REPLICATING AND IMPROVING PERMUTATION PHASE DEFENSE

Oyku Han
oykuh@andrew.cmu.edu

Harlin Lee
harlinl@andrew.cmu.edu

Stanley Nnamdi Adom
sadom@andrew.cmu.edu

ABSTRACT

Deep neural networks have shown the capacity to perform classification tasks better than humans in some instances. However, they are susceptible to fatal adversarial attacks. Permutation Phase Defense (PPD), is a method purposed at the ICLR conference to combat adversarial attacks. It amalgamates random permutation of image pixel and phase components of its Fourier transform. In this report, our PPD implementation is compared to original paper, and methods to improve PPD are proposed. Testing on MNIST and CIFAR-10 dataset, the l_∞ norm attacks produced similar outcome while some in l_2 diverged. The novelty which resulted in the highest performance increase was to configure the PPD pipeline with intelligent ordering. Our scripts are available online.

1 INTRODUCTION

Deep Neural Networks (DNNs) are among the most preferred methods in image classification tasks, and their performance can match human's on simple tasks such as classification of numbers on the MNIST dataset. [Ciresan et al. (2012)] However, DNNs are also vulnerable to adversarial attacks, which perturb the input deliberately such that the network is fooled into making false predictions. In the field of image recognition, adversarial examples are images that are very close to the decision boundary of a classification network, and their difference from clean, unperturbed images is undetectable by eye. [Goodfellow et al. (2015)]

Successful adversarial attacks on DNNs are a serious source of concern for many applications including autonomous driving and medical image diagnostics. Finding a robust defense mechanism against attacks is an important open question. Permutation Phase Defense (PPD), which is submitted to the 2019 International Conference on Learning Representations (ICLR), proposes a method to mitigate the effects of the adversary on the network by encrypting and transforming the input to the frequency domain. [Anonymous (2019)]

As part of the 2019 ICLR Reproducibility Challenge, we have reproduced the results presented in the PPD paper that mostly back up the PPD findings except a few cases in l_2 norm attacks. We came up with further tests to analyze paper's experimental design and their shortcomings by evaluating each component of PPD individually. Finally, we have investigated several ways to increase the robustness of PPD against adversaries and shown that the accuracy can be increased significantly by intelligently re-configuring PPD layers.

2 RELATED WORK

2.1 ADVERSARIAL ATTACK AND DEFENSE

After the introduction of adversarial examples in Szegedy et al. (2014), several methods have been proposed as defense mechanisms. Three main methods of defense are adversarial training, hiding the model, or masking the input data from the adversary.

Goodfellow et al. (2015) proposed an adversarial training method, where the network is trained using the adversarial examples from Fast Gradient Sign Method to make the network robust to small

perturbations. Madry et al. (2018) utilizes the Projected Gradient Descent attack during network training such that the network would be resistant to a predefined set of attacks. Although the adversarial training helps immensely in the defense against the attacks that was used in the training, they still leave the network open to different types of attacks, and is computationally expensive.

Another solution was to hide the classifier weights or gradients from the adversary. But Papernot et al. (2017) discovered that by transferring adversarial examples from one model to an adversarially trained substitute, an adversarial attack that would not succeed on the initial network was proven to work on the substitute. In addition, Athalye et al. (2017) showed that the gradients can be approximately recovered, which is able to deceive defense techniques that rely on hiding the classifier.

Another common approach is to mask a portion of the input data from the adversary. Hiding parts of the input via replacing segments of image or recovering initially randomly dropped pixels by total variance minimization was proposed as a defense in Guo et al. (2018). However, to maintain good prediction performance, most of the image has to be maintained close to its original form, which does not help in defending against adversaries, as shown in Athalye et al. (2017).

2.2 Permutation Phase Defense (PPD)

As discussed, hiding the classifier or the input data can be circumvented, while adversarial training is not robust to more sophisticated adversaries than the one it was trained with. Therefore, Permutation Phase Defense (PPD) was proposed, and it claims to defend the neural network against adversarial attacks without being constrained to certain attacks.

The first line of defense in PPD is to permute the pixels of the input image using a fixed random seed that is hidden from the adversary. The permuted images then undergo Fourier transformation, after which only the phase information is kept to train a DNN for image classification. This architecture is depicted in Figure 1. The preferred type of neural network for PPD is a Multi-Layer Perceptron (MLP). Thus, the shuffling of the pixels provides better encryption than partially hiding the input, and does not disrupt the prediction accuracy, while working in the phase domain ensures that the perturbation gets distributed over pixels, and its effect is limited to that of random noise.

