

Exploring the Back Alleys: Analysing The Robustness of Alternative Neural Network Architectures against Adversarial Attacks

Yi Xiang Marcus Tan,^{*,†} Yuval Elovici,^{*,†}, and Alexander Binder^{*,‡}

^{*}ST Engineering Electronics-SUTD Cyber Security Laboratory

[‡]Information Systems Technology and Design (ISTD) Pillar, Singapore University of Technology and Design

[†]Department of Software and Information Systems Engineering, Ben-Gurion University of the Negev

[‡]Deutsche Telekom Innovation Laboratories at Ben-Gurion University of the Negev

Abstract—Recent discoveries in the field of adversarial machine learning have shown that Artificial Neural Networks (ANNs) are susceptible to adversarial attacks. These attacks cause misclassification of specially crafted adversarial samples. In light of this phenomenon, it is worth investigating whether other types of neural networks are less susceptible to adversarial attacks. In this work, we applied standard attack methods originally aimed at conventional ANNs, towards stochastic ANNs and also towards Spiking Neural Networks (SNNs), across three different datasets namely MNIST, CIFAR-10 and Patch Camelyon. We analysed their adversarial robustness against attacks performed in the raw image space of the different model variants. We employ a variety of attacks namely Basic Iterative Method (BIM), Carlini & Wagner L2 attack (CWL2) and Boundary attack. Our results suggests that SNNs and stochastic ANNs exhibit some degree of adversarial robustness as compared to their ANN counterparts under certain attack methods. Namely, we found that the Boundary and the state-of-the-art CWL2 attacks are largely ineffective against stochastic ANNs. Following this observation, we proposed a modified version of the CWL2 attack and analysed the impact of this attack on the models' adversarial robustness. Our results suggest that with this modified CWL2 attack, many models are more easily fooled as compared to the vanilla CWL2 attack, albeit observing an increase in L_2 norms of adversarial perturbations. Lastly, we also investigate the resilience of alternative neural networks against adversarial samples transferred from ResNet18. We show that the modified CWL2 attack provides an improved cross-architecture transferability compared to other attacks.

I. INTRODUCTION

Second generation neural networks have been empirically successful in solving a plethora of tasks. The different variants of Artificial Neural Networks (ANN) have been used in applications namely providing authenticating services [1], detecting anomalous behaviours in cyber-physical systems [2], image recognition [3], or simply playing a game of Go [4].

However, in 2013, the first research was performed showing that ANNs were vulnerable to adversarial attacks [5], a phenomenon that involves the creation of specially perturbed samples from their original counterparts, imperceptible upon visual inspection, which can be misclassified by ANNs. Since then, many researchers introduced other adversarial attack methods against such ANN models, whether under a white-

box [6]–[10] or a black-box [11], [12] scenario. This raises questions about the reliability of ANNs, which can be a cause for concern especially when used in cyber-security or mission critical contexts. [13], [14].

Recently, a third generation of neural networks from the field of computational neuroscience, namely Spiking Neural Networks (SNNs), has been researched upon as a means to model the biological properties of the human brain more closely as compared to their second generation ANN counterparts. In contrast to ANNs, SNNs train on spike trains rather than image pixels or a set of predefined features. There have been different variants of SNNs, differing in terms of the learning rule used (whether through standard backpropagation [15]–[17] or via Spike-Timing-Dependent Plasticity (STDP) [18]–[20]) or the architecture. In this work, we focused on the STDP-based learning variant of SNNs.

Stochastic ANNs have also been used to perform image classification tasks. In this work, we focused on two sub-categories of such stochastic ANNs, one involving making both its hidden weights and activations are in a binary state [21], while the other only requiring its hidden activations to be binary [22]–[24]. These variants of networks use Bernoulli distributions in order to binarize its features.

Since there are strong evidences showcasing the weaknesses of ANNs to adversarial attacks, we question if there exists alternative variants of neural networks that are inherently less susceptible to such a phenomenon. In this work, we have decided to turn our attention to analysing the resilience of both SNN and stochastic ANN variants against adversarial attacks.

The authors in [25] gave a preliminary study of investigating the adversarial robustness of two variants of SNNs that employ the use of gradient backpropagation during training, namely ANN-to-SNN conversion [16] and also Spike-based training [15]. The authors examined the robustness of the SNNs, and also a VGG-9 model in the white-box and black-box settings. They concluded that SNNs trained directly on spike trains are more robust to adversarial attacks as compared to SNNs converted from their ANN counterparts. However, in their experiments, the authors performed their attacks on intermediate

spike representations of images, which is the result of passing images through a Poisson Spike Generation phase followed by rate computation. Though their work shows preliminary results on the robustness of SNNs, we find that their simplified approach of constructing adversarial samples yields unrelatable deviations between the natural and their adversarial counterparts in the image space. Also, they investigate variants of SNNs that are trained via backpropagation. We attempt to address those points in our work, by focusing on STDP-based learning SNNs and also constructing adversarial samples in the input space. To the best of our knowledge, we did not find prior work examining the adversarial robustness of networks employing the use of BSNs, though there exists works that explored adversarial attacks against Binary Neural Networks (BNNs) [26], [27]. The authors in [26] performed two white-box attacks and a black-box attack (the Fast Gradient Sign Method (FGSM) [6], CWL2 and the transferability from a substitute model procedure proposed by [28]) and showed that stochasticity in binary models do improve the robustness against attacks.

Unlike [25], we examined two very recent works in the field of SNN: the Multi-Class Synaptic Efficacy Function-based leaky-integrate-and fire neuRON (MCSEFRON) model [20] and Reward-modulated STDP spike-timing-dependent plasticity in deep convolutional network proposed in [29]. For the remaining of this paper, we would like to refer to the latter model as SNN_m for notation simplicity. For our stochastic ANN variants, we used Binary Stochastic Neurons (BSN) to give our models binarized activations in a stochastic manner. Also, we used Binarized Neural Networks (BNN) that binarizes weights and activations as our second variant of the stochastic ANN. We used the vanilla ResNet18 model as a bridge across the different variants of neural networks.

The contributions of this work are as follows:

- 1) We analyse to what extent conventional adversarial attacks (white-box and black-box) can be performed in the original image space against SNNs with different information encoding schemes. This is of interest as it includes networks not trained with backpropagation.
- 2) We shed light on the effectiveness of adversarial attacks against stochastic neural network models. In order to provide a reasonable comparison across the models, we employed the vanilla ResNet18 CNN as a baseline.
- 3) We propose an augmented version of a state-of-the-art white-box attack, CWL2, and analyse the robustness of the different network variants to samples generated via such attacks.
- 4) We investigate the susceptibility of alternative variants of neural networks are against transferred adversarial samples across architectures, constructed from ResNet18.
- 5) As a last novel contribution we measure the efficiency of attacks against stochastic mixtures of different architectures. Given the availability of different variants of neural networks, a stochastic mixture of them is an imaginable defense mechanism which does not rely on detecting adversarials.

The remaining of our paper is organised in the following manner. We start off with details of the attacks we used, also providing brief introduction to SNNs and stochastic ANNs in Section II. In Section III, we discuss our experimental setup and also our findings. This is followed by a discussion of attacking stochastic architecture mixtures in Section IV, should such a defence mechanism be employed. After which, in Section V, we provide some discussion points with regards to stochastic ANNs. Finally, we conclude our work in Section VI.

II. BACKGROUND

A. Adversarial Attacks Against Neural Networks

The concept of adversarial examples were first introduced by [5]. The authors demonstrated that misclassification of ANNs were possible by adding a set of specially crafted perturbations to an image, albeit imperceptible upon visual inspection. Following their work, several other researchers explored various methods to launch adversarial attacks in an attempt to further evaluate the robustness of ANNs. One of which was the FGSM, where it uses the sign of the gradients that were computed from the loss with respect to the input space, to perform a single-step perturbations on the input itself. They adopted the same loss function that was used to train the image classifier to obtain the gradients. Several studies [7], [8] extended this attacking technique by applying the algorithm to the input image sample for multiple iterations to construct a stronger adversarial sample. Currently, however, the Carlini & Wagner (CW) attack [10] is the state-of-the-art white-box adversarial attack method, capable of producing misclassified and visually imperceptible images, that manage to make *defensive distillation* [30] ineffective against adversarial attacks.

