

Enhancing (?) security with MimeticNets

Bryce Chudomelka

December 14, 2020

Abstract

Computer vision has achieved orders of magnitude improvement in accuracy over the past decade, but the neural networks used in these applications are not without fault. Adversarial attacks on computer vision models pose security risks when neural networks are used in security-critical fields, such as autonomous vehicles, facial recognition, or malware detection. We implement a secondary network layer for image pre-processing to simultaneously improve natural accuracy and adversarial robustness with mimetic operators. A fully connected network is used to learn a nonlinear function in the Perona-Malik model to add clarity to the input image prior too the forward pass. Various model combinations are trained to provide benchmarks, while the successes and drawbacks of this methodology are discussed.

1 Introduction

The modern landscape of scientific computing has transformed over the past decade to incorporate neural networks in fields that were largely dominated by deterministic models. Computer vision (CV) is an example of such a field that has been entirely upended by probabilistic models, *i.e.*, deep neural networks, where the methods prior had previously been guided by partial differential equations (PDEs) [9]. Deep neural networks (DNNs) have dramatically improved the performance of CV algorithms because models are able to *learn* abstract hierarchical representations from large data sets which are readily available. All state-of-the-art (SOTA) models utilize DNNs to perform a variety of CV tasks, such as object detection, semantic segmentation, image restoration, etc., and are currently being used in many security


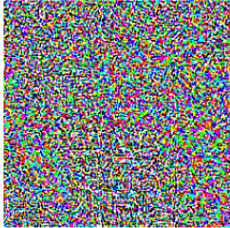

	$+ .007 \times$		$=$	
x		$\text{sign}(\nabla_x J(\theta, x, y))$		$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$
“panda”		“nematode”		“gibbon”
57.7% confidence		8.2% confidence		99.3 % confidence

Figure 1: The canonical example provided by Goodfellow et al. visually demonstrates the effectiveness of the FGSM attack. A picture of a panda on the left with confidence 57.7% is perturbed by non-trivial noise then added together to yield, what looks like a panda. The DNN incorrectly classifies this as a gibbon with a 99.3% confidence. This behavior is similar to an optical illusion because of the perceptual similarity between the perturbed and original image.

critical tasks such as facial recognition and autonomous vehicles[1, 3, 11].

The work done in this research attempts to address the problem of adversarial robustness from the point of PDEs using Mimetic Finite Differences. Our contribution to this field utilizes a DNN, referred to as MimeticNet¹, for image restoration and, hopefully, improves adversarial robustness [13]. First, we begin by training some common CV DNNs with, and without adversarial training, then evaluate the accuracy of each model for benchmarking. Next, we implement a secondary network that, when inserted before the DNN, effectively pre-processes the input image using Perona-Malik diffusive model, and *learns* an optimal form of the differential equation to restore images that improve network accuracy [12].

2 Related Works

Previous work introduced in [5] demonstrated the peculiar behavior of these networks; see Figure 1 for the canonical example. Many attack and defensive methods have been proposed since this discovery, but there has not been a solution. A quick white-box attack on a DNN can be found using the Fast Gradient Sign Method (FGSM) proposed in [5]. This attack uses the network parameter weights, θ and the model loss function, \mathcal{L} , to calculate the gradient, in order to generate an adversarial example, x_{adv} .

$$x_{adv} = x + \varepsilon \cdot \text{sign}(\nabla_x \mathcal{L}(\theta, x, y)) \quad (1)$$

Madry et al. 2017 explores the problem of adversarial robustness in the context of optimization. He mathematically justifies that this cat and mouse game of attack and defend is logically equivalent to the min-max saddle-point problem [10]. A robust network tries to minimize the empirical risk, \mathbf{E} , over the data set, \mathcal{D} , with image x and target classification label y , according to an adversarial attack on the network which is trying to maximize the loss function, \mathcal{L} , with a perturbation δ , with some set, \mathcal{S} , of allowable perturbations; thus a min-max optimization problem.