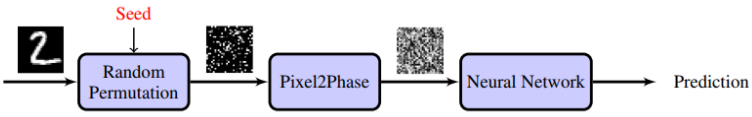


Figure 1: Architecture of the Permutation Phase Defense model, reproduced from the original paper.

The original PPD paper trains an ensemble of PPD models — each with a different fixed random seed — on MNIST and CIFAR-10. The adversary is assumed to have access to only a single model. The prediction accuracies on clean data are reported as 96% and 45% on MNIST and CIFAR-10 with single PPD model. The accuracy increases to 97.75% and 48% on MNIST and CIFAR-10 when an ensemble of 10 PPD models are used. The authors claim that 10 models ensure good enough performance. On the other hand, in the case of adversarial examples, an ensemble of 50 PPD models achieve more than 97.4% accuracy on MNIST where the perturbation is less than or equal to 0.1. Similarly, for CIFAR-10, the ensemble can at the very least obtain an accuracy of 45.1% for at most 0.1 perturbation. As expected, the accuracy decreases as the perturbation is increased.

3 Methods

3.1 Original PPD, as Proposed in ICLR

The initial goal of this project is to reproduce the experiments defined in the original paper. Therefore, identical to the ICLR paper, our PPD model is also a simple 3 layered MLP (800→300→10), where the input image is shuffled and its phase component is passed to the network. The shuffling and phase extraction operations are discussed in detail in Section 2.2 and shown in Figure 1. No dropouts or batch normalization layers were used, since the paper did not mention any. As the PPD

paper reports that using an ensemble of 10 models is sufficient enough for a defense mechanism, we replicated their approach using an ensemble of 10 PPD models. However, the original PPD paper does not define their ensemble method, so we chose to use majority voting rule.

Similar to the ICLR paper, we tested the ensemble model against 7 adversarial attacks in two norms, l_2 and l_∞. Attacks on l_∞ norm are Fast Gradient Sign Method (FGSM), Basic Iterative Method (BIM), Projected Gradient Descent ($l_\infty-$PGD), Momentum Iterative Method (MIM) whereas l_2 attacks include Fast Gradient Method (l_2-FGM), (l_2-PGD), Carlini and Wagner Method (CW). They were implemented using the Cleverhans API developed in Papernot et al. (2018).

3.2 Experiments on Single PPD

On top of the experiments defined in the original paper, additional experiments were devised to test the efficiency of PPD separate from the ensemble model. Unless otherwise specified, the adversarial examples are obtained with FGSM with ℓ_∞ norm of 0.4 for simplicity.

Extension to CNN: ICLR paper's classification accuracy on clean CIFAR-10 images is very low (45%). To improve it, we created a simple CNN from [Keras (2018)], described in Figure 4, and combined it with the PPD pipeline.

Effect of hidden permutation seed: The effectiveness of the encryption in the form of shuffling is tested on a single PPD model (MLP and CNN). The goal is to observe whether the concealment of the seed from the adversary has a significant effect, as claimed by the authors.

Effect of permutation vs. phase: PPD is made up of 2 operations: permutation and phase extraction. To determine individual contributions of permutation and phase, we implemented one of these operations at a time for a CNN or MLP based model.

Experimenting on larger dataset: ICLR paper reports accuracies for MNIST and CIFAR-10 datasets. In order to better judge the defense of PPD, we tested PPD on MESSIDOR. It consists of 957 low retina images in the train set and 253 in test set. [Decencière et al. (2014)]

3.3 Variations of PPD

We have explored several variations of PPD in an attempt to improve its performance.

Moving pixel2phase to hidden layers: In the original design, the permuted image is passed through pixel2phase layer before going through the neural network. We moved the pixel2phase layer in between hidden layers so that more information from the input data is retained and used in the decision process. We investigated this using both MLP and CNN based models, with respectively 3 and 6 different versions as shown in Figures 3 and 4. We removed pixel permutation step in the CNN based models to see how retaining spatial information would affect the performance of the convolutional layers.

