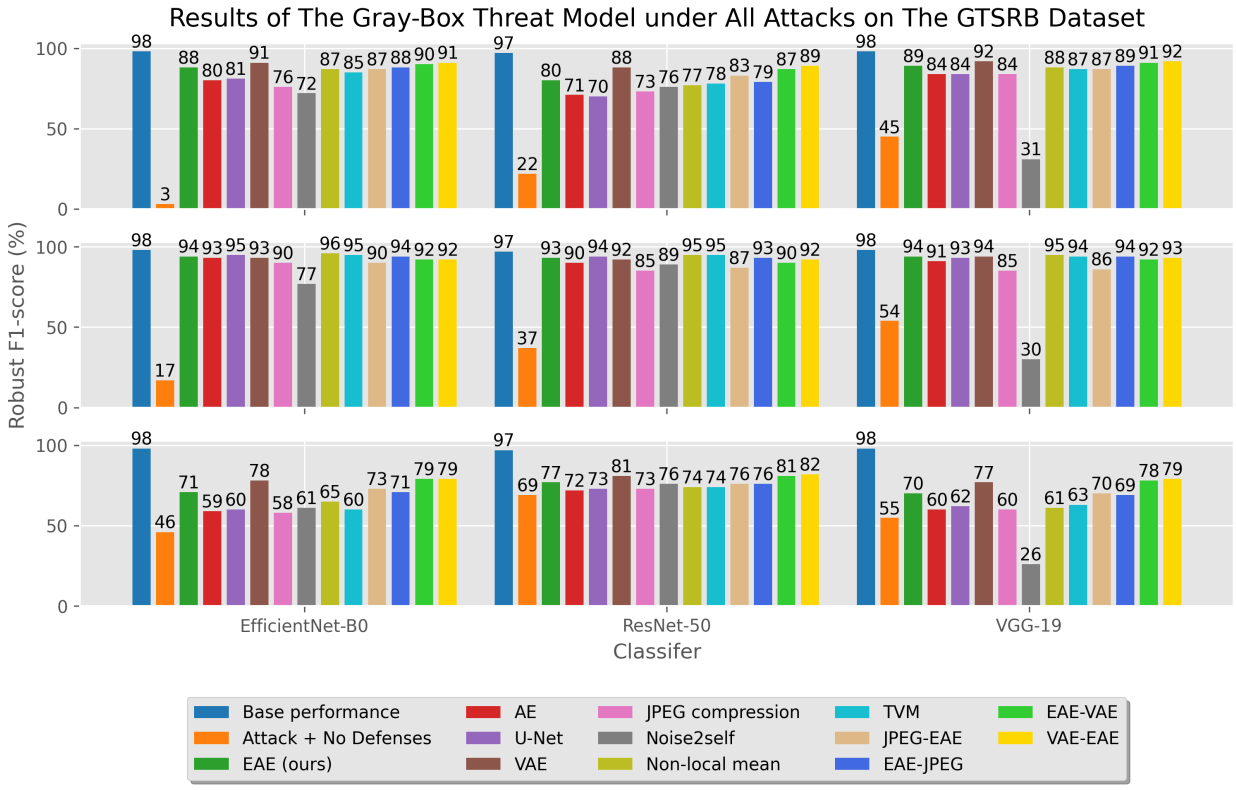

Figure 5.8: Robust classification performance of image denoisers in the gray-box threat model against all attackers on the GTSRB dataset. **The top, middle, and bottom subplots are the results of the PGD, FGSM, and DeepFool attacks respectively.**

Based on figure 5.8, all classifiers accomplished high natural F1 scores which were at least 97%. However, PGD, FGSM, and DeepFool attackers caused significant misclassification to all classifiers. For instance, the robust F1 scores of EfficientNet-B0 and VGG-19 classifiers under all attacks were under 50% while the same figures for the ResNet-50 classifier were

below 70%. On the other hand, the effect of adversarial perturbations was reversed by all image denoisers, except the Noise2self method. Moreover, under the attack of PGD and DeepFool algorithms, the robust F1 scores of all classifiers aided by the VAE denoiser were around 5% higher than the same figures of classifiers defended by the EAE. Meanwhile, both VAE and EAE denoiser performed similarly while denoising adversarial examples crafted by the FGSM algorithm on this dataset. Furthermore, the denoising performance of EAE exceeded the performance of AE and U-Net denoisers by a large difference. For instance, the robust F1 scores of all classifiers aided by the EAE denoiser were roughly 7% higher than AE or U-Net denoiser. Similarly, under the PGD and DeepFool attacks, the robust F1 scores of all classifiers protected by EAE denoiser were around 2% higher than JPEG compression, Non-local mean, and TVM denoisers. Meanwhile, the denoising performances of EAE, AE, U-Net, and traditional image denoisers, except the Noise2self method, were nearly identical in terms of denoising adversarial examples crafted by the FGSM attacker. Even though the Non-local mean outperformed all other image denoisers, the robust F1 scores of classifiers aided by the Non-local mean method were roughly 1.5% higher than the same figures accomplished by EAE and VAE. Overall, even though all image denoisers, except Noise2self, performed similarly against FGSM attacks, EAE and VAE denoiser outperformed all other neural-networks-based and traditional denoisers with regard to denoising adversarial examples crafted by PGD and DeepFool attackers.

The VAE-EAE and EAE-VAE chained denoiser slightly outperformed VAE and EAE in terms of defending classifiers against PGD and DeepFool attacks. To be more specific, the

robust F1 scores of all classifiers protected by either VAE-EAE or EAE-VAE were 2% higher than the same figures of either VAE or EAE denoiser. However, the denoising performance of VAE-EAE, EAE-VAE, EAE, and VAE were the same regarding nullifying the adversarial perturbation crafted by the FGSM attacker. As a result, the improvement of the overall robustness of chaining VAE and EAE was inconclusive. Similarly, the denoising performance of the EAE-JPEG chained denoiser surpassed the performance of JPEG compression. Particularly, the robust F1 scores of all classifiers defended by the EAE-JPEG chained denoiser were around 7% higher than the JPEG compression. However, EAE, JPEG-EAE, and EAE-JPEG chained denoiser performed similarly in terms of defending classifiers against FGSM attacks. Therefore, the improvement of the system robustness of the chained denoisers, comprised of EAE and JPEG compression, was not significant.

5.4.4 Overall Evaluation of Gray-box Threat Model Results

EAE was shown to be an effective image denoiser in defending all classifiers against all attacks in this threat model. EAE substantially outperformed AE and U-Net by 10%, and all traditional image denoisers by 5% in robust F1 scores while defending all classifiers against PGD and DeepFool attacks respectively. On the other hand, VAE outperformed EAE by roughly 4% in defending classifiers against PGD and DeepFool attacks in the gray-box threat model on the GTSRB dataset. Meanwhile, EAE outperformed VAE by around 2% in defending classifiers against PGD and DeepFool attacks on the CIFAR-100 dataset. Moreover, VAE and EAE had similar performances in most other scenarios. Thus, VAE

performance was slightly better than the performance of EAE concerning defending classifiers in this threat model on all datasets. Lastly, the improvement of all chained denoisers was not significant in this threat model.