The methods described above and many other methods proposed by the scientific community [9], [31]–[33] pertain to attacks done in a white-box setting, in which it is assumed that the attacker has full knowledge and access to the ANN image classifier. However, several researchers [11], [12] have also shown that it is also possible to attack a model, without the need of any knowledge of the targeted model (i.e. black-box attacks). In [11], the authors used the decision made by the targeted image classifier to perturb the input sample. In [12], the authors made use of the concept of transferability of adversarial samples across neural networks to attack the victim classifier. Their method is a two-step process, which first involves approximating the decision boundary of the targeted classifier by training a surrogate model to convert a black-box to a white-box problem. Next, they attack the surrogate model in a white-box fashion thereafter launching the resultant adversarial sample towards the targeted classifier. In the next section, we describe the attacks we used in our work, exploring both the white-box and black-box categories.

B. Attack Algorithms Used

To attack the model in a black-box setting, we used a decision-based method known as *Boundary Attack* [11]. This approach initialises itself by generating a starting sample that

is labelled as adversarial to the victim classifier. Following which, random walks are taken by this sample along the decision boundary that separates the correct and incorrect classification regions. These random walks will only be considered valid if it fulfils two constraints, i) the resultant sample remains adversarial and ii) the distance between the resultant sample and the target is reduced. Essentially, this approach performs rejection sampling such that it finds smaller valid adversarial perturbations across the iterations.

We used the *Basic Iterative Method* (BIM) [8] as one of the means to perform white-box attacks. This method is basically an iterative form of the FGSM attack which is represented as such:

$$x_{t+1} = x_t + \alpha * \text{sign}(\nabla J(F(x_t), y; \theta)) \quad (1)$$

where ∇J represents the gradients of the loss calculated with respect to the input space x_t and its original label y , t represents the iterations. This approach takes the sign of the gradients, multiply it with a scaling factor α , and adding this perturbation to the sample at the t^{th} iteration.

The CW attack is a targeted attack strategy, in which an objective function is optimised such that it yields a resultant imperceptible image, while being labelled as an adversarial class by the targeted image classifier. This image would then be used to cause misclassification. More specifically, the adversary has to solve the following objective function:

$$\min_{\delta} \|\delta\|_2 + c \cdot f(x + \delta) \quad (2)$$

where the first term minimises the L_2 norm of the perturbation while the second term ensures misclassification. c is a constant. This attack method is considered as state-of-the-art and can still be used to bypass several detection mechanisms [34].

C. Spiking Neural Networks

1) *MCSEFRON*: MCSEFRON [20] is a two-layered SNN that has time-dependent weights connecting between neurons. It adopts the STDP learning rule and it trains based on variations between the relative timings between the actual and desired post-synaptic spike time. It encodes images into spike trains via the same mechanism as [35], which involves projecting the real-valued normalised image pixels (in [0,1]) onto multiple overlapping receptive fields (RF) represented by Gaussians. After the training is done, it makes decisions based on the earliest post-synaptic spikes while ignoring the rest.

2) *SNN_M*: The Reward-modulated STDP (R-STDP) in deep convolutional networks [29], referred to as SNN_M, makes use of three convolution layers, with the first two trained in an unsupervised manner via STDP and the last convolution trained via Reward-modulated STDP. The input images had to be first preprocessed by six Difference of Gaussian (DoG) filters, which were followed by the encoding into spike trains by the intensity-to-latency [36] scheme. The SNN_M does not require any external classifiers as they used a neuron-based decision-making trained via R-STDP in the final convolution layer. The R-STDP is based on reinforcement learning concepts, where correct decisions will lead to STDP while incorrect decisions will lead to anti-STDP.

III. EXPERIMENTS AND RESULTS

We used three datasets for our experiments, namely MNIST [37], CIFAR-10 [38] and, Patch Camelyon [39] which we refer to as PCam. The libraries we used in our experiments are PyTorch [40] and SpykeTorch [41] for constructing our image classifiers. For attacks, we used the Foolbox [42] library at version 1.8.0.

A. Image Classification Baseline

In this work, we explored eight different variants and architectures of neural networks: ResNet18, MCSEFRON, SNN_M, three BSN architectures and two BNN architectures. The BSN architectures used are a 2-layered, 4-layered Multilayer Perceptron, and a modified LeNet [43] which we will refer to as BSN-2, BSN-4 and BSN-L respectively. For the BNNs, we explored both deterministic and stochastic binarization strategies, which we will refer to as BNN-D and BNN-S respectively.

1) *Training the Classifiers*: For the ANN, we used the ResNet18 [44] from PyTorch's torchvision. We would like to refer the reader to our supplementary materials for more details regarding the hyperparameters we used for this model and also for the other variants, that will be discussed in the paragraphs below within this section.

For the case of MCSEFRON, we used five receptive fields (RFs) and a learning rate of 0.1 for MNIST while using three RFs and a learning rate of 0.5 for CIFAR-10. The other hyperparameters were set at their default values. We used the authors' implementation of MCSEFRON in Python¹ for training. In training MCSEFRON, we performed sub-sampling strategies on the training data. We used the first batch of training data of CIFAR-10; we used the first 30000 samples of PCam.

As mentioned in Section II, in the case of SNN_M, the model's input images are preprocessed by the DoG filters. The number of DoG filters used will determine the input channel of the first convolution layer in SNN_M. Hence, for a three-channelled image (e.g. CIFAR-10), we first take the mean of the channels to convert the images to a single channel, prior to passing them to the DoG filters. Unfortunately for this model, we could not find a suitable set of hyperparameters that performs reasonably on the PCam dataset. While training, we noticed that the outputs of the network was consistently the same, regardless of the number of training iterations. Hence, we could not report the Adversarial Success Rates (ASRs) and their respective norms for the attacks against SNN_M using the PCam dataset.

For BSNs, we used a batch size of 128 and used Adam optimizer for the BSN-4 and BSN-L variants while using Stochastic Gradient Descent (SGD) for BSN-2. The other hyperparameters we used can be found in the supplementary material. We adapted the code from this GitHub repository², with the network definition of the BSN-L architecture in

¹<https://github.com/nagadarshan-n/MC-SEFRON>

²<https://github.com/Wizaron/binary-stochastic-neurons>

PyTorch requiring modification on all intermediate activations with BSN modules.

For the BNNs, we used the same hyperparameters across the various datasets and models and adapted the code from this GitHub repository³, which was originally used by the authors in [21]. We used a learning rate of 0.005 and weight decay of 0.0001 with a batch size of 256. We also used the Adam optimiser to train our models for 20 epochs in MNIST, 150 epochs in CIFAR-10 and 50 epochs in PCam. We manually set the learning rate to 0.001 at epoch 101 and 0.0005 at epoch 142, following the authors in [21]. For BNN-D and BNN-S, we used the ResNet18 architecture as the structure of the network, while the binarization of the weights and activations will only occur at the forward pass.

2) *Baseline Classification Performance*: The baseline image classification performances are summarised in Table I. It is evident that these results are not state-of-the-art. However, getting the most optimal performance is not the focus of this work. Having said that, we would like to highlight the accuracy obtained for MCSEFRON on the CIFAR-10 dataset. We hypothesise that the reason behind the significantly poor performance is due to the inherent architecture of the model. As MCSEFRON can be considered as a single layered neural network without any convolution layers, its performance is highly limited on more complex image datasets, like CIFAR-10. In a prior work that studied the performance limitations of models without convolutions [45], they managed to obtain an accuracy of only approximately 52% to 57% on CIFAR-10, using a deeper and more dense fully-connected neural network (see Figure 4(a) in [45]).

TABLE I: Baseline image classification performances of the various models. Metrics reported refers to the classification accuracy.