$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbf{E}_{(x,y) \sim \mathcal{D}} \left[\max_{\delta \in \mathcal{S}} \mathcal{L}(\theta, x + \delta, y) \right] \quad (2)$$

Recently, many novel networks have been created to defend against adversarial attacks [7]. In this work, we will utilize the Strong Stability Preserving Network (SSPNets); in particular, we use the 2nd order implementation, see [7] for details. SSPNets have shown improved attack accuracy versus ResNets, all else being equal, and served as another model for comparison [6]. By comparing the performance between these two choices of networks we can gain possible insight into any patterns that emerge, *i.e.*, benchmarking the MimeticNet.

Enhancing network security is critical to ensuring the proper functionality of DNNs where security takes precedent. This can be achieved in a variety of ways, such as pre-processing, network architecture, or post-processing. We can utilize a secondary neural network to enhance the perturbed image prior to input. The effects of image restoration via PDEs was successfully

¹<https://github.com/BryceWayne/MimeticNets>

explored using mimetic operators [2]. This can be easily implemented by discretizing the differential operators with operators that *mimic* the operations they represent; thus, we say they are mimetic. Bazan et al 2011 implement the Perona-Malik model which is defined as,

$$\begin{aligned} u_t - \nabla \cdot (g(|\nabla u|^2) \cdot \nabla u) &= 0, & \text{on } \Omega \times [0, \infty), \\ u(\mathbf{x}, 0) &= u_0(\mathbf{x}), & \text{on } \Omega, \\ \langle g \cdot \nabla u, \mathbf{n} \rangle &= 0, & \text{on } \partial\Omega \times (0, \infty). \end{aligned} \quad (3)$$

3 Methodology

MimeticNet is based upon the Perona-Malik model in Equation (3). The primary layer in the DNN is comprised of either ResNet or SSPNet, the SSP2 variety, with 3 residual, blocks. These networks were trained with, or without, adversarial training for 200 epochs using a cross entropy loss function and the Adam optimizer, with the standard optimization settings. The models were trained on the Fashion MNIST (FMNIST²) data set [14] which is comprised of 70,000 28×28 greyscale images; 60,000 for training and 10,000 for testing. Each model was selected based upon the best out-of-sample accuracy over the 200 epochs.

This extra layer is trained from initialization for 50 epochs using the Adam optimizer and a cross entropy loss function, \mathcal{L} [4, 8]. Where the loss is only computed between the original input image, x , and the output classification, \hat{y} . The underlying ResNet network model parameters are frozen prior to training ensuring that the MimeticNet performance is solely based upon improving the network. We use the mimetic operators D , and G , to represent the divergence and gradient operators, respectively, to pre-process the data.

MimeticNet *learns*, through training, the underlying function that models the noise distribution, \hat{g} , to allow for 'restoration' of the DNNs accuracy. We are able to rewrite Eq. (3) in terms of operators with direct substitution, and in doing so arrive at our model.

$$u_t - D(\hat{g} \cdot Gu) = 0 \quad (4)$$

Using a forward-Euler method, we can discretize our equation in time and arrive at our temporal update equation.

$$u_{n+1} = u_n - \Delta t \cdot D(\hat{g} \cdot Gu) \quad (5)$$

²<https://github.com/zalandoresearch/fashion-mnist>

Model	Adversary	Natural
ResNet	Vanilla	0.9908
ResNet	FGSM	0.9062
ResNet	PGD	0.8931
SSP	Vanilla	0.9893
SSP	FGSM	0.8940
SSP	PGD	0.8937

Table 1: Different models were trained with and without adversarial training. The models selected were ResNet and SSPNets, specifically SSP2; see [7] for more details. The vanilla networks perform better than networks trained on adversarial images. The trade-off of natural accuracy for adversarial robustness is observed; see [15] for details.

It is common to use some $\Delta t \ll 1$, but within the machine learning community it is common to learn a nonlinear function able to take the full step. Finally, I will attack these models with Equation (1) and (2) to observe robustness.