5.5 White-Box Threat Model Results

As the Non-local mean, Noise2self, and TVM methods took a significantly long period to reconstruct 1000 images and do not have GPU-support implementations, they were not selected as image denoisers in this threat model. However, JPEG compression, which took significantly less amount of time to reconstruct images, was utilized as the sole traditional image denoiser in this threat model. Additionally, this section presents and evaluates the denoising performance of a group of image denoisers, including EAE, AE, U-Net, VAE, JPEG compression, and chained denoisers.

5.5.1 CIFAR-100 Dataset

According to figure 5.9, all selected image denoisers dramatically nullified the adversarial effects of the PGD and FGSM attackers. Meanwhile, only the chained denoisers and VAE significantly improved the robust F1 scores of classifiers under the DeepFool attacks. Additionally, EAE and JPEG compression performed similarly in terms of defending classifiers against adversarial attacks. For example, under the attack of PGD and FGSM algorithms, the robust F1 scores of all classifiers protected by EAE denoiser were around 5% higher than the same figures of JPEG compression. Meanwhile, under the attack of the DeepFool algorithm, the robust F1 scores of all classifiers aided by JPEG compression were roughly

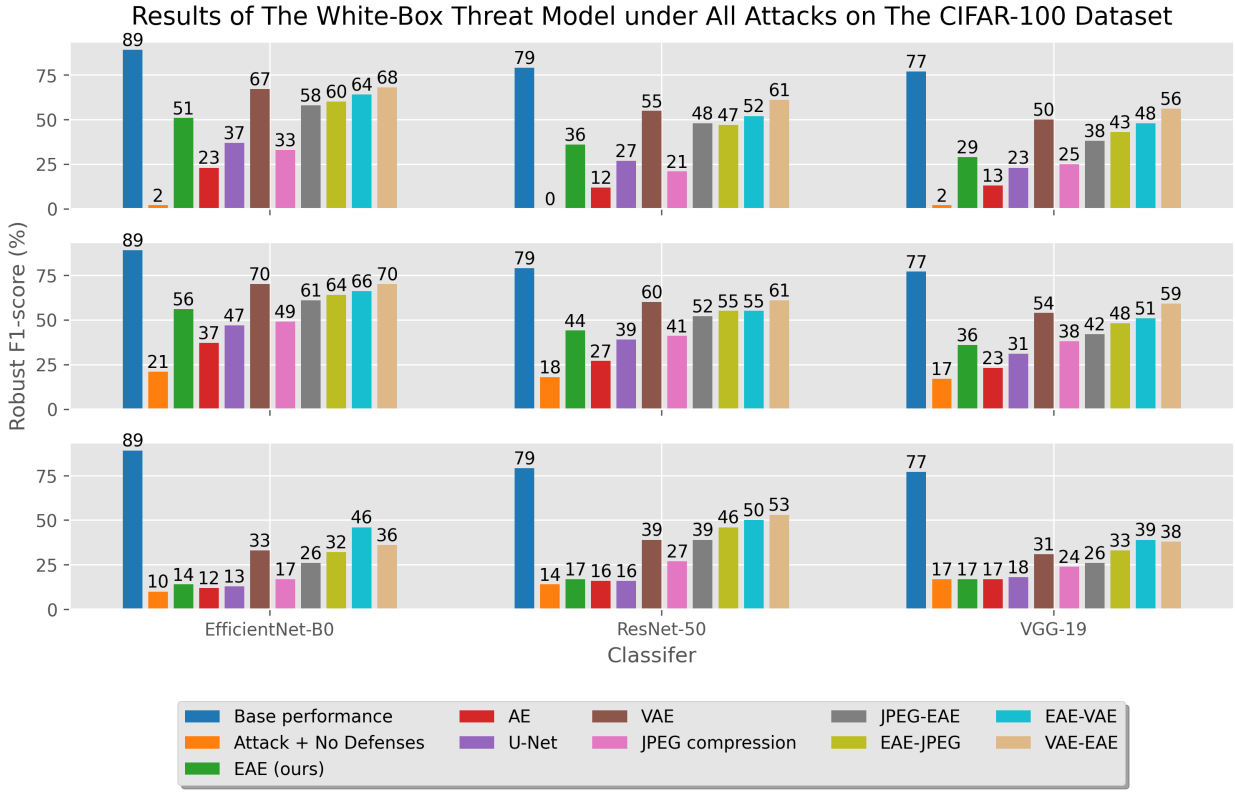

Figure 5.9: Robust classification performance of image denoisers in the white-box threat model against all attackers on the CIFAR-100 dataset. **The top, middle, and bottom subplots are the results of the PGD, FGSM, and DeepFool attacks respectively.**

6% higher than EAE denoiser. Moreover, the denoising performance of EAE was significantly higher than the performance of either AE or U-Net denoiser in terms of defending classifiers against PGD and FGSM attacks. To be more specific, the robust F1 scores of classifiers defended by EAE denoiser were around 7% higher than AE and U-Net. However, EAE, AE, and U-Net had similar performances regarding defending classifiers against DeepFool attacks. Besides, the classifiers defended by VAE denoiser achieved robust F1 scores

of around 10% higher than EAE denoiser. Overall, EAE outperformed AE, U-Net, and JPEG compression in terms of protecting classifiers against PGD and FGSM attacks while moderately underperforming VAE in most cases on this dataset.

Chained denoisers outstripped all image denoisers by a significant difference on this CIFAR-100 dataset. Specifically, VAE-EAE chained denoiser outperformed all other image denoisers in terms of defending classifiers against PGD and FGSM attacks. For instance, under the attacks of PGD and FGSM attackers, the robust F1 scores of classifiers protected by VAE-EAE were around 5% and 10% higher, than when they were protected by VAE and EAE denoisers respectively. Moreover, VAE-EAE helped ResNet-50 and VGG-19 classifiers achieve robust F1 scores of at least 7% higher than VAE against DeepFool attacks. Meanwhile, under the protection of EAE-VAE, EfficientNet-B0 accomplished 15% higher robust F1 scores than VAE denoiser against DeepFool attacks. Furthermore, the robust F1 scores of three classifiers defended by EAE-JPEG chained denoiser, were at least 10% higher than when they were protected by either EAE or JPEG compression against three evasion attacks. As a result, the improvement provided by chaining EAE with either JPEG compression or VAE was conclusive in this dataset.