	MNIST	CIFAR-10	PCam
Resnet18	0.988	0.842	0.789
MCSEFRON	0.861	0.372	0.671
SNN _M	0.964	0.391	-
BSN-2	0.958	0.489	0.723
BSN-4	0.968	0.535	0.735
BSN-L	0.981	0.582	0.779
BNN-D	0.989	0.876	0.798
BNN-S	0.967	0.687	0.744

B. Modifying SNN implementation for Adversarial Attacks

As SNNs are inherently very different from conventional ANNs, there is a need to adapt the original implementation of the SNNs to fit our purposes. We made two modifications in our work. First, because there might be instances in which non-differentiable operations were performed (i.e sign function), when adapting such SNNs for our use, we replaced the built-in sign functions with our custom sign function, which performs the same operation but allows gradients to pass through in a straight through fashion in the backward pass. This ensures that the gradients are non-zeros everywhere. Also, since we

examined SNNs that were trained via STDP, such a change does not violate the learning rule of the SNNs. Furthermore, as we are only interested in the behaviour of such models when faced with adversarial samples, we extracted the critical parts of the network (i.e decision-making forward pass) only in our adaptation.

Secondly, as SNNs make decisions based on either earliest spike times or maximum internal potentials, their outputs are more commonly a single valued integer, depicting the predicted class. However, for attacks to be done on such networks, we require logits of networks. Hence, we simulated logits in our modification by using the post-synaptic spike times for the case of MCSEFRON and the potentials for the case of SNN_M for all of the classes. When spike times were used, we took the negative of spike times so that the max of the vector of spike times correspond to the actual prediction.

C. White-box Attacks Against Neural Networks

We report the proportion of adversarial samples that are successful in causing misclassification and term it as Adversarial Success Rate (ASR; in range [0,1]). Furthermore, we report the mean L_2 norms per pixel of the differences between natural images and their adversarial counterparts. We derived that metric by dividing the L_2 norm by the total number of pixels in the image. In our experiments, we sub-sampled 500 samples from the test set of the respective datasets during the evaluation of the BIM attack and 100 samples for the evaluation of the other attacks. We performed sub-sampling due to the computational intractability of performing the attacks on the entire dataset. Note that we only selected samples that were originally classified correctly while ignoring the rest.

1) *Basic Iterative Method (BIM)*: For the BIM attack, we varied the attack strength (symbolised by ϵ measured in L_∞ space) while keeping the step sizes and iterations fixed at 0.05 and 100 respectively. We explored ϵ values of 8/255, 16/255 and 32/255 in our experiments, showing the results of $\epsilon = 32/255$ while the rest can be found in our supplementary materials.

For an initial sanity check, one may inspect Figure 1. The BIM attack has one parameter, attack strength ϵ . One can observe an intuitively reasonable trade-off of adversarial success rate (ASR) versus L_2 norm of the distance of the adversarial samples to the original inputs, as the ϵ values vary according to $\epsilon = 8/255, 16/255, 32/255, 64/255, 96/255, 126/255$.

Two notable observations can be made about BIM from Tables IIa and IIb: Firstly, when comparing vulnerability of different networks against BIM, spiking neural networks, with the exception of MCSEFRON on CIFAR-10, tend to be the most robust. Secondly, when comparing attacks for a given architectures, BIM yields the highest ASR on binarized stochastic networks of all attacks, however this is achieved at the cost of L_2 -norms which are multiples of all other methods.

2) *Carlini & Wagner L2 (CWL2)*: For the CWL2 attack, we used the default attack parameters as specified in Foolbox. Exemplified by the results from ResNet18 in Table IIa, the CWL2 attack is an extremely powerful attack that manages

³<https://github.com/itayhubara/BinaryNet.pytorch>

TABLE II: Adversarial success rate (Table (a)) and mean L_2 norms per pixel (Table (b)) for the attacks. For stochastic ANNs, an average of five runs were taken. ϵ is the attack strength for BIM attack. 500 samples were taken for BIM attacked experiments while 100 samples were sampled in non-BIM attacked experiments. The default attack parameters were used as defined by Foolbox except for the case of ModCWL2, where K in Equation 3 was defined as 50.

(a) ASR (in [0,1]) of the different variants of models.

Dataset	Attack Method	Resnet18	SNN _M	MCSEFRON	BSN-2	BSN-4	BSN-L	BNN-D	BNN-S
MNIST	BIM	1.000	0.120	0.294	0.774	0.874	0.506	1.000	0.566
	CWL2	0.970	0.620	0.420	0.204	0.180	0.010	0.980	0.030
	ModCWL2	1.000	1.000	1.000	0.334	0.308	0.232	1.000	0.370
	Boundary	1.000	1.000	1.000	0.008	0.014	0.012	0.980	0.030
CIFAR-10	BIM	1.000	0.694	0.998	0.981	0.955	0.884	1.000	0.953
	CWL2	1.000	0.990	0.990	0.402	0.200	0.234	1.000	0.142
	ModCWL2	1.000	1.000	1.000	0.528	0.226	0.230	1.000	0.188
	Boundary	1.000	1.000	1.000	0.290	0.182	0.192	0.944	0.114
PCam	BIM	1.000	-	0.534	0.920	0.912	0.772	0.974	0.910
	CWL2	1.000	-	0.280	0.168	0.112	0.102	0.92	0.126
	ModCWL2	1.000	-	0.800	0.138	0.144	0.152	0.930	0.114
	Boundary	0.730	-	1.000	0.102	0.068	0.080	0.190	0.000

(b) Mean L_2 norms per pixel between the original image and its perturbed adversarial image of the different variants of models. Note that the values reported have been scaled up by a factor of 1000, purely for illustration purposes only.

Dataset	Attack Method	Resnet18	SNN _M	MCSEFRON	BSN-2	BSN-4	BSN-L	BNN-D	BNN-S
MNIST	BIM	2.1667	2.4142	1.4126	2.6164	2.2969	2.5716	1.5606	2.1711
	CWL2	0.9057	3.5731	0.5137	2.7403	2.5473	1.0335	0.0000	2.2963
	ModCWL2	5.8529	7.7747	4.6039	8.6806	9.0173	9.5389	0.2909	9.9991
	Boundary	1.3986	10.7922	3.4964	1.2268	1.3307	0.8380	0.0000	0.1485
CIFAR-10	BIM	0.9318	1.3968	0.8924	1.1667	0.9829	1.0891	0.9606	1.0494
	CWL2	0.0782	0.5601	0.0724	0.0874	0.0187	0.0059	0.0376	0.0507
	ModCWL2	0.1102	0.4354	0.0766	0.9369	0.0244	0.0163	0.0640	0.0590
	Boundary	0.1346	2.3423	2.2432	1.9463	0.0000	0.0776	1.1771	0.1411
PCam	BIM	0.9794	-	1.4248	0.9110	0.9179	1.0017	1.0343	0.9888
	CWL2	0.0870	-	0.7915	0.0954	0.2124	0.1738	0.1001	0.2636
	ModCWL2	0.1367	-	3.2671	0.0535	0.1422	0.2458	0.1384	0.2213
	Boundary	0.0856	-	3.1918	0.2154	0.0146	0.0013	2.0002	-

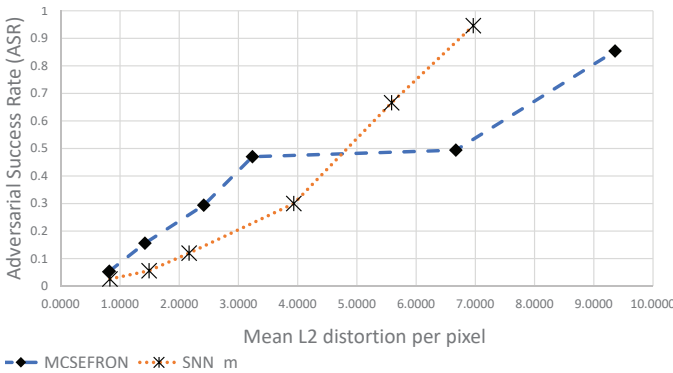


Fig. 1: Plot of ASR against the mean L_2 distortion per pixel when varying ϵ values on the MNIST dataset using the BIM attack. Targeted models were MCSEFRON and SNN_M. The mean L_2 distortion per pixel has been scaled up by a factor of 1000 for illustration purposes only.

to fool the model almost all of the time. However, this attack is not very effective against stochastic ANNs. As shown in Table IIa, stochastic ANNs only has a maximum ASR of 0.402 on the CIFAR-10 dataset for the BSN-2 model and a minimum of 0.01 ASR for BSN-L model on MNIST. Although this attack method is known to be state-of-the-art in generating successful adversarial samples with the least perturbation,

its efficacy drops significantly when faced with such model variants.