4 Results

Vanilla networks show a clear pattern of degradation as we iterate the image for pre-processing; as seen in Figures 2 and 3. As N increases, the number of times the image is processed prior to being fed into the network, then the accuracy decays smoothly back to baseline, *i.e.*, no further improvement can be made. The numerical results for each computation are located in Tables 2 and 3. The best accuracy occurred for $N = 1$, regardless of the underlying model. This could be a result of taking the full Δt step but my initial results for Δt being on the order of 10^{-2} did not yield any successful results.

This same behavior is not observed when the underlying network was trained with adversarial training, which can be viewed in Figures 4 and 5. In fact, the opposite was true when the base layer in the network was trained with images perturbed via Equation (1). As the number of iterations N increased, the network accuracy was restored by as much as 1%. The numerical results for each computation are located in Tables 4 and 5. These results exhibit stochastic tendencies, unlike the vanilla models, but do demonstrate a slight trend; perhaps some more experimentation here could improve accu-

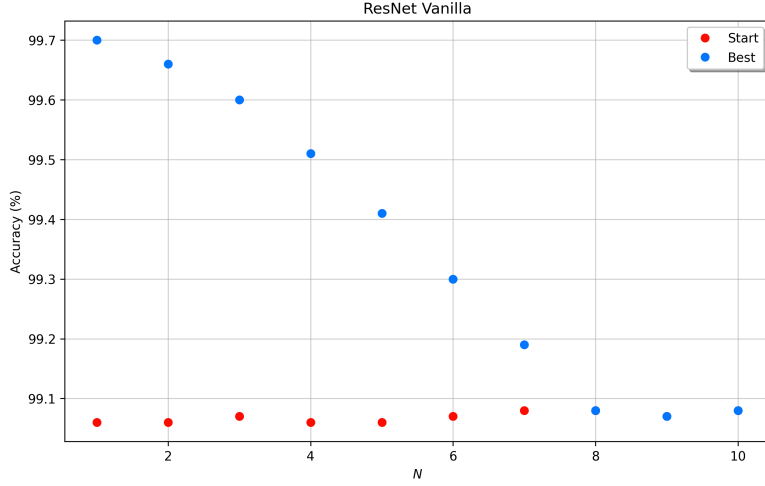


Figure 2: A plot of Accuracy (%) vs. Iteration (N) demonstrates the effectiveness of only using one iteration. The performance improvement of the network yields an impress 99.7% accuracy after one iteration, but is inversely correlated with N .

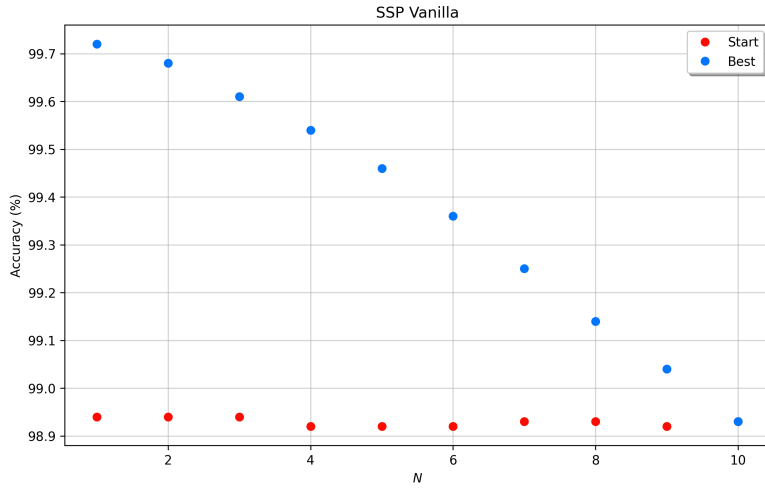


Figure 3: A plot of Accuracy (%) vs. Iteration (N) demonstrates the effectiveness of only using one iteration. The performance improvement of the network yields an impress 99.7% accuracy after one iteration, but is inversely correlated with N . This similar behavior was observed in the vanilla ResNet model as well.