5.5.2 CIFAR-10 Dataset

Similar to the CIFAR-100 dataset, all selected image denoisers successfully invalidated the effect of adversarial perturbations crafted by PGD and FGSM attackers. For example, under the PGD attacks, the robust F1 scores of classifiers without defenses were 0% while the robust

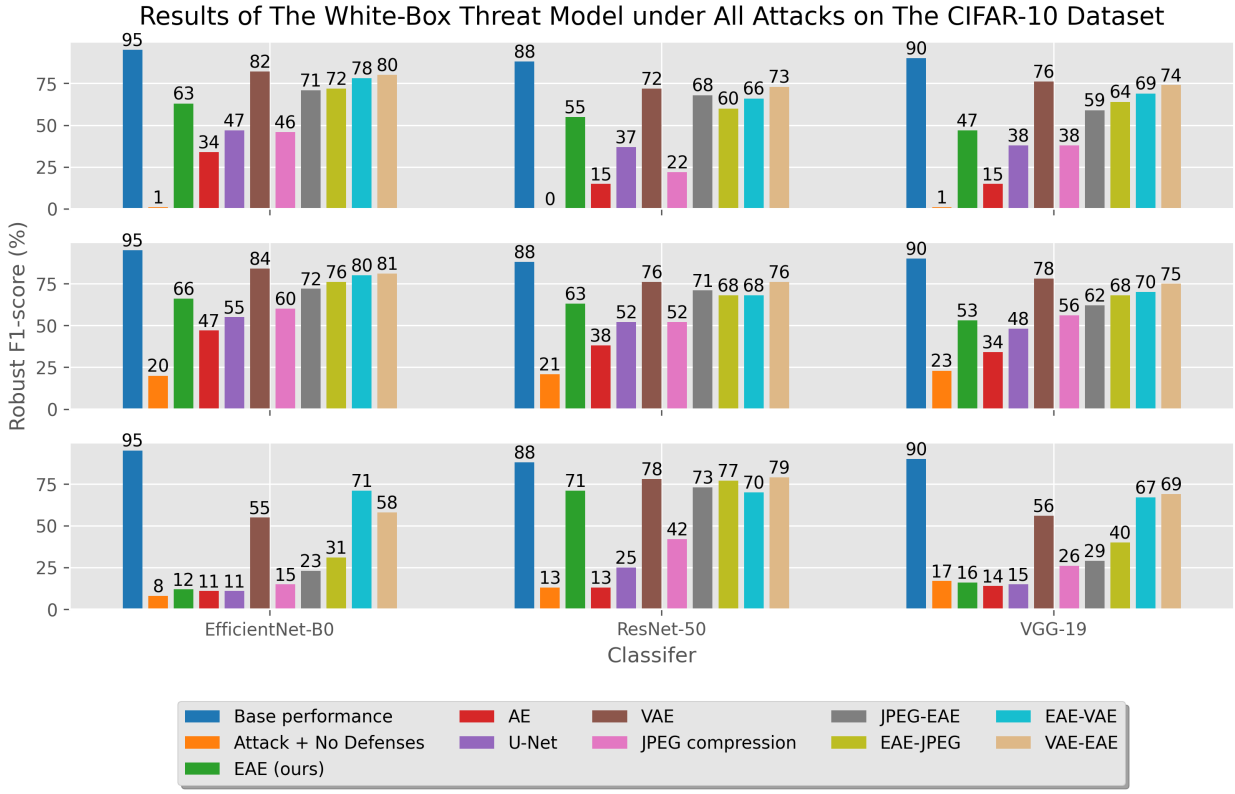

Figure 5.10: Robust classification performance of image denoisers in the white-box threat model against all attackers on the CIFAR-10 dataset. **The top, middle, and bottom subplots are the results of the PGD, FGSM, and DeepFool attacks respectively.**

F1 scores were at least 15% when they were protected by any image denoisers. However, the impact of the adversarial perturbations generated by the DeepFool algorithms were mostly negated by chained denoisers and VAE denoiser. Additionally, VAE outperformed all other image denoisers in terms of defending all three classifiers against PGD and FGSM attacks. Specifically, under all three attacks, the EfficientNet-B0 and VGG-19 classifiers protected by VAE denoiser accomplished at least 10% higher robust F1 scores than EAE.

However, the performance difference between EAE and VAE dramatically narrowed when it comes to defending the ResNet-50 classifier against DeepFool attacks. Furthermore, EAE outperformed JPEG compression against most attacks. Specifically, under the DeepFool attacks, the robust F1 scores of three classifiers protected by EAE denoiser were on average 6% higher than JPEG compression. All classifiers secured by the EAE denoiser achieved an average of 12% higher robust F1 scores than when they were defended by JPEG compression under PGD and FGSM attacks. Moreover, EAE provided significantly better protection for classifiers against PGD and FGSM attacks than AE and U-Net. Particularly, the robust F1 scores of classifiers defended by EAE were on average 11% higher than when they were under the protection of either AE or U-Net against PGD or FGSM attacks. In addition to PGD and FGSM attacks, all three denoisers, including, EAE, AE, and U-Net, performed similarly when it comes to protecting EfficientNet-B0 and VGG-19 classifiers against DeepFool attacks. Overall, EAE outperformed AE, U-Net, and JPEG compression while underperforming VAE in terms of defending classifiers against most attacks.

The chained denoisers comprised of EAE and JPEG compression noticeably outperformed either EAE or JPEG compression. For example, EAE-JPEG chained denoiser helped all classifiers achieve an average of 10% higher than when they were defended by either EAE or JPEG compression against all attacks. Noticeably, the denoising performance of EAE-JPEG was roughly 88% of the performance of VAE denoiser while taking 25% less time to denoise 1000 images. On the other hand, under the DeepFool attacks, the robust F1 scores of all classifiers protected by either VAE-EAE or EAE-VAE chained denoiser were roughly

3% higher than VAE denoiser. In fact, the chained denoisers comprised of VAE and EAE outperformed all other image denoisers in protecting classifiers against DeepFool attacks. However, the performance of EAE-VAE and VAE-EAE chained denoisers and VAE were similar in protecting all classifiers against PGD and FGSM attacks. Thus, the improvement of robustness provided by chaining EAE and VAE denoiser was inconclusive on this dataset.

5.5.3 GTSRB Dataset

Based on figure 5.11, all image denoisers noticeably reversed the effect of adversarial perturbations crafted by PGD and FGSM attackers. For example, the robust F1 scores of all classifiers without defenses were below 45% and 53% under the PGD and FGSM attacks respectively. Meanwhile, under the PGD and FGSM attacks, the robust F1 scores of all classifiers protected by image denoisers were around 65%, which was significantly higher than without defenses. Besides, the impact of adversarial examples created by DeepFool attacks was moderately nullified by image denoisers. For instance, there was an average of 8% increase in robust F1 scores when all three classifiers were protected by image denoisers against DeepFool attacks. In fact, all image denoisers had similar denoising performance in terms of defending classifiers against DeepFool attacks. Furthermore, the robust F1 scores of classifiers secured by VAE denoiser were around 14% higher than when they were defended by the EAE against PGD and FGSM attacks. On the other hand, the denoising performance of EAE surpassed the ones of AE, U-Net, and JPEG compression in several cases. For instance, under PGD and FGSM attacks, the robust F1 scores of all classifiers protected by