D. Black-box Attacks Against Neural Networks

1) *Boundary Attack*: The results in Table IIa shows that the effectiveness of the attack does not differ greatly among susceptible models, likewise among less susceptible models. Interestingly, the Boundary attack performs exceptionally well in terms of ASR against deterministic models, i.e. ResNet18, SNNs and BNN-D. Whereas for the stochastic ANNs, this attack method is much less efficient in finding adversarial samples. It even failed to find any for the case of BNN-S for the PCam dataset. This observation indicates that the attack method does not depend greatly on the architecture of the model but instead, on the nature of the model.

In the case of deterministic models, the decision boundary remains stable after training due to its fixed weights and activations for the same input sample. On the other hand, for stochastic ANNs, its weights and activations will vary based on a probability distribution, resulting in slightly varied predictions for the same sample at different times. Having a stochastic decision boundary will compromise the ability to obtain accurate feedback for the traversal of adversarial sample candidates which explains the poor performance of this attack.

E. Augmented Carlini & Wagner L2 Attack Against Neural Networks

Given the relatively poor ASR obtained by CWL2 and Boundary attacks against stochastic ANNs, we wonder whether a potential attacker may utilise randomness in augmenting input samples in the attack procedure to create attacks which result in samples further away from the decision boundary and thus are able to mislead stochastic ANNs. Recall that the CWL2 attack involves solving the objective function as defined in Equation 2. We modify this function to include an additional term that performs random augmentations on the input image, both rotations and translations, and then optimising it. Equation 3 formulates our modified attack, ModCWL2.

$$\min_{\delta} \|\delta\|_2 + c \cdot f(x + \delta) + \frac{1}{K} \sum_{i=1}^K f(R(x + \delta)) \quad (3)$$

where K is the number of iterations to perform random transformations, symbolised by $R(\cdot)$, on the input sample. Our function $R(\cdot)$ involves first making random rotations followed by random translations. In this work, we defined the allowable range of rotation angles to 180 degrees clockwise and counterclockwise, sampled from a uniform distribution. Also, we select at random the translation direction and pixels (integer from 0 to 10) to be applied on the image.

This modification will induce a trade-off between resultant L_2 norms and ASR. One can understand it in the following way: performing K times random transformations R will turn a single sample x into a cluster of K samples. Moving the cluster as a whole over the decision boundary requires a larger step than moving a single sample, depending on the radius of the cluster.

Figure 2 illustrates a boxplot of the CWL2 and ModCWL2 attacks' ASR and norms. The ModCWL2 is more consistent than CWL2 in achieving a higher ASR, based on the lower Inter-Quartile Range (IQR) and a much higher median and mean for ModCWL2, across the targeted models. More specifically, the IQR of the ASR of the CWL2 and ModCWL2 attacks are 0.82 and 0.772 respectively. The ModCWL2 attack has a higher median of 0.528 as compared to CWL2 with median 0.28. However, it is clear that the difference between the original and adversarial samples is much greater and more varied for the case of ModCWL2. The IQR of the norms of the CWL2 and ModCWL2 attacks are 0.0773 and 0.514 respectively. The ModCWL2 attack has a slightly higher median of 0.0246 as compared to CWL2 with median 0.0174.

In total, out of 12 configurations of stochastic networks in Table IIa, ModCWL2 performs better in 9 configurations and worse in 3 configurations compared to CWL2 in terms of ASR.

F. Transferability of Adversarial Samples

In this section, we discuss the transferability of adversarial samples derived from the vanilla ResNet18 to other architectures. This is a plausible scenario, arising when the attacker

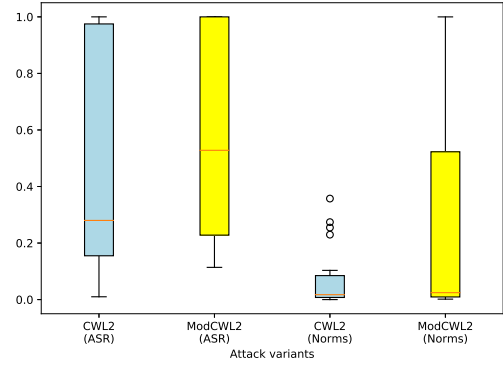


Fig. 2: Boxplot of the ASR and mean L_2 norms per pixel that we obtained in our experiments (taken from Tables IIa and IIb). Note that the L_2 norms reported here are scaled up by a factor of 100.

chooses a CNN (i.e. ResNet18) as target for adversarial attacks, since it is the most commonly used neural network variant. He or she then generates adversarial samples from the CNN, and launches them against the actual target model which is based on a different architecture. In this work, we evaluate this transferability phenomenon on the MNIST dataset as their corresponding baseline classification models achieved the lowest test error rates and it is the common dataset that is applicable across all models. We chose a subset of network variants instead of the full range of models in this set of experiments as we ignored repetitive variants and also variants already highly susceptible to the standard mode of attacks.

We draw the following observations based on Table III. Firstly, we observe highest transferability rates for MCSE-FRON and, in particular, for BNN-S. For the latter, one may postulate that it is due to the similar base architectures between BNN-S and ResNet18 as BNN-S uses ResNet18 as a structure while replacing components with binarized and stochastic counterparts.

Secondly, for SNN_M and BSN-L model variants and attack types not including ModCWL2, the success rate is low, thereby showing a certain robustness of SNN_M and BSN-L against direct transfer attacks. We consider it an important contribution of our study, demanding further investigation.

A third observation is that ModCWL2 performs well across all architectures when compared to the other attacks. This result shows another strength of ModCWL2. Only with BNN-S, it is clearly outperformed by BIM.

IV. ATTACKING STOCHASTIC ARCHITECTURE MIXTURES

In the previous sections, we observed that several network architectures appear to be moderately robust against transferability attacks. Inspired by this, a defender could employ stochastic switching of a mixture of neural networks with differing architectures to circumvent adversarial attack attempts. To do this, at inference time the defender chooses at random a neural network to be used to evaluate the input sample. This is a special case of drawing a distribution over

TABLE III: Transferability rate of the resultant adversarial samples generated from ResNet18 using various attack types on MNIST. Only adversarial samples successful against ResNet18 were considered. A higher rate indicates a more successful misclassification attempt of the generated adversarial samples from the ResNet18 transferred to the respective targeted models.

Attack method	ϵ	MCSEFRON	SNN _M	BSN-L	BNN-S
BIM	8/255	0.2190	0.0000	0.0556	0.3450
	16/255	0.3210	0.0101	0.0453	0.3560
	32/255	0.3200	0.0080	0.0600	0.4460
	64/255	0.3380	0.0060	0.0480	0.4340
	128/255	0.3380	0.0060	0.0760	0.4500
CWL2	-	0.0211	0.0000	0.0206	0.0938
ModCWL2	-	0.3100	0.2600	0.2600	0.2800
Boundary	-	0.2000	0.0000	0.0100	0.2200

TABLE IV: ASR (in [0,1]) against selected ensemble of architectures for the BIM attacks taken at $\epsilon = 32/255$, for both MNIST and CIFAR-10. 100 samples were sampled in each experiment.