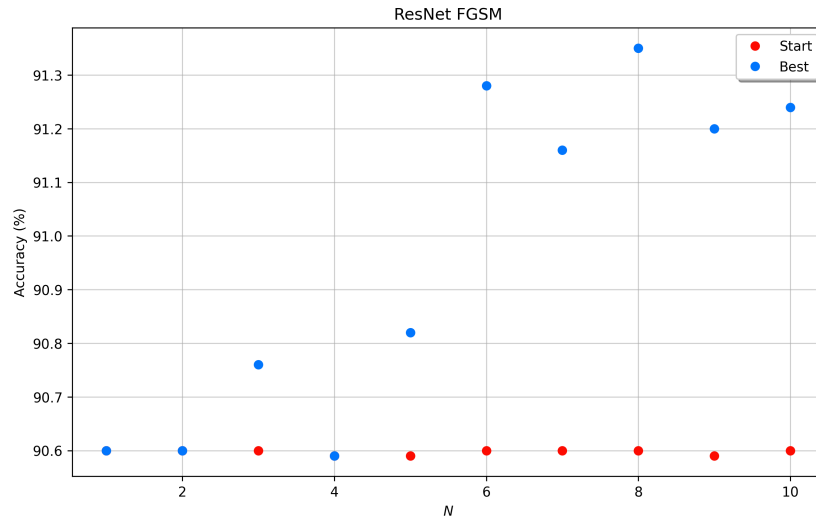


Figure 4: A plot of Accuracy (%) vs. Iteration (N) for the ResNet model trained on adversarial examples via the FGSM attack demonstrates a positive trend, but only after 6 or more iterations of the MimeticNet. The overall accuracy of the underlying network was improved by 0.71% at most after 8 iterations. This behavior is in contrast with vanilla ResNet, but has been observed already; see [15].

Model	Adversary	N	Improvement	Start	Best
ResNet	Vanilla	1	0.64	99.06	99.70
ResNet	Vanilla	2	0.61	99.06	99.66
ResNet	Vanilla	3	0.53	99.07	99.60
ResNet	Vanilla	4	0.45	99.06	99.51
ResNet	Vanilla	5	0.35	99.06	99.41
ResNet	Vanilla	6	0.24	99.07	99.30
ResNet	Vanilla	7	0.12	99.08	99.19
ResNet	Vanilla	8	0.00	99.08	99.08
ResNet	Vanilla	9	0.00	99.07	99.07
ResNet	Vanilla	10	0.00	99.08	99.08

Table 2: A vanilla ResNet model with 3 blocks was trained for 200 epochs and had a benchmark classification accuracy of 99.06%. Next, we placed the MimeticNet on top of the model, froze the ResNet parameters and retrained the model, all while measuring the loss using the standard Cross Entropy loss function. MimeticNet enhanced the image classification accuracy by 0.64% to obtain an overall best accuracy of 99.70% after only one iteration. It is observed that the repeated iteration of MimeticNet reduces the image processing capabilities of MimeticNet.

racy.

There does not seem to be a clear pattern of restoration when models were trained with adversarial images perturbed by Equation (2); as seen in Figures 6 and 7. Overall model accuracy improvement is observed but not as a function of N , the number of iterations. The numerical results for each computation are located in Tables 6 and 7. Clearly, this model could be improved, either by changing the differential equation or the step size Δt .

Unfortunately, there is no added benefit in defense from attacking these models. In fact, there is a complete degradation in network performance and the resulting network output is random, with model accuracy 10%, *i.e.*, the same or, or worse, then randomly guessing. Thus, I chose not to include the results. I attribute this to a bug, rather than the network itself, but I was unable to debug the model. This could be explored in future works.