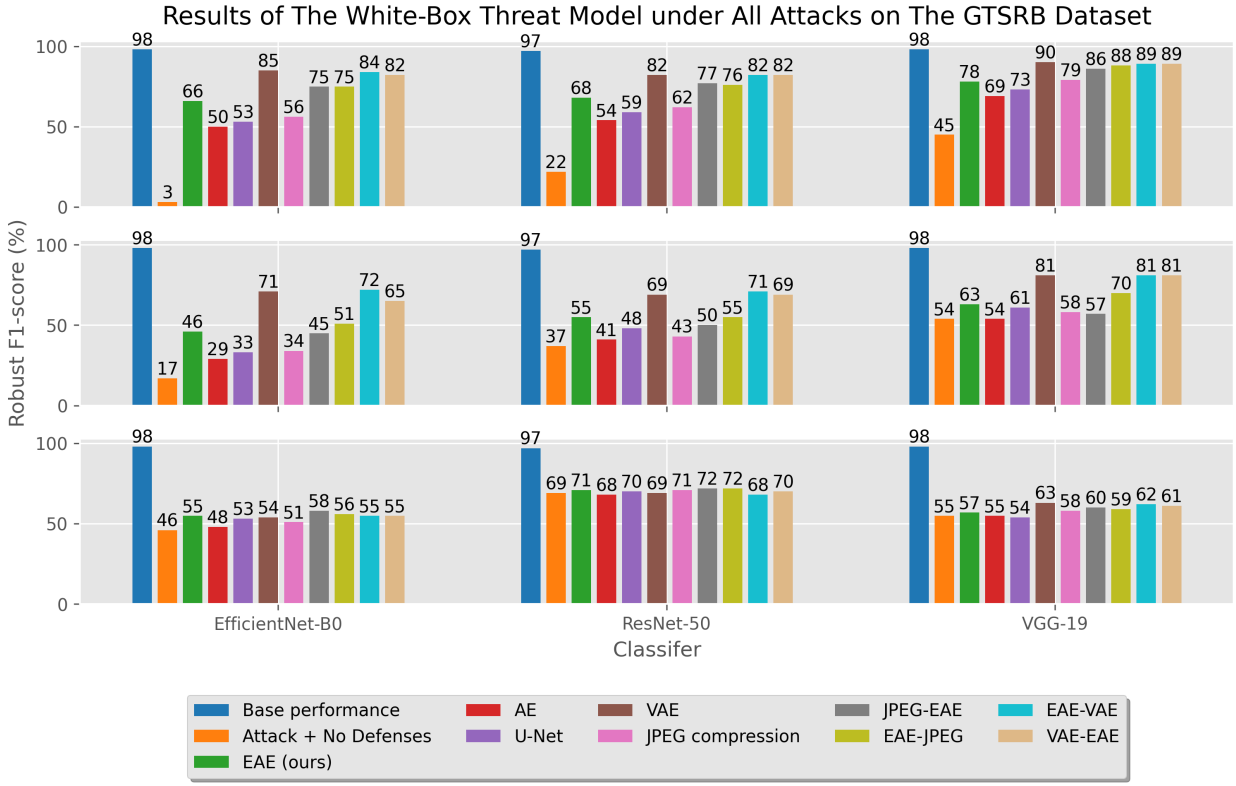

Figure 5.11: Robust classification performance of image denoisers in the white-box threat model against all attackers on the GTSRB dataset. **The top, middle, and bottom subplots are the results of the PGD, FGSM, and DeepFool attacks respectively.**

EAE denoiser were around 7% higher when they were secured by either AE, U-Net, or JPEG compression. Overall, EAE outperformed all image denoisers, except for VAE and chained denoisers, in terms of defending classifiers against most attacks.

The chained denoiser EAE-VAE and VAE-EAE helped classifiers achieve at least 5% higher robust F1 scores than the EAE denoiser. However, both EAE-VAE and VAE-EAE did not outperform VAE. On the other hand, under the attacks of PGD and FGSM attackers,

the robust F1 scores of all classifiers protected by the EAE-JPEG chained denoiser were around 8% higher when they were defended by either EAE or JPEG compression. As a result, there were conclusive improvements in robust F1 scores on this dataset when EAE and JPEG compression were chained together. Meanwhile, the improvement of robustness provided by either VAE-EAE or EAE-VAE denoiser was inconclusive on this dataset.

5.5.4 Overall Evaluation of White-box Threat Model Results

Similar to the results of the gray-box threat model, EAE noticeably outperformed AE, U-Net, and JPEG compression in terms of denoising adversarial examples crafted in most attacks. For example, under PGD and FGSM attacks on three datasets, the robust F1 scores of classifiers protected by EAE denoiser were around 9% higher than when they were protected by either AE, U-Net, or JPEG compression. On the other hand, on all datasets, the robust F1 scores of classifiers defended by VAE denoiser were around 8% higher than when they were protected by EAE against all three attackers. Additionally, on all three datasets, by chaining EAE denoiser and JPEG compression, there were consistent increases of around 10% in robust F1 scores compared to the same figures of either EAE denoiser or JPEG compression against most attacks. Additionally, EAE-JPEG chained denoiser achieved around 90% of the VAE performance against most of the adversarial attacks on this threat model. However, the VAE-EAE chained denoiser only showed clear increases in robust F1 scores of all classifiers against most attacks on the CIFAR-100 dataset. Therefore, the robust F1 scores improvement of chaining VAE and EAE denoiser within this white-box

threat model was inconclusive.

Chapter 6

DISCUSSION

For three datasets, Emulated Autoencoder was shown to surpass most image denoisers in nullifying the majority of the impacts caused by PGD, FGSM, and DeepFool attacks in the gray-box threat model. For the white-box threat model, the EAE outperformed AE, U-Net, and JPEG compression by substantial differences in robust F1 scores against most attacks on all datasets. Additionally, Pareto frontiers, which are optimal solutions in terms of performance and reconstruction time trade-off, of image denoisers can be found in the figure 6.1. Averaging across all classifiers, attacks, and threat models, VAE and EAE were the top-performing defenses where EAE maintained 85% of the robust F1 score of VAE at 0.8% of the time-cost which is specifically demonstrated in the figure 6.1. Moreover, the EAE denoiser required no training and took the smallest amount of time to reconstruct 1000 images compared to other image denoisers. Additionally, the proposed denoiser barely reduced the benign performance of classifiers compared to the VAE. By physically skipping and modifying pixels with bilinear interpolation, EAE successfully negated the impacts of adversarial perturbations, specially crafted by the PGD and FGSM attackers, in most attack scenarios of this experiment. Besides, some image denoisers, such as EAE, AE, U-Net, and JPEG compression did not achieve high performance in denoising DeepFool-crafted

adversarial examples as to PGD and FGSM attacks in a few attack scenarios of the white-box threat model.