	ResNet18 + BSN-L	ResNet18 + BNN-S	ResNet18 + BSN-L + BNN-S
MNIST	0.17	0.08	0.12
CIFAR-10	0.78	0.78	0.64

networks from e.g. a Dirichlet prior. We explore three different selected combinations of ensembles, 1) ResNet18 with BSN-L, 2) ResNet18 with BNN-S, and 3) ResNet18 with BSN-L and BNN-S. Here, we investigate the ASR in attacking against such ensembles. In our experiments, we applied the BIM attack due to its good performance against stochastic networks, using the mean of the gradients with respect to the input across the ensemble of models, with an attack strength chosen at $\epsilon = 32/255$. This is inspired by [46]. While they considered ensembles of CNNs, we explore a stochastic mixture of differing architectures.

One can compare the results from Table IV for MNIST against Table III. Table III permits to estimate the ASR of a transferability attack against a stochastic mixture. For example, using a transferability attack against a (0.5,0.5)-mixture of ResNet18 and BSN-L would result in an ASR of $0.5 \cdot 1 + 0.5 \cdot 0.06 = 0.53$. Surprisingly we can see that directly attacking stochastic mixtures seems to perform poorly, at least for MNIST. As for BSN-L, transferability would result in an ASR of $0.5 \cdot 1 + 0.5 \cdot 0.446 = 0.723$, which is much better than the observed 0.12 in Table III. This raises the question whether such robustness of stochastic mixtures holds also for other datasets and larger neural networks, or whether more efficient attacks can be designed against stochastic mixtures.

V. DISCUSSION

One notable observation is that stochastic networks are almost equally very vulnerable as CNNs, when BIM is used with sufficient strength. It is the simplest of all considered attacks. Its advantage for stochastic networks is that it does not attempt to stay close to the decision boundary as explicitly enforced in boundary attacks, and implicitly enforced by CWL2 attacks where the regulariser term attempts to keep the adversarial close to the initial sample. For stochastic networks

the decision boundary is defined only in an expected sense. Staying close to expected decision boundary results in a higher failure rate of adversarials. The simplicity of BIM allows it to take larger steps across the expected decision boundary.

Another observation is that transferability across architectures is limited, which calls for further investigation of non-averaged combination of different architectures.

VI. CONCLUSION

We performed adversarial attacks on a wide variety of models (e.g. SNNs, BSNs, BNNs), across different datasets namely MNIST, CIFAR-10 and PCam in the raw input image space, with the goal of investigating the adversarial robustness of alternative variants of neural networks. We note that there exists alternative variants of neural networks (i.e. stochastic ANNs) that are vulnerable to the simple BIM and moderately robust against more elaborate adversarial attacks than conventional ANNs. It is a partially positive result that stochastic networks are more robust against elaborate attacks. Unfortunately, detecting a stochastic network by its outputs is trivial.

Given the above, we were motivated to modify a state-of-the-art CWL2 attack, in order to investigate the robustness of such models against this modified attack. We found that our modification do increase the ASR against such model variants substantially, though incurring higher L_2 norms in adversarial perturbations. We also analysed the hypothetical scenario whereby the attacker is unsure of the targeted image classifier and thus attempt a transferability attack based on a conventional ANN (i.e. ResNet18). We found that such an attack strategy would be highly ineffective, if there is an architecture mismatch between the source and target models. Finally, we question the success of adversarial attacks should an ensemble utilising a stochastic switch of networks for inference be employed, and found that though ASR do decrease, the change in MNIST is more pronounced than that of CIFAR-10, which calls for further investigation.

VII. ACKNOWLEDGEMENTS

This work was supported by both ST Electronics and the National Research Foundation (NRF), Prime Ministers Office, Singapore under Corporate Laboratory @ University Scheme (Programme Title: STEE Infosec-SUTD Corporate

Laboratory). Alexander Binder also gratefully acknowledges the support by PIE-SGP-AI-2018-01.

REFERENCES

- [1] P. Chong, Y. Elovici, and A. Binder, "User authentication based on mouse dynamics using deep neural networks: A comprehensive study," *IEEE Transactions on Information Forensics and Security*, 2019.
- [2] J. Goh, S. Adepu, M. Tan, and Z. S. Lee, "Anomaly detection in cyber physical systems using recurrent neural networks," in *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, Jan 2017, pp. 140–145.
- [3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.
- [5] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.
- [6] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," pp. 1–11, 2014. [Online]. Available: <http://arxiv.org/abs/1412.6572>
- [7] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.
- [8] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.
- [9] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.
- [10] N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," *Proceedings - IEEE Symposium on Security and Privacy*, pp. 39–57, 2017.
- [11] W. Brendel, R. Jonas, and B. Matthias, "Decision-Based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models," in *ICLR*, 2018, pp. 1–12. [Online]. Available: <https://openreview.net/forum?id=SyZi0GWZ>
- [12] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*. ACM, 2017, pp. 506–519.
- [13] Y. X. M. Tan, A. Iacovazzi, I. Homoliak, Y. Elovici, and A. Binder, "Adversarial attacks on remote user authentication using behavioural mouse dynamics," *arXiv preprint arXiv:1905.11831*, 2019.
- [14] I. Rosenberg, A. Shabtai, Y. Elovici, and L. Rokach, "Low Resource Black-Box End-to-End Attack Against State of the Art API Call Based Malware Classifiers," 2018. [Online]. Available: <http://arxiv.org/abs/1804.08778>
- [15] C. Lee, S. S. Sarwar, and K. Roy, "Enabling Spike-based Backpropagation in State-of-the-art Deep Neural Network Architectures," pp. 1–25, 2019. [Online]. Available: <http://arxiv.org/abs/1903.06379>
- [16] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: Vgg and residual architectures," *Frontiers in neuroscience*, vol. 13, 2019.
- [17] D. Huh and T. J. Sejnowski, "Gradient descent for spiking neural networks," *Advances in Neural Information Processing Systems*, vol. 2018-Decem, pp. 1433–1443, 2018.
- [18] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, vol. 9, pp. 1–9, 2015.
- [19] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "Std-based spiking deep convolutional neural networks for object recognition," *Neural Networks*, vol. 99, pp. 56–67, 2018.
- [20] A. Jeyasothy, S. Sundaram, S. Ramasamy, and N. Sundararajan, "A novel method for extracting interpretable knowledge from a spiking neural classifier with time-varying synaptic weights," pp. 1–16, 2019.
- [21] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," *Advances in Neural Information Processing Systems*, no. Nips, pp. 4114–4122, 2016.
- [22] Y. Bengio, N. Léonard, and A. Courville, "Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation," pp. 1–12, 2013. [Online]. Available: <http://arxiv.org/abs/1308.3432>
- [23] T. Raiko, M. Berglund, G. Alain, and L. Dinh, "Techniques for Learning Binary Stochastic Feedforward Neural Networks," pp. 1–10, 2014. [Online]. Available: <http://arxiv.org/abs/1406.2989>
- [24] M. Yin and M. Zhou, "Arm: Augment-Reinforce-Merge Gradient for Stochastic Binary Networks," pp. 1–21, 2019. [Online]. Available: <https://github.com/mingzhang-yin/ARM-gradient>
- [25] S. Sharmin, P. Panda, S. S. Sarwar, C. Lee, W. Ponghiran, and K. Roy, "A Comprehensive Analysis on Adversarial Robustness of Spiking Neural Networks," 2019. [Online]. Available: <http://arxiv.org/abs/1905.02704>
- [26] A. Galloway, G. W. Taylor, and M. Moussa, "Attacking Binarized Neural Networks," pp. 1–14, 2017. [Online]. Available: <http://arxiv.org/abs/1711.00449>
- [27] E. B. Khalil, A. Gupta, and B. Dilkina, "Combinatorial Attacks on Binarized Neural Networks," pp. 1–12, 2018. [Online]. Available: <http://arxiv.org/abs/1810.03538>
- [28] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical Black-Box Attacks against Machine Learning," 2016. [Online]. Available: <http://arxiv.org/abs/1602.02697>
- [29] M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, S. J. Thorpe, and T. Masquelier, "Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks," *Pattern Recognition*, vol. 94, pp. 87–95, 2019. [Online]. Available: <https://doi.org/10.1016/j.patcog.2019.05.015>
- [30] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 582–597.
- [31] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 372–387.
- [32] A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard, "Sparsefool: a few pixels make a big difference," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9087–9096.
- [33] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into Transferable Adversarial Examples and Black-box Attacks," no. 2, pp. 1–24, 2016. [Online]. Available: <http://arxiv.org/abs/1611.02770>
- [34] N. Carlini and D. Wagner, "Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods," 2017. [Online]. Available: <http://arxiv.org/abs/1705.07263>
- [35] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1–4, pp. 17–37, 2002.
- [36] J. Gautrais and S. Thorpe, "Rate coding versus temporal order coding: a theoretical approach," *Biosystems*, vol. 48, no. 1–3, pp. 57–65, 1998.
- [37] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [38] A. Krizhevsky *et al.*, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.
- [39] B. S. Veeling, J. Linmans, J. Winkens, T. Cohen, and M. Welling, "Rotation equivariant CNNs for digital pathology," Jun. 2018.
- [40] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.
- [41] M. Mozafari, M. Ganjtabesh, A. Nowzari-Dalini, and T. Masquelier, "Spyketorch: Efficient simulation of convolutional spiking neural networks with at most one spike per neuron," *Frontiers in Neuroscience*, vol. 13, p. 625, 2019. [Online]. Available: <https://www.frontiersin.org/article/10.3389/fnins.2019.00625>
- [42] J. Rauber, W. Brendel, and M. Bethge, "Foolbox: A python toolbox to benchmark the robustness of machine learning models," *arXiv preprint arXiv:1707.04131*, 2017.
- [43] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