Model	Adversary	N	Improvement	Start	Best
SSP	Vanilla	1	0.78	98.94	99.72
SSP	Vanilla	2	0.75	98.94	99.68
SSP	Vanilla	3	0.68	98.94	99.61
SSP	Vanilla	4	0.63	98.92	99.54
SSP	Vanilla	5	0.55	98.92	99.46
SSP	Vanilla	6	0.44	98.92	99.36
SSP	Vanilla	7	0.33	98.93	99.25
SSP	Vanilla	8	0.21	98.93	99.14
SSP	Vanilla	9	0.11	98.92	99.04
SSP	Vanilla	10	0.00	98.93	98.93

Table 3: A vanilla SSP model with 3 blocks was trained for 200 epochs and had a benchmark classification accuracy of 98.94%; slightly less than ResNet. MimeticNet enhanced the image classification accuracy by 0.78% to obtain an overall best accuracy of 99.72% after only one iteration. It is observed that the repeated iteration of MimeticNet reduces the image processing capabilities of MimeticNet, just as with vanilla ResNet.

Model	Adversary	N	Improvement	Start	Best
ResNet	FGSM	1	0.00	90.60	90.60
ResNet	FGSM	2	0.00	90.60	90.60
ResNet	FGSM	3	0.18	90.60	90.76
ResNet	FGSM	4	0.00	90.59	90.59
ResNet	FGSM	5	0.24	90.59	90.82
ResNet	FGSM	6	0.75	90.60	91.28
ResNet	FGSM	7	0.62	90.60	91.16
ResNet	FGSM	8	0.82	90.60	91.35
ResNet	FGSM	9	0.68	90.59	91.20
ResNet	FGSM	10	0.71	90.60	91.24

Table 4: A FGSM ResNet model with 3 blocks was trained for 200 epochs and had a benchmark classification accuracy of 90.60%. MimeticNet enhanced the image classification accuracy by 0.82% to obtain an overall best accuracy of 91.35% after 8 iterations. It is observed that the repeated iteration of MimeticNet increases the image processing capabilities of MimeticNet.

Model	Adversary	N	Improvement	Start	Best
SSP	FGSM	1	0.25	89.39	89.62
SSP	FGSM	2	0.23	89.38	89.59
SSP	FGSM	3	0.25	89.39	89.62
SSP	FGSM	4	0.57	89.38	89.89
SSP	FGSM	5	0.56	89.38	89.88
SSP	FGSM	6	0.94	89.39	90.22
SSP	FGSM	7	1.00	89.38	90.27
SSP	FGSM	8	0.91	89.37	90.18
SSP	FGSM	9	0.78	89.38	90.07
SSP	FGSM	10	0.87	89.38	90.16

Table 5: A FGSM SSP model with 3 blocks was trained for 200 epochs and had a benchmark classification accuracy of 89.38%. MimeticNet enhanced the image classification accuracy by 1% to obtain an overall best accuracy of 90.27% after 7 iterations. It is observed that the repeated iteration of MimeticNet increases the image processing capabilities of MimeticNet.

Model	Adversary	N	Improvement	Start	Best
ResNet	PGD	1	0.68	89.33	89.94
ResNet	PGD	2	0.56	89.33	89.84
ResNet	PGD	3	0.49	89.33	89.77
ResNet	PGD	4	0.51	89.33	89.78
ResNet	PGD	5	0.50	89.33	89.77
ResNet	PGD	6	0.49	89.32	89.76
ResNet	PGD	7	0.49	89.33	89.76
ResNet	PGD	8	0.53	89.32	89.80
ResNet	PGD	9	0.32	89.32	89.61
ResNet	PGD	10	0.33	89.32	89.61

Table 6: A PGD ResNet model with 3 blocks was trained for 200 epochs and had a benchmark classification accuracy of 89.33%. MimeticNet enhanced the image classification accuracy by 0.68% to obtain an overall best accuracy of 89.94% after only one iteration.