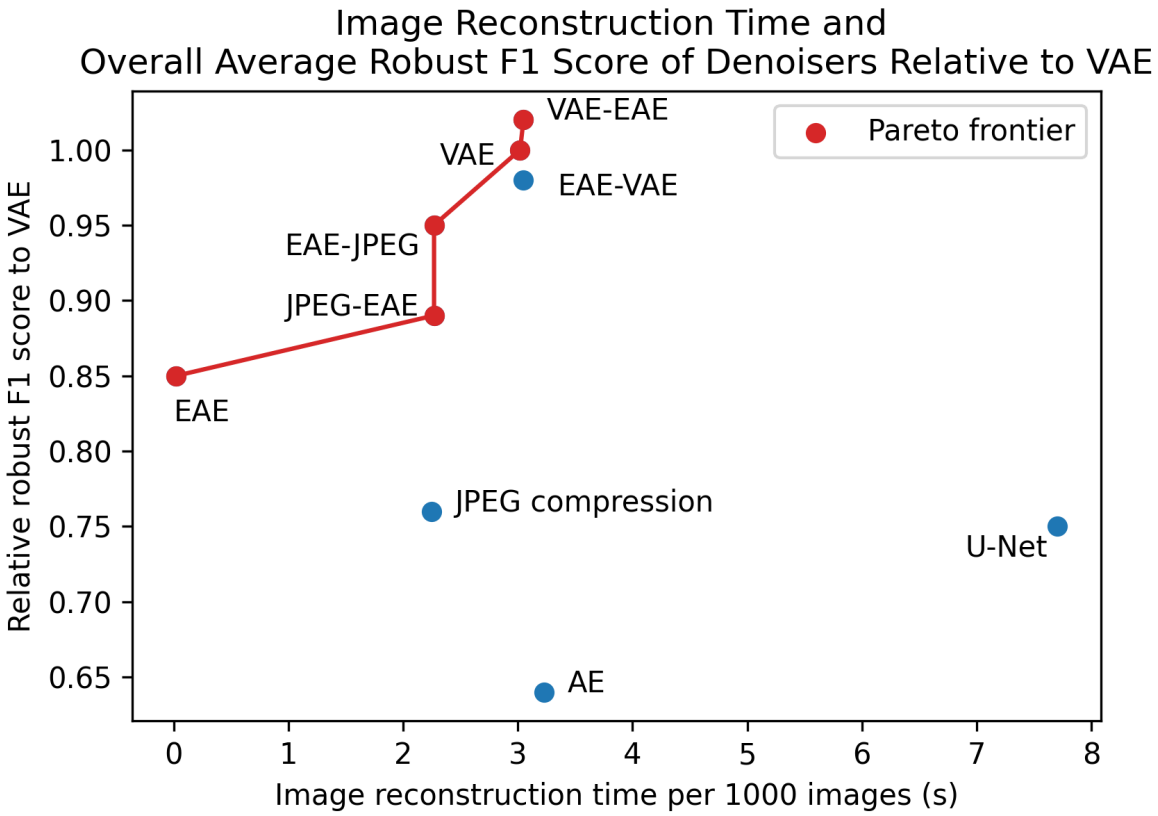

Figure 6.1: Image reconstruction time (s) per 1000 images and overall average robust F1 score of denoisers relative to VAE

Similar to EAE, chained denoisers were also shown to be effective against evasion attacks. One of the main components of chained denoisers is the Emulated Autoencoder. The reason is two-fold; first, EAE took significantly less time than other denoisers to reconstruct images and minimally impacted the benign classification performance. Secondly, as the impact of

adversarial perturbations might be negated by one denoiser but not the other; therefore, by chaining image denoisers, the adversarial defense system theoretically mitigates the evasion attacks more sufficiently than standalone image denoisers. As a result, by chaining EAE denoiser and JPEG compression, the chained denoiser EAE-JPEG outperformed EAE and JPEG compression consistently in the white-box threat model. However, EAE-JPEG did not provide clear improvements in the gray-box threat model as in the white-box threat model. This could be because either EAE or JPEG compression already achieved high performance in the gray-box threat model that it takes a more sophisticated adversarial defense mechanism to further improve the overall performance.

VAE denoiser performed similarly to EAE in the gray-box threat model while outperforming the Emulated Autoencoder in the white-box threat model. However, VAE caused more substantial losses in benign classification performance of classifiers in the CIFAR-100 dataset and took a significantly longer period to train and denoise images than EAE. This might be due to the complex structure of VAE. The effectiveness of the VAE denoiser might be due to the mean and the standard deviation vector processing the information before it was used to construct the randomly sampled latent space. By doing so, the data distribution was better captured by VAE than other image denoisers to mitigate the negative impacts of evasion attacks.

There are a few limitations to using EAE as an adversarial defense. Firstly, if adversarial performance of defense is required to be equal to or higher than the average performance of VAE, EAE is unable to satisfy such requirement. Lastly, the support vector machine (SVM)

was shown to underperform CNN-based classifiers on large datasets [19]. Thus, non-CNN-based classifiers such as SVM may not perform well with EAE. This could be because SVM does not capture the data distribution as sufficiently as CNN-based classifiers.

6.1 Practical Implications

As evasion attacks may evolve in the future, real-world systems may include sets of adversarial defense mechanisms such as adversarial training or traps to increase the system's robustness and block malicious sources of data. Thus, these may dramatically increase the total time for the overall system or cause losses of benign F1 scores of CNN classifiers. Thus, EAE along with EAE-based chained denoisers were among the top selections if applications require denoisers with significantly lower image reconstruction time, smaller drop in benign classification performance, instant deployment without training, and high performance against most attack scenarios. On the other hand, when an image denoiser was demanded to protect classifiers slightly better than EAE against evasion attacks in the white-box threat model, VAE is a better fit as an image denoiser. VAE is only the better option than EAE when larger drops in benign F1 scores, training time, and image reconstruction time are not major issues. Additionally, if the gray-box threat model is the primary attacking threat model to a ML system, EAE can be selected as an adversarial defense due to its high effectiveness against mentioned evasion attacks in the gray-box threat model. Meanwhile, if the white-box threat model is the primary threat model, EAE alone achieves roughly 75% of the VAE performance in defending against most attacks. However, according to the figure 6.1,

EAE-JPEG chained denoiser achieved nearly 94% of VAE overall performance meanwhile taking 25% less image reconstruction time, demanding no training, and causing smaller drops in benign F1 scores. Furthermore, unlike the experiment conducted in this thesis, training for neural-networks-based denoisers can be continuous to capture the latest data distribution in the real-world scenario. Thus, the training cost of denoisers such as VAE can be a recurring one which makes adversarial defense such as EAE a better fit in the real-world system where recurring training of image denoisers is a major obstacle. Moreover, based on the results of training and image reconstruction time of image denoisers, utilizing EAE as an image denoiser is more viable than using VAE to process a high volume of data in real-world ML applications. Lastly, EAE can be utilized as an adversarial defense due to its simplicity for CNN in the growing field of the Internet of Things where hardware limitations may occur.

The performance of the EAE-JPEG chained denoiser was consistently higher than either EAE or JPEG compression in all cases in the white-box threat model. Therefore, by chaining EAE with JPEG, high denoising performance can be achieved in both threat models, while taking 25% less image reconstruction time and smaller losses in benign F1 scores than VAE. As a result, it is worth chaining EAE with other image denoisers in the real-world system.

It was estimated that Instagram, which is one of the social media platforms, saw 95 million images posted daily on its system[51]. The estimated total processing time of VAE, EAE-JPEG, and EAE can be found in table 6.1. In table 6.1, it is assumed that 224 x 224 x 3 is the image resolution, and one NVIDIA RTX-6000 GPU is utilized. VAE may take roughly

3 days and 7 hours to finish processing. Meanwhile, it is estimated that EAE only takes 36 minutes to complete the same task. As a result, more than one hundred NVIDIA RTX-6000 GPU are required to run in parallel for VAE to achieve a similar image reconstruction time to EAE. The one-time cost may include purchasing the GPU along with other computer hardware to accommodate the upgraded workstation. The additional recurring costs may include housing and utility to allow the new workstation to function properly. These costs can be issues for businesses desiring to solidify the adversarial defense against potential evasion attacks.