Adding pixel2magnitude: The original model only keeps the phase values after the Fourier transformation, and disregards the spatial information. We examined the effect of re-introducing color information by including the magnitude of Fourier transform through a pixel2magnitude layer. The permuted image is passed through pixel2phase and pixel2magnitude simultaneously. The phase and magnitude results are then concatenated and fed through the PPD pipeline.

4 Results

4.1 Reproducing PPD Results

The experiments described in the PPD were replicated to the best of our capabilities, except for CW due to memory limitations. In Table 1, the accuracies of our replicated ensemble using 10 PPD models are compared to the accuracies for an ensemble of 50 models reported in the original PPD paper.

For the MNIST dataset, when l_∞ attacks are used, the replicated experiments back the results of the original paper as the reported accuracy differences are under 1.1% change for perturbations (ϵ) under 0.3. For higher perturbation levels, the difference between the replicated and original work is

	MNIST					CIFAR-10				
l_∞ **perturbation**	0.03	0.1	0.2	0.3	0.4	0.03	0.1	0.2	0.3	0.4
replicated FGSM	98.04%	97.76%	97.10%	**92.71%**	83.73%	52.65%	**44.59%**	**38.44%**	34.12%	**28.87%**
original FGSM	97.8%	97.6%	97.1%	95.4%	91%	48.2%	47.9%	45.3%	39.7%	31.1%
replicated BIM	97.99%	97.80%	97.49%	96.72%	93.84%	**52.55%**	47.47%	42.45%	38.32%	34.61%
original BIM	97.8%	97.6%	96.7%	95.2%	91%	48.2%	47.7%	45.2%	39%	30.1%
replicated PGD	98.06%	97.86%	97.26%	96.25%	95.73%	52.42%	48.01%	42.27%	39.06%	38.78%
original PGD	97.8%	97.7%	97%	95.2%	91.3%	48.3%	47.6%	45.4%	41.6%	37.1%
replicated MIM	**97.98%**	**97.71%**	**96.95%**	93.91%	**85.65%**	52.96%	45.38%	39.61%	**33.90%**	29.53%
original MIM	**97.8%**	**97.6%**	**95.8%**	**87.4%**	**67.5%**	**48.2%**	**47.5%**	**40.4%**	**26.1%**	**15.6%**
l_2 **perturbation**	0.1	0.7	1.1	3.2	4	0.3	2	4	6.5	10.5
replicated FGM	**8.23%**	**7.92%**	**7.85%**	**7.76%**	**7.73%**	**48.66%**	**48.38%**	**48.53%**	**48.53%**	**48.51%**
original FGM	97.8%	97.8%	97.8%	97.3%	97.1%	48.1%	48.1%	48%	47.3%	45.4%
replicated PGD	97.95%	97.87%	98.06%	98.04%	97.98%	55.19%	54.97%	54.68%	54.60%	54.38%
original PGD	97.8%	97.8%	97.8%	**97.3%**	**96.7%**	48.3%	48.3%	**47.5%**	**45.7%**	**39.3%**
replicated CW	Memory error									
original CW	**97.7%**	**97.4%**	**97.7%**	97.7%	97.8%	**48%**	**48%**	47.9%	46.5%	39.5%

Table 1: Comparison of the prediction accuracies reported in the original paper and our replicated network. Accuracies shown in green indicate that our ensemble of 10 PPD models was able to perform better than their counterpart reported in the original paper. The lowest accuracies are also highlighted in bold.