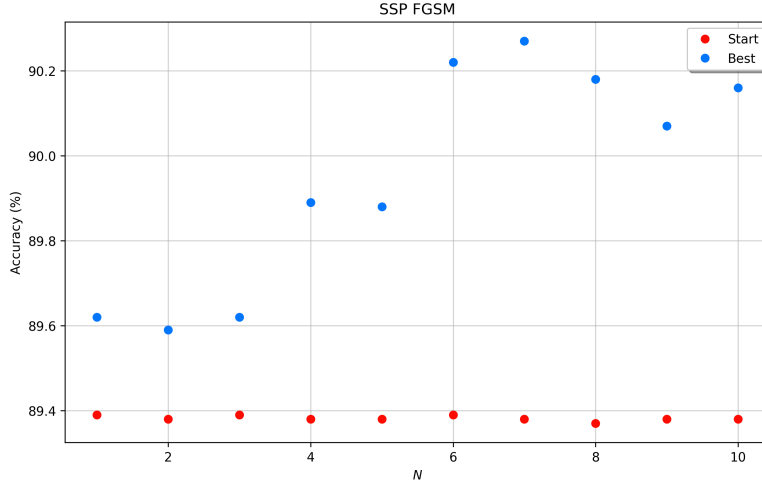


Figure 5: A plot of Accuracy (%) vs. Iteration (N) for the SSP model trained on adversarial examples via the FGSM attack demonstrates a positive trend. The overall accuracy of the underlying network was improved by 1.00% at most after 7 iterations; similar behavior to ResNet. This behavior differs from vanilla ResNet and SSP, yet is similar to Figure 4.

Model	Adversary	N	Improvement	Start	Best
SSP	PGD	1	0.47	89.37	89.79
SSP	PGD	2	0.39	89.37	89.72
SSP	PGD	3	0.34	89.36	89.66
SSP	PGD	4	0.37	89.35	89.69
SSP	PGD	5	0.38	89.36	89.71
SSP	PGD	6	0.44	89.36	89.75
SSP	PGD	7	0.25	89.36	89.58
SSP	PGD	8	0.42	89.35	89.73
SSP	PGD	9	0.74	89.36	90.02
SSP	PGD	10	0.80	89.36	90.08

Table 7: A PGD SSP model with 3 blocks was trained for 200 epochs and had a benchmark classification accuracy of 89.36%. MimeticNet enhanced the image classification accuracy by 0.80% to obtain an overall best accuracy of 90.08% after only one iteration.

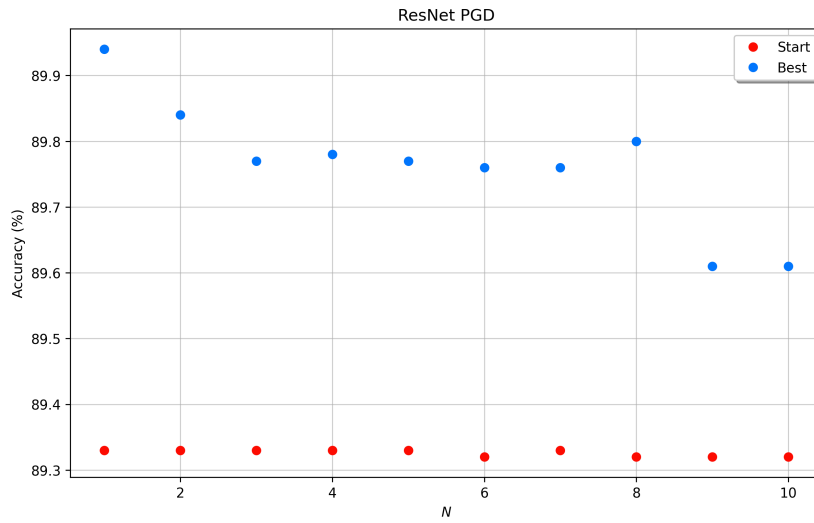


Figure 6: A plot of Accuracy (%) vs. Iteration (N) for the ResNet model trained on adversarial examples via the PGD attack does not demonstrate an overall trend, unlike the vanilla or FGSM variations. In fact, the improvement seems relatively consistent regardless of N , with possibly some stochastic variation. The overall accuracy of the underlying network was improved by 0.89% at most after 1 iterations.