- [45] Z. Lin, R. Memisevic, and K. Konda, “How far can we go without convolution: Improving fully-connected networks,” *arXiv preprint arXiv:1511.02580*, 2015.
- [46] W. He, J. Wei, X. Chen, N. Carlini, and D. Song, “Adversarial Example Defenses: Ensembles of Weak Defenses are not Strong,” 2017. [Online]. Available: <http://arxiv.org/abs/1706.04701>
- [47] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [48] J. Chung, S. Ahn, and Y. Bengio, “Hierarchical multiscale recurrent neural networks,” *arXiv preprint arXiv:1609.01704*, 2016.

APPENDIX

We provide more details regarding the training of our classifiers, attack results for stochastic ANNs, transferability experiments. Furthermore, we provide experimental results of an alternate black-box attack proposed by [28] and also adversarial images of the different attacks against different model variants.

A. Details on Image Classifier Training

1) *ResNet18*: We trained the MNIST model for 20 epochs from scratch while loading pre-trained versions of ResNet18 (trained on ImageNet) and fine-tuning them for CIFAR10 and PCam. We selected the best performing set of weights based on the test dataset. For the case of MNIST, where the images are single channelled, we had to replace the initial convolution layer with one that has a input channel of 1 instead of 3, since the original model on torchvision was trained on ImageNet.

During training, we performed data augmentation on the training set and also normalised the data to between 0 and 1. For the case of CIFAR10 and PCam, we then performed colour normalisation by subtracting the mean and scaling to unit variance using the mean and standard deviation calculated from a subset of 50000 training samples from each dataset respectively. We did not perform any colour normalisation for MNIST.

TABLE V: Hyperparameters used for training the ResNet18 model.

	MNIST	CIFAR10	PCam
Optimiser	SGD	Adam [47]	Adam
Learning Rate (LR)	0.001	0.001	0.001
Weight decay (WD)	0.00001	0.0001	0.0001
LR decay step (Step)	5	3	3
LR decay factor (γ)	0.5	0.95	0.95
Batch size (BS)	128	256	256

2) *Mozafari et al. (SNN_m)*: The training code for SNN_m that we used was adopted from the SpykeTorch [41] library, following the tutorial provided. Each convolution layer was trained independently and chronologically via the STDP learning rule. For the last convolution layer, Reward-STDP was used instead where correct decisions were rewarded while incorrect decisions were punished. More details about the training procedure can be found in [29]. Table VI describes the hyperparameters we used and other hyperparameters that were not mentioned here indicates that the default values were used.

3) *Binary Stochastic Neuron Models*: We performed the similar preprocessing steps as described in A1. For our BSN models, we used the Straight-Through (ST) estimator in the backpropagation of the gradients. Slope annealing was done in conjunction with the ST estimator for the BSN-2 and BSN-4 variants only. More details about the slope annealing trick can be found in [48]. Table VII summarises some of the hyperparameters that we used in our experiments for the BSNs. As mentioned in the main text, we used a batch size of 128 across the settings and used Adam optimiser for the BSN-4 and BSN-L variants while using SGD for BSN-2.

Recall that BSN-2 indicates a simple 2-layered Multi-layer Perceptron (MLP) while BSN-4 indicates a 4-layered MLP variant. The difference between the standard MLP and our BSN here is that all hidden activations are modified to perform binarization of activations in a stochastic manner.

B. Full Experiment Results for BIM Attacks

Tables Xa and Xb shows the experiment results we obtained when we vary the attack strength for the BIM attack. We used three different values of ϵ , namely 8/255, 16/255 and 32/255. It is clear that BNN-D is highly susceptible to even the simple adversarial attacks with low attack strength, as the ASR for BNN-D is higher than that of ResNet18 while having lower L_2 norms per pixel at $\epsilon = 8/255$.

C. Full Experiment Results for Stochastic ANNs

Tables XI and XII summarises the full results we obtained for the case of stochastic ANNs, which includes the mean and standard deviation (STD) across 5 runs of the same experiment settings each. As we can observe from the relatively low standard deviations in the tables, the results we obtained for the stochastic ANNs are consistent. A ‘-’ was indicated for the case of Boundary attack on BNN-S due to the unsuccessful attempt in finding adversarial samples.

D. Transferability Experiment Results

Table XIV shows the full experiment results we obtained from the transferability experiments. This is a replica of Table 3 in the main paper and it is presented here again for easy referencing only. Based on Tables XIIIa and XIIIb, there was a reasonable amount of valid adversarial samples generated from ResNet18 that are launched towards the other model variants since they yield a high ASR. Recall that we have 500 samples for BIM and 100 samples for the other attack variants.

E. Performing Surrogate-based Black-box Attacks

In this section, we describe our attempt in performing another method of another black-box attack, as proposed by [28]. In short, this method is a two-step process; first, it involves training a surrogate model based on a very small subset of unseen data, whereby the ground truth label will be derived from the target classifier (*called the oracle*). The surrogate’s purpose is to approximate the decision boundary of the targeted classifier as close as possible. Once the surrogate

TABLE VI: Hyperparameters used for training SNN_m . Threshold of ∞ indicates that the maximum value was taken.

Dataset	Layer	No. of feature maps	Kernel window	Threshold
MNIST	1st conv	30	(5,5,6)	15
	2nd conv	250	(3,3,30)	10
	3rd conv	200	(5,5,250)	∞
CIFAR10	1st conv	300	(5,5,6)	15
	2nd conv	800	(3,3,300)	10
	3rd conv	500	(5,5,800)	∞

TABLE VII: Hyperparameters used for training the BSNs.

	BSN-2			BSN-4			BSN-L		
	MNIST	CIFAR10	PCam	MNIST	CIFAR10	PCam	MNIST	CIFAR10	PCam
LR	0.01	0.01	0.01	0.0001	0.0001	0.0001	0.001	0.01	0.01
WD	0.0001	0.001	0.0001	1E-6	1E-6	1E-6	1E-5	0.0001	0.0001
Step	5	5	5	5	5	5	5	50	5
γ	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.5	0.75

TABLE VIII: Dimensions of the Fully Connected (FC) layers for the BSN-2 model across the datasets.

	MNIST	CIFAR10	PCam
Input layer	784	3072	3072
Hidden layer	100	100	100
Output layer	10	10	2

TABLE IX: Dimensions of the Fully Connected (FC) layers for the BSN-4 model across the datasets.

	MNIST	CIFAR10	PCam
Input layer	784	3072	3072
Hidden layer 1	600	2000	2000
Hidden layer 2	400	750	750
Hidden layer 3	200	100	100
Output layer	10	10	2

has been trained, white-box attacks can be performed on this surrogate, since it exists locally with respect to the attacker. Thus, this converts a black-box to a white-box problem, which makes it significantly easier to launch attacks. We refer the reader to the work done in [28] for more details about the exact methodology.