Table 6.1: Estimated total image reconstruction time for processing 95 million images on Instagram

Method	Time
VAE	3 days and 7 hours
EAE-JPEG	2 days and 12 hours
EAE	36 minutes

Chapter 7

CONCLUSION

In this thesis, the trade-offs, including loss of benign F1 scores of classifiers, training, and image reconstruction time, of numerous image denoisers have been explored. As a result, VAE caused noticeable drops in benign F1 scores of CNN classifiers in the CIFAR-100 dataset while adding hours of training time and doubling the overall inference time of the ML system. On the other hand, EAE outperformed most image denoisers and performed similarly to VAE in the gray-box threat model while requiring no training, taking significantly less amount of image reconstruction time and causing dramatically smaller drops in benign F1 scores. Furthermore, even though the performance of EAE was less than the same figure of VAE in the white-box threat model, EAE can be chained with JPEG compression to accomplish a similar performance to VAE while taking a smaller amount of reconstruction time. Thus, similar to chaining EAE with JPEG compression, sequentially combining EAE with other image denoisers or other adversarial defenses might increase the overall robustness while maintaining the cost of the defense system.

In the real-world ML system, integrating image denoisers possessing insignificant trade-offs, such as EAE, to a system as adversarial defenses is beneficial. The reason is that the current adversarial defense system might have already increased the overall time and

negatively impact the benign classification performance. Therefore, by adding EAE, the overall robustness of the system can be improved without significant trade-offs. This is important to systems that require secure and fast ML systems that process high volume data such as visual content moderation in social networks or self-driving vehicles.

7.1 Future Work

The results found in this thesis helps broaden the view of selecting image denoisers based on trade-offs and performance as defenses against evasion attacks. Additionally, the approach of using EAE as well as chaining EAE with another image denoiser yielded fine results in terms of defending classifiers against several evasion attacks in both the gray-box and white-box threat model. Thus, the effect of chaining EAE with two or more image denoisers can be explored in future work. Furthermore, a black-box threat model, which does not allow attackers access to either the parameter of the target classifiers or the defense, can be conducted in the future even though this threat model is significantly weaker than both gray-box and white-box one. Next, as different data formats such as audio, text, or location can be interpreted as multi-dimensional arrays, which is similar to image data, EAE can be evaluated as a defense against evasion attacks on those non-image data. Lastly, EAE and JPEG compression can be benchmarked on Internet of Things (IoT) devices such as Raspberry Pi in terms of image reconstruction time to measure their viability in the real-world IoT system.

BIBLIOGRAPHY

[1] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016. [Online]. Available: http://arxiv.org/abs/1604.07316

[2] S. Cheng and G. Bender, "AutoML: Automating the design of machine learning models for autonomous driving," Jan. 2019. [Online]. Available: https://blog.waymo.com/2019/07/automl-automating-design-of-machine.html

[3] C.-J. Wu, D. Brooks, K. Chen, D. Chen, S. Choudhury, M. Dukhan, K. Hazelwood, E. Isaac, Y. Jia, B. Jia, T. Leyvand, H. Lu, Y. Lu, L. Qiao, B. Reagen, J. Spisak, F. Sun, A. Tulloch, P. Vajda, X. Wang, Y. Wang, B. Wasti, Y. Wu, R. Xian, S. Yoo, and P. Zhang, "Machine Learning at Facebook: Understanding Inference at the Edge," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019.

[4] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," 2019.

[5] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *International Conference on Learning Representations*, 2015.

[6] K. Yuan, D. Tang, L. Liao, X. Wang, X. Feng, Y. Chen, M. Sun, H. Lu, and K. Zhang, "Stealthy Porn: Understanding Real-World Adversarial Images for Illicit Online Promotion," *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 952–966, 2019.

[7] "DARTS: Deceiving Autonomous Cars with Toxic Signs," 2018.

[8] B. C. Kim, J. U. Kim, H. Lee, and Y. M. Ro, "Revisiting role of autoencoders in adversarial settings," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 1856–1860.

[9] Y. Luo and H. Pfister, "Adversarial defense of image classification using a variational auto-encoder," *CoRR*, vol. abs/1812.02891, 2018. [Online]. Available: http://arxiv.org/abs/1812.02891

[10] X. Zhang and F. Jia, "Dilated denoising u-net network improving the adversarial robustness," *Journal of Physics: Conference Series*, vol. 1621, no. 1, p. 012082, aug 2020. [Online]. Available: https://doi.org/10.1088/1742-6596/1621/1/012082

[11] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=SyJ7ClWCb

[12] Y. Deng, X. Zheng, T. Zhang, C. Chen, G. Lou, and M. Kim, "An analysis of adversarial attacks and defenses on autonomous driving models," in *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2020, pp. 1–10.

[13] J. Li, F. Schmidt, and Z. Kolter, "Adversarial camera stickers: A physical camera-based attack on deep learning systems," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 3896–3904. [Online]. Available: https://proceedings.mlr.press/v97/li19j.html

[14] NVIDIA, "Solutions for Self-Driving Cars & Autonomous Vehicles," 2022. [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/

[15] ——, "The World's First Ray Tracing GPU nvidia Quadro RTX 6000," 2019. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/quadro-product-literature/quadro-rtx-6000-us-nvidia-704093-r4-web.pdf

[16] M. Labrie, "NVIDIA Introduces DRIVE AGX Orin — Advanced, Software-Defined Platform for Autonomous Machines," Dec. 2019. [Online]. Available: https://nvidianews.nvidia.com/news/nvidia-introduces-drive-agx-orin-advanced-software-defined-platform-for-autonomous-machines

[17] T. Guzella and W. Caminhas, "A review of machine learning approaches to Spam filtering," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10 206–10 222, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741740900181X

[18] "Can Intelligent Hyperparameter Selection Improve Resistance to Adversarial Examples?" 2019.

[19] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," *Pattern Recognition Letters*, vol. 141, pp. 61–67, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865520302981

[20] L. Kanal, "Perceptron," in *Encyclopedia of Computer Science*. United Kingdom: John Wiley and Sons Ltd., Jan. 2003.

[21] Jaspreet, "A Concise History of Neural Networks," Aug. 2016. [Online]. Available: https://towardsdatascience.com/a-concise-history-of-neural-networks-2070655d3fec

[22] "Neural Networks History." [Online]. Available: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html

[23] Y. Chauvin and D. Rumelhart, *Backpropagation: Theory, Architectures, and Applications*, 1st ed. Hillsdale, N.J: Lawrence Erlbaum Associates, 1995.

[24] Y. LeCun and Y. Bengio, "Convolutional Networks for Images, Speech, and Time-Series," in *The handbook of brain theory and neural networks*. MA, U.S.A: MIT Press, 1998.

[25] X. Ying, "An Overview of Overfitting and its Solutions," *Journal of Physics: Conference Series*, vol. 1168, no. 2, 2019.

[26] H. Gholamalinezhad and H. Khosravi, "Pooling Methods in Deep Neural Networks, a Review," *ArXiv*, vol. abs/2009.07485, 2020.

[27] K. Weiss, T. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, no. 9, 2016.

[28] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and Its Various Variants," *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 415–419, 2018.