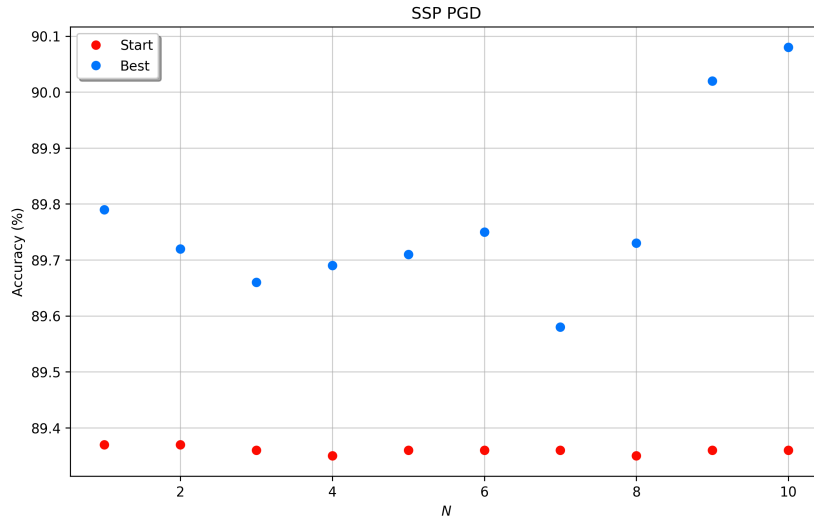


Figure 7: A plot of Accuracy (%) vs. Iteration (N) for the SSP model trained on adversarial examples via the PGD attack does not demonstrate an overall trend; possibly some stochastic behavior for $N = 10$. The improvement from MimeticNet seems relatively consistent regardless of N . The overall accuracy of the underlying network was improved by 0.80% at most after 10 iterations.

5 Conclusion

MimeticNets are fast, add an additional layer to an already trained SOTA model, and can improve classification accuracy of image classification models under a variety of conditions. These networks use mimetic operators to perform image processing techniques based upon the Perona-Malik model, a diffusive model, and can easily be modified to solve nonlinear partial differential equations. MimeticNets improve the network accuracy of vanilla networks in a single iteration and can restore up to roughly 1% of natural accuracy, where the original models already had approximately 99% natural accuracy. Also, MimeticNets can restore some accuracy in adversarial trained models but more iterations of the differential equation are required. This could be explored in future works. Also, one might consider trying a different underlying differential equation model, because they can easily be generated and generalized using mimetic operators.

References

- [1] Jon Barker. Malware detection in executables using neural networks. *NVIDIA Developer Blog*, Aug 2020. <https://developer.nvidia.com/blog/malware-detection-neural-networks/>.
- [2] C. Bazan, M. Abouali, J. Castillo, and P. Blomgren. Mimetic finite difference methods in image processing. *Computational and Applied Mathematics*, 30:701 – 720, 00 2011.
- [3] Ben Dickson. The security threats of neural networks and deep learning algorithms. Dec 2018. <https://bdtechtalks.com/2018/12/27/deep-learning-adversarial-attacks-ai-malware/>.
- [4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
- [5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.
- [6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.
- [7] Byungjoo Kim, Bryce Chudomelka, Jinyoung Park, Jaewoo Kang, Youngjoon Hong, and Hyunwoo J. Kim. Robust neural networks inspired by strong stability preserving runge-kutta methods. In *European Conference in Computer Vision*, 2020.
- [8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems 25*, 2012.
- [10] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.
- [11] Nvidia. News center, Jul 2018. <https://news.developer.nvidia.com/accelerating-autonomous-vehicle-safety/>.

- [12] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.
- [13] Bryce Wayne. Mimetic networks for image pre-processing, Dec 2020. [=https://github.com/BryceWayne/MimeticNets](https://github.com/BryceWayne/MimeticNets).
- [14] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.
- [15] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.