In our experiments, we used a ResNet18 model as the surrogate architecture, while considering three different model variants as our target classifier (i.e. oracle), namely BSN-L, BNN-S and MCSEFRON. We evaluated on both the MNIST and CIFAR-10 datasets, taking 20% of the test data to be used for training the surrogate, and also performing the Jacobian-based Data Augmentation mentioned in [28] for finding more data points during the training of the surrogate. The remaining test data were used to evaluate the performance of the surrogate models.

As illustrated in Table XV, the ModCWL2 attack consistently achieve a higher transferability rate as compared to the vanilla CWL2 attack, which is also consistent with our explanation provided in the main text under the section ‘‘Augmented Carlini & Wagner L2 Attack Against Neural Networks’’. Another observation is that the BIM attack is in general more efficient in attaining transferable adversarial samples than the other more complex attack methods. This observation, together with the results in Table XIV, suggests

that iterative attacks are better performing for attacks involving transferability.

F. Adversarial Images of Various Attacks Across Datasets and Models

Here, we illustrate some sample adversarial images, compared to their original counterparts, for the different attacks and also against the various targeted models in Figure 3. We only show images from the MNIST dataset as they are the most relatable among the three datasets. The first column shows the original image that was used to construct the respective adversarial samples. The subsequent columns represent the different attack methods while the rows represent the different model variants. Below each image, there is a label that indicates what the predicted label was based on the corresponding model.

TABLE X: Adversarial success rate and mean L_2 norms per pixel for the BIM attack. For stochastic ANNs, an average of five runs were taken.

(a) ASR (in $[0,1]$) of the different variants of models. Step size of 0.05 and 100 iterations were performed. 500 samples were sampled for this experiment.

Attack strength, ϵ	Dataset	Resnet18	$SN N_m$	MCSEFRON	BSN-2	BSN-4	BSN-L	BNN-D	BNN-S
8/255	MNIST	0.190	0.026	0.054	0.099	0.172	0.038	0.992	0.196
	CIFAR10	0.990	0.358	0.980	0.944	0.941	0.706	1.000	0.851
	PCam	0.944	-	0.244	0.826	0.772	0.468	0.859	0.754
16/255	MNIST	0.796	0.056	0.156	0.336	0.585	0.129	1.000	0.375
	CIFAR10	1.000	0.534	0.998	0.977	0.964	0.852	1.000	0.951
	PCam	0.974	-	0.416	0.912	0.879	0.669	0.926	0.857
32/255	MNIST	1.000	0.120	0.294	0.774	0.874	0.506	1.000	0.566
	CIFAR10	1.000	0.694	0.998	0.981	0.955	0.884	1.000	0.953
	PCam	1.000	-	0.534	0.920	0.912	0.772	0.974	0.910

(b) Mean L_2 norms per pixel between the original image and its perturbed adversarial image of the different variants of models. Note that the values reported have been scaled up by a factor of 1000.

Attack strength, ϵ	Dataset	Resnet18	$SN N_m$	MCSEFRON	BSN-2	BSN-4	BSN-L	BNN-D	BNN-S
8/255	MNIST	0.8301	0.8201	0.4681	0.8386	0.8634	0.7615	0.8289	0.8125
	CIFAR10	0.5524	0.5338	0.5581	0.5221	0.5239	0.4606	0.5483	0.5234
	PCam	0.5530	-	0.5324	0.5277	0.5402	0.4907	0.5461	0.5453
16/255	MNIST	1.4925	1.4212	0.9072	1.5716	1.5529	1.4866	1.3604	1.4353
	CIFAR10	0.8809	0.9129	0.8866	0.8745	0.8548	0.7619	0.8767	0.8462
	PCam	0.8452	-	0.8600	0.8345	0.8516	0.8058	0.8698	0.8680
32/255	MNIST	2.1667	2.4142	1.4126	2.6164	2.2969	2.5716	1.5606	2.1711
	CIFAR10	0.9318	1.3968	0.8924	1.1667	0.9829	1.0891	0.9606	1.0494
	PCam	0.9794	-	1.4248	0.9110	0.9179	1.0017	1.0343	0.9888

TABLE XI: ASR and mean L_2 norms per pixel for the BIM attack on the stochastic ANNs. Note that for each experiment, we took the average and standard deviation across 5 runs of the same setting.

(a) ASR (in range $[0,1]$) across the different datasets for stochastic ANNs.

ϵ	Dataset	BSN-2		BSN-4		BSN-L		BNN-S	
		Mean	STD	Mean	STD	Mean	STD	Mean	STD
8/255	MNIST	0.0992	0.0085	0.1724	0.0107	0.0384	0.0082	0.1960	0.0108
	CIFAR10	0.9440	0.0107	0.9408	0.0100	0.7056	0.0237	0.8508	0.0118
	PCam	0.8256	0.0117	0.7720	0.0049	0.4680	0.0164	0.7540	0.0022
16/255	MNIST	0.3356	0.0204	0.5852	0.0084	0.1288	0.0138	0.3748	0.0231
	CIFAR10	0.9772	0.0056	0.9644	0.0077	0.8524	0.0118	0.9512	0.0084
	PCam	0.9120	0.0111	0.8788	0.0081	0.6688	0.0169	0.8568	0.0081
32/255	MNIST	0.7736	0.0097	0.8740	0.0116	0.5056	0.0132	0.5656	0.0289
	CIFAR10	0.9812	0.0020	0.9552	0.0106	0.8840	0.0131	0.9532	0.0111
	PCam	0.9204	0.0098	0.9116	0.0114	0.7724	0.0135	0.9100	0.0097

(b) Mean L_2 norms per pixel between the original samples and their adversarial counterparts, across the different datasets for stochastic ANNs. Note that the metrics reported here are scaled up by a factor of 1000.

ϵ	Dataset	BSN-2		BSN-4		BSN-L		BNN-S	
		Mean	STD	Mean	STD	Mean	STD	Mean	STD
8/255	MNIST	0.8386	0.0196	0.8634	0.0104	0.7615	0.0411	0.8125	0.0074
	CIFAR10	0.5221	0.0038	0.5239	0.0034	0.4606	0.0051	0.5234	0.0026
	PCam	0.5277	0.0035	0.5402	0.0055	0.4907	0.0055	0.5453	0.0027
16/255	MNIST	1.5716	0.0108	1.5529	0.0094	1.4866	0.0122	1.4353	0.0113
	CIFAR10	0.8745	0.0043	0.8548	0.0086	0.7619	0.0068	0.8462	0.0063
	PCam	0.8345	0.0093	0.8516	0.0115	0.8058	0.0039	0.8680	0.0047
32/255	MNIST	2.6164	0.0175	2.2969	0.0063	2.5716	0.0216	2.1711	0.0135
	CIFAR10	1.1667	0.0074	0.9829	0.0136	1.0891	0.0147	1.0494	0.0091
	PCam	0.9110	0.0035	0.9179	0.0072	1.0017	0.0119	0.9888	0.0053

TABLE XII: ASR and mean L_2 norms per pixel for the other attack variants on the stochastic ANNs. Note that for each experiment, we took the average and standard deviation across 5 runs of the same setting.

(a) ASR (in range[0,1]) across the different datasets for stochastic ANNs.