[29] M. Barreno, B. Nelson, R. Sears, A. Joseph, and J. D. Tygar, "Can Machine Learning Be Secure?" in *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '06. Taipei, Taiwan: Association for Computing Machinery, 2006.

[30] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks," 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb

[31] S.-l. Yin, X.-l. Zhang, and L.-y. Zuo, "Defending against adversarial attacks using spherical sampling-based variational auto-encoder," *Neurocomputing*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221019366

[32] X. Li and S. Ji, "Defense-VAE: A Fast and Accurate Defense Against Adversarial Attacks," vol. 1168, 2020.

[33] prakharroy, "Dilated Convolution," Mar. 2022. [Online]. Available: https://www.geeksforgeeks.org/dilated-convolution/

[34] "Keeping the Bad Guys Out: Protecting and Vaccinating Deep Learning with JPEG Compression," 2017.

[35] J. Jiang, B. Li, M. Yu, C. Liu, J. Sun, W. Huang, and Z. Lv, "AdvRefactor: A Resampling-Based defense Against Adversarial Attacks," in *Process Communication Model*, 2018, pp. 815–825.

[36] G. Kurtzer, V. Sochat, and M. Bauer, "Singularity: Scientific containers for mobility of compute," *PubMed*, vol. 12, no. 5, 2017.

[37] "Adversarial Robustness Toolbox v1.0.0," 2019.

[38] L. He, H. Wu, and X. Zhao, "The application of Lanczos interpolation in video scaling system based on FPGA," vol. 11719. Shanghai, China: SPIE, 2021, pp. 120 – 126. [Online]. Available: https://doi.org/10.1117/12.2589518

[39] Y.-T. Chou, S.-H. Chiou, and J.-F. Yang, "Image interpolation by using Gaussian regularized regression with cross-based window," 2015, pp. 1–4.

[40] O. Rukundo and H. Cao, "Nearest Neighbor Value Interpolation," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 4, pp. 25–30, 2012.

[41] J. Chao, "Understanding Bilinear Image Resizing," 2018. [Online]. Available: https://chao-ji.github.io/jekyll/update/2018/07/19/BilinearResize.html

[42] A. Krizhevsky, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[43] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, no. 0, pp. –, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608012000457

[44] "Deep Residual Learning for Image Recognition," Dec. 2015.

[45] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," San Diego, CA, USA, 2015.

[46] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: a simple and accurate method to fool deep neural networks," 2016, pp. 2574–2582.

[47] "Noise2Self: Blind Denoising by Self-Supervision," Jun. 2019.

[48] "A study of the effect of JPG compression on adversarial images," Aug. 2016.

[49] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," vol. 2, 2005, pp. 60–65.

[50] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples," in *Proceedings of Machine Learning Research*, 2018, pp. 274–283.

[51] M. Lister, "31 Mind-Boggling Instagram Stats & Facts for 2022," Jan. 2022. [Online]. Available: https://www.wordstream.com/blog/ws/2017/04/20/instagram-statistics

[52] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, vol. 32, pp. 323–332, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608012000457

[53] J. Bull, "Find Pareto Frontiers in Python," Jul. 2012. [Online]. Available: https://oco-carbon.com/metrics/find-pareto-frontiers-in-python/

Appendix A

TABLES

Table A.1: Hyperparameter selections of model training on the CIFAR-100 dataset

Attacking method	Model		
	EfficientNet-B0	ResNet-50	VGG-19
Layers to unfreeze	0	0	-20
optimizer learning rate	0.01	0.01	0.01
Pooling	avg	avg	avg
Dropout rate	0.2	0.3	0.2

Table A.2: Hyperparameter selections of model training on the CIFAR-10 dataset

Attacking method	Model		
	EfficientNet-B0	ResNet-50	VGG-19
Layers to unfreeze	0	-20	0
optimizer learning rate	0.001	0.001	0.001
Pooling	avg	avg	avg
Dropout rate	0.2	0.3	0.3

Table A.3: Hyperparameter selections of model training on the GTSRB dataset

Attacking method	Model		
	EfficientNet-B0	ResNet-50	VGG-19
Layers to unfreeze	0	0	0
optimizer learning rate	0.001	0.001	0.001
Pooling	avg	avg	avg
Dropout rate	0.3	0.2	0.3

Table A.4: Attacker setting

Attacking method	Overshoot parameter	
	CIFAR-100 and CIFAR-10 dataset	GTSRB dataset
PGD	0.85	1.2
FGSM	0.9	4.0
DeepFool	1.0	1.1

Table A.5: Performance of various intermediary sizes of EAE defending the EfficientNet-B0 classifier against evasion attacks in gray-box threat model on the CIFAR-100 dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	13.76	13.69	13.69	13.57
16	58.88	57.59	57.149	55.23
20	75.31	71.61	73.68	64.44
24	82.53	76.61	80.069	66.31
25	83.21	77.08	80.92	66.28
26	84.66	77.36	82.23	65.19
27	85.01	76.96	82.63	64.25
28	85.67	77.73	82.66	63.23
29	85.58	76.17	82.77	62.12
30	86.07	75.73	83.09	61.16
31	84.31	74.09	81.15	62.47
32	89.75	74.05	87.62	49.43
34	85.98	72.32	82.2	55.76
40	87.54	66.1	82.65	48.43
64	87.84	45.94	79.8	38.54
128	89.41	11.94	71.2	23.85

Table A.6: Performance of various intermediary sizes of EAE defending the ResNet-50 classifier against evasion attacks in gray-box threat model on the CIFAR-100 dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	13.11	12.86	13.15	13.11
16	46.82	45.16	46.21	44.71
20	59.7	56.11	58.71	54.87
24	67.89	61.37	66.3	60.2
25	69.29	62.47	67.71	61.68
26	70.79	63.01	69.04	62.76
27	72.15	63.08	70.26	63.49
28	73.68	63.58	71.73	63.22
29	73.65	62.17	71.36	63.59
30	73.67	62.03	71.35	63.25
31	72.81	61.11	70.67	63.21
32	78.33	53.01	74.47	57.11
34	75.23	57.93	72.44	61.61
40	76.68	51.23	72.69	57.81
64	77.57	30.65	71.06	49.61
128	78.32	6.26	62.09	35.47

Table A.7: Performance of various intermediary sizes of EAE defending the VGG-19 classifier against evasion attacks in gray-box threat model on the CIFAR-100 dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	11.61	11.61	11.64	11.61
16	43.75	42.82	43.71	42.65
20	57.56	52.55	55.84	48.94
24	66.72	56.48	63.8	49.04
25	68.59	58.31	65.76	54.64
26	69.32	58.03	66.71	53.07
27	70.46	57.34	67.48	52.75
28	71.49	57.19	68.35	52.83
29	71.78	55.91	68.72	51.99
30	72.51	55.51	68.98	51.86
31	71.11	55.98	67.91	55.09
32	76.58	45.78	72.51	37.77
34	73.22	51.31	69.01	49.07
40	74.12	44.53	68.46	44.65
64	74.81	27.74	65.57	40.17
128	76.54	6.83	51.84	28.14