Model	Dataset	CWL2		ModCWL2		Boundary	
		Mean	STD	Mean	STD	Mean	STD
BSN-2	MNIST	0.2040	0.0492	0.3340	0.0463	0.0080	0.0040
	CIFAR10	0.4020	0.0354	0.5280	0.0117	0.2900	0.0636
	PCam	0.1680	0.0194	0.1380	0.0040	0.1020	0.0382
BSN-4	MNIST	0.1800	0.0352	0.3080	0.0412	0.0140	0.0080
	CIFAR10	0.2000	0.0329	0.2260	0.0377	0.1820	0.0223
	PCam	0.1120	0.0331	0.1440	0.0403	0.0680	0.0248
BSN-L	MNIST	0.0100	0.0110	0.2320	0.0371	0.0120	0.0117
	CIFAR10	0.2340	0.0350	0.2300	0.0374	0.1920	0.0299
	PCam	0.1020	0.0319	0.1520	0.0325	0.0800	0.0167
BNN-S	MNIST	0.0300	0.0126	0.3700	0.0385	0.0300	0.0110
	CIFAR10	0.1420	0.0279	0.1880	0.0232	0.1140	0.0377
	PCam	0.1260	0.0185	0.1140	0.0258	-	-

(b) Mean L_2 norms per pixel between the original samples and their adversarial counterparts, across the different datasets for stochastic ANNs. Note that the metrics reported here are scaled up by a factor of 1000.

Model	Dataset	CWL2		ModCWL2		Boundary	
		Mean	STD	Mean	STD	Mean	STD
BSN-2	MNIST	2.7403	0.2163	8.6806	0.2813	1.2268	2.0642
	CIFAR10	0.0874	0.0120	0.9369	0.1156	1.9463	0.2299
	PCam	0.0954	0.0189	0.0535	0.0203	0.2154	0.3932
BSN-4	MNIST	2.5473	0.2728	9.0173	0.2855	1.3307	1.6021
	CIFAR10	0.0187	0.0109	0.0244	0.0121	0.0635	0.0687
	PCam	0.2124	0.0399	0.1422	0.0484	0.0146	0.0164
BSN-L	MNIST	0.0008	0.0007	9.5389	0.1533	0.8897	0.7185
	CIFAR10	0.0059	0.0026	0.0163	0.0061	0.0776	0.0516
	PCam	0.1738	0.0476	0.2458	0.1237	0.0013	0.0013
BNN-S	MNIST	2.2963	1.4531	9.9991	0.2580	0.1485	0.0726
	CIFAR10	0.0507	0.0147	0.0590	0.0193	0.1411	0.1388
	PCam	0.2636	0.0557	0.2213	0.0500	-	-

TABLE XIII: ASR (in range [0,1]) of the respective attacks against ResNet18.

(a) BIM attack. ASR based on 500 samples.

$\epsilon = 8/255$			$\epsilon = 16/255$			$\epsilon = 32/255$		
MNIST	CIFAR10	PCam	MNIST	CIFAR10	PCam	MNIST	CIFAR10	PCam
0.19	0.998	0.944	0.796	1	0.974	1	1	1

(b) Other attack variants. ASR based on 100 samples.

	CWL2	ModCWL2	Boundary
MNIST	0.97	1	1
CIFAR10	1	1	1
PCam	1	1	0.73

TABLE XIV: Transferability rate of the resultant adversarial samples generated from ResNet18. Only adversarial samples successful against ResNet18 were considered. A higher rate indicates a more successful misclassification attempt of transferred samples to their respective targeted models.

Attack method	ϵ	MCSEFRON	$SN N_m$	BSN-L	BNN-S
BIM	8/255	0.219	0	0.0556	0.345
	16/255	0.321	0.0101	0.0453	0.356
	32/255	0.32	0.008	0.06	0.446
	64/255	0.338	0.006	0.048	0.434
	128/255	0.338	0.006	0.076	0.45
CWL2	-	0.0211	0	0.0206	0.0938
ModCWL2	-	0.31	0.26	0.26	0.28
Boundary	-	0.2	0	0.01	0.22

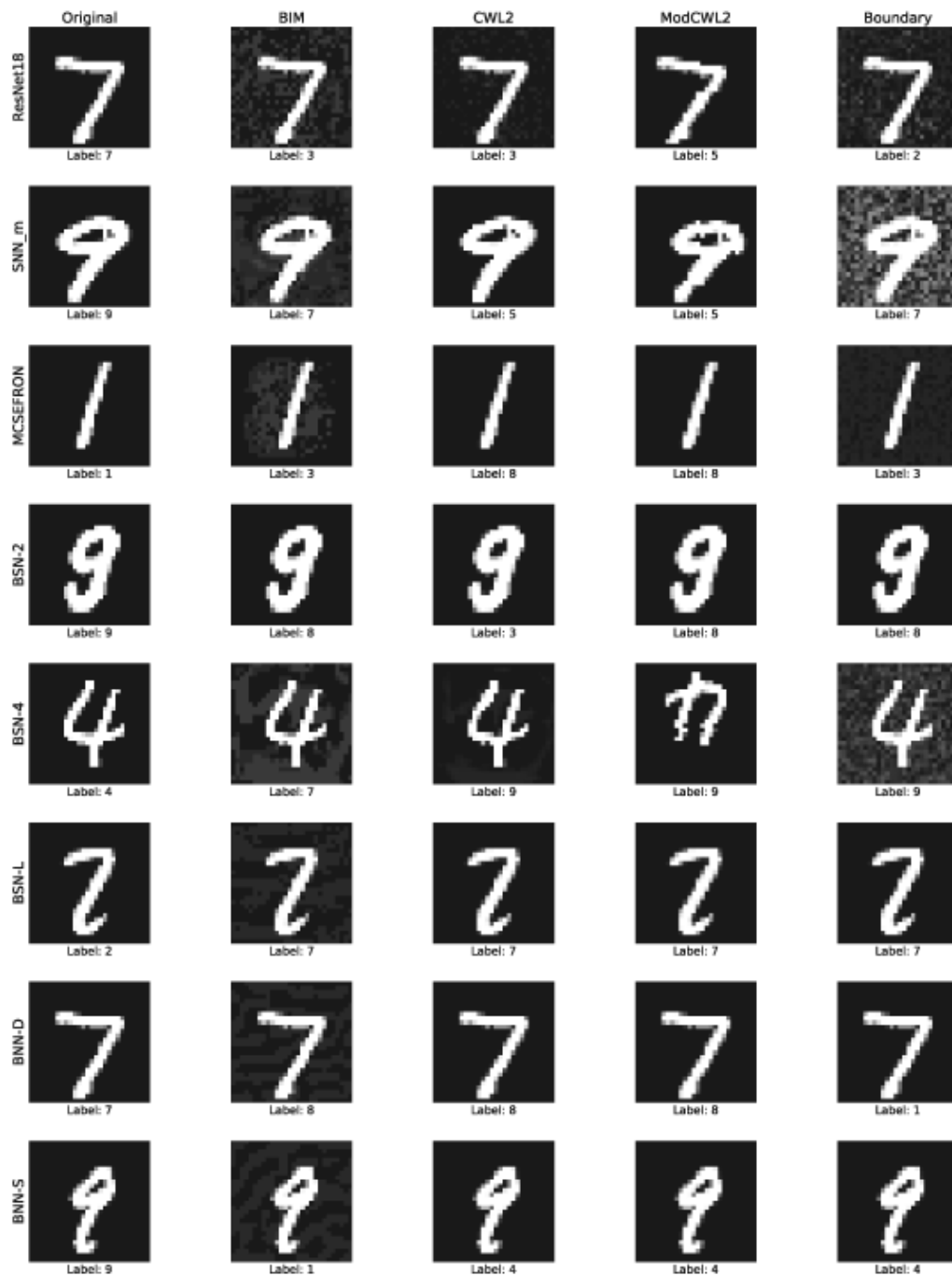


Fig. 3: Sample adversarial images from the different attack methods against the different variants of neural networks on the MNIST dataset.

TABLE XV: Transferability rates of the successful adversarial samples on the ResNet18 model when launched against the respective oracles. 100 samples were used for each experiment. For the BIM attack, $\epsilon = 32/255$ was used as the attack strength throughout.

Dataset	Attack Method	Oracle		
		BSN-L	BNN-S	MCSEFRON
MNIST	BIM	0.1717	0.4900	0.3367
	CWL2	0.0106	0.0225	0.1098
	ModCWL2	0.3200	0.1800	0.3900
CIFAR-10	BIM	0.6300	0.3367	0.7300
	CWL2	0.2747	0.1939	0.6588
	ModCWL2	0.3100	0.2100	0.6200