Table A.8: Performance of various intermediary sizes of EAE defending the EfficientNet-B0 classifier against evasion attacks in gray-box threat model on the CIFAR-10 dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	17.98	17.63	17.75	17.59
16	70.72	68.65	69.53	65.18
20	83.54	79.67	81.64	69.64
24	89.39	83.72	87.39	70.46
25	89.8	84.13	87.9	68.78
26	90.46	84.35	88.68	68.62
27	90.98	84.83	89.56	67.17
28	91.43	85.01	89.65	67.8
29	91.39	83.91	89.56	64.33
30	91.38	82.4	88.71	63.65
31	91.55	83.14	89.29	67.78
32	94.34	83.19	92.75	49.2
34	91.15	79.65	88.13	57.81
40	92.7	75.85	89.57	49.72
64	93.18	58.34	87.22	40.66
128	94.16	17.37	79.94	24.72

Table A.9: Performance of various intermediary sizes of EAE defending the ResNet-50 classifier against evasion attacks in gray-box threat model on the CIFAR-10 dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	23.26	23.17	23.2	22.96
16	63.48	62.53	63.14	62.09
20	75.38	73.85	75.02	72.68
24	80.27	77.95	79.64	76.1
25	81.41	78.93	80.87	77.56
26	82.09	78.93	81.33	78.03
27	82.75	79.7	82.02	78.79
28	91.43	80.2	83.16	79.26
29	83.82	79.87	82.84	79.14
30	83.79	79.84	82.96	78.77
31	82.98	78.82	82	78.32
32	88.54	80.76	86.12	76.96
34	84.88	78.84	83.51	77.81
40	85.94	77.87	84.19	76.15
64	86.8	70.44	83.62	68.96
128	88.49	23.87	78.08	48.1

Table A.10: Performance of various intermediary sizes of EAE defending the VGG-19 classifier against evasion attacks in gray-box threat model on the CIFAR-10 dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	20.2	20.26	20.14	20.14
16	52.3	50.91	51.88	51.21
20	70.89	66.51	68.9	62.09
24	80.27	73.13	77.98	63.08
25	81.37	75.27	79.56	68.64
26	82.67	75.46	80.76	67.14
27	83.76	75.94	81.73	67.37
28	84.8	75	82.73	64.59
29	84.6	75.5	82.61	65.8
30	84.49	74.56	82.22	65.41
31	82.22	73.75	80.36	68.69
32	90.22	72.75	87.92	47.46
34	84.36	72.02	81.57	61.8
40	86.68	69.34	83.7	59.36
64	85.36	46.7	78.58	50.9
128	89.44	6.6	70.54	34.41

Table A.11: Performance of various intermediary sizes of EAE defending the EfficientNet-B0 classifier against evasion attacks in gray-box threat model on the GTSRB dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	13.96	13.78	13.64	13.47
16	79.74	67.74	78.61	40.2
20	91.77	67.69	89.72	75.44
24	94.98	80.07	90.68	71.93
25	95.76	80.71	94.6	72.89
26	95.35	87.26	94.5	73.1
27	95.87	87.8	94.4	73.01
28	95.57	88	94.24	70.76
29	95.83	88.19	93.98	72.09
30	96.58	88.89	95.38	70.03
31	96.02	88.31	94.24	72.27
32	96.77	87.32	95.99	65.13
34	96.56	86.6	95.06	66.63
40	96.64	82.85	95.01	64.29
64	96.73	69.31	94.02	58.61
128	96.8	59.9	70.12	57.89

Table A.12: Performance of various intermediary sizes of EAE defending the ResNet-50 classifier against evasion attacks in gray-box threat model on the GTSRB dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	14.03	13.73	13.4	13.42
16	79.61	75.59	75.6	67.41
20	92.61	76.15	89.72	74.73
24	94.52	75.74	93.12	73.04
25	94.44	75.13	92.88	72.51
26	94.8	76.12	93.25	71.75
27	94.81	78.8	93.4	75.9
28	95.08	79.5	93.31	76.52
29	94.79	82.76	93.12	70.52
30	95.12	81.62	94.16	69.07
31	95.03	80.43	92.89	70.29
32	94.97	82.98	93	73.15
34	95.05	77.96	94.2	69.47
40	95.65	73.43	94.33	69.55
64	95.57	60.74	95.57	68.86
128	96	40.5	60.9	68.5

Table A.13: Performance of various intermediary sizes of EAE defending the VGG-19 classifier against evasion attacks in gray-box threat model on the GTSRB dataset

Intermediary size	Natural F1 score	Robust F1 score		
		PGD	FGSM	DeepFool
8	14.03	13.73	13.4	13.42
16	79.61	75.59	75.6	67.41
20	92.61	76.15	89.72	74.73
24	94.52	75.74	93.12	73.04
25	94.44	75.13	92.88	72.51
26	94.8	76.12	93.25	71.75
27	94.81	78.8	93.4	75.9
28	95.08	79.5	93.31	76.52
29	94.79	82.76	93.12	70.52
30	95.12	81.62	94.16	69.07
31	95.03	80.43	92.89	70.29
32	94.97	82.98	93	73.15
34	95.05	77.96	94.2	69.47
40	95.65	73.43	94.33	69.55
64	95.57	60.74	95.57	68.86
128	96	40.5	60.9	68.5

Table A.14: Selected hyperparameter set of Autoencoder

	CIFAR-100	CIFAR-10	GTSRB
Dropout rate	0.25	0.25	0.25
Loss function	MSE	MSE	MSE
Number of filters	128	128	128
Optimizer	Adam	Adam	Adam
Optimizer learning rate	1e-05	1e-05	1e-05
Noise level	1e-3	0	0.03

Table A.15: Selected hyperparameter set of U-Net

	CIFAR-100	CIFAR-10	GTSRB
Dropout rate	0.5	0.5	0.5
Loss function	MSE	MSE	MSE
Start neurons	28	32	28
Optimizer	Adam	Adam	Adam
Optimizer learning rate	1e-3	1e-3	1e-3
Noise level	1e-3	0	0.03

Table A.16: Selected hyperparameter set of VAE

	CIFAR-100	CIFAR-10	GTSRB
Filters in layers	[64, 32, 16]	[64, 32, 16]	[64, 32, 16]
Latent dimension	1024	1024	1024
Loss function	MSE	MSE	MSE
Optimizer	Adam	Adam	Adam
Optimizer learning rate	1e-5	1e-5	1e-5
Noise level	0.03	0.03	0.03

Table A.17: Selected hyperparameter set of JPEG compression

	CIFAR-100	CIFAR-10	GTSRB
Image quality	23	23	23

Table A.18: Selected hyperparameter set of TVM

	CIFAR-100	CIFAR-10	GTSRB
Probability of the Bernoulli distribution	0.3	0.3	0.3
Solver	L-BFGS-B	L-BFGS-B	L-BFGS-B

Table A.19: Selected hyperparameter set of Non-local mean

	CIFAR-100	CIFAR-10	GTSRB
Fast mode	True	True	True
Patch distance	11	11	11
Cut off distance	0.1	0.1	0.1

Table A.20: Selected hyperparameter set of Noise2self

	CIFAR-100	CIFAR-10	GTSRB
Calibrated method	[bilinear, nearest]	[bilinear, nearest]	[bilinear, nearest]
Intermediary size	[8, 16, 28, 32]	[8, 16, 28,32]	[8, 16, 28, 32]