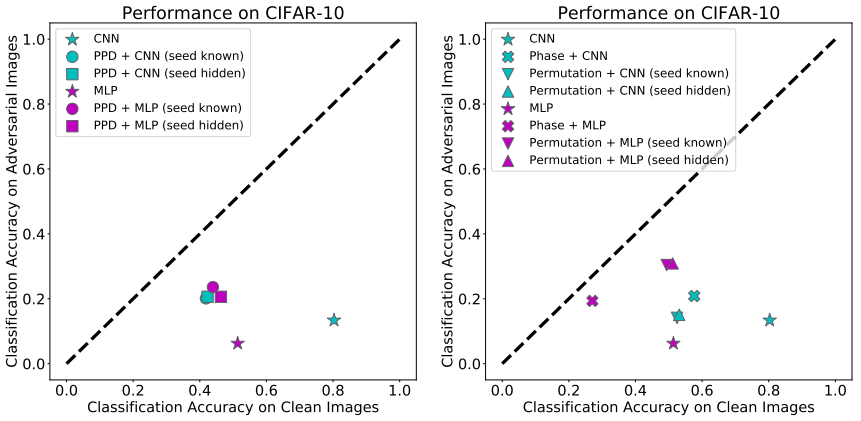
most noticeable in MIM attack where for ϵ equal to 0.3 and 0.4, our implementation achieves a better accuracy with 6.5% and 18.1% increase respectively. For the more complicated dataset of CIFAR-10, the original PPD paper accuracies differ from our own by at most 5% for most of the attacks for various ϵ values. The exceptions are FGSM with ϵ values of 0.2 and 0.3, MIM application with a perturbations of 0.3 and 0.4 whose accuracies differed by 6.9%, 5.6%, 7.8% and 13.93% respectively. It is important to note that for MIM, our implementation performance was better than the original with the aforementioned difference.

However, a comparison on the l_2 attack, FGM, implementations for MNIST uncover a discrepancy where the highest accuracy according to our initial experiment is 8.23% whereas the lowest accuracy listed in the original paper was 97.1%. This might be due to several factors including the possible difference in ensemble method or the number of models used in the ensemble or implementation issues in our experiment. On the other hand, rest of the experiments regarding l_2 attacks on both MNIST and CIFAR-10 proved that the original accuracies mentioned are valid as all of our implementation was able to achieve higher accuracies using PPD as the defense.

4.2 Testing a Single PPD

We analyze a single PPD model's performance on the CIFAR-10 dataset, and focus on the ℓ_∞ FGSM attack with $\epsilon = 0.4$. Figure 2a summarizes the performance of the following six models on CIFAR-10: (DNN type, PPD specifications) $\sim$ {MLP, CNN} $\times$ {no PPD, PPD (seed known to adversary), PPD (seed hidden to adversary)}. On the other hand, Figure 2b summarizes the performance of the following eight models on CIFAR-10: (DNN type, PPD components) $\sim$ {MLP, CNN} $\times$ {no PPD, Phase, Permutation (seed known to adversary), Permutation (seed hidden to adversary)}.

Lastly, we investigated the performance of the PPD model on MESSIDOR dataset as it is more complex and larger. Without random permutaiton, MESSIDOR has achieved 83.4% accraucy using a pretrained Inception v3 model [Asim Smailagic (2018)]. On the other hand, with the PPD model, accuracy was 55.47% for clean images. After applying FGSM in l_∞ and l_2 norms, the accuracy



(a) With and without PPD.

(b) With and without permutation/phase transformation

Figure 2: MLP and CNN performance on CIFAR-10

went down to 44.57% and 44.37% respectively. Due to the complex nature of the MESSIDOR, basic MLP of PPD couldn't extract the features of the images and underfit.

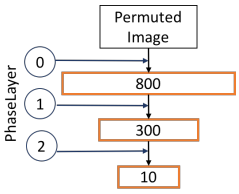


Figure 3: MLP based model architectures. Having PhaseLayer at position 0 is equivalent to the original PPD model.

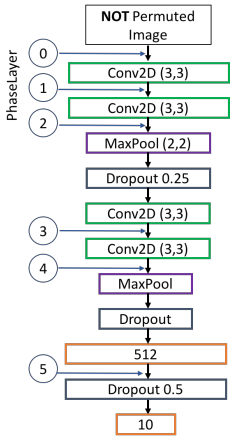


Figure 4: CNN based model architectures that were explored.

		MNIST Acc.		CIFAR10 Acc.	
	PhaseLayer	Clean	Adv.	Clean	Adv.
MLP	0	0.953	0.697	0.446	0.238
	1	0.979	0.863	0.492	0.290
	2	0.977	0.841	0.497	0.329
CNN	0	Skipped		0.554	0.214
	1			NaN	NaN
	2			0.631	0.086
	3			0.188	0.125
	4			0.581	0.071
	5			0.738	0.13

Table 2: Classification accuracy of clean and adversarial MNIST/CIFAR10. PhaseLayer was inserted into various points of the MLP and CNN.

4.3 Improving PPD

Table 2 lists the results of moving the pixel2phase layer between hidden layers while Figures 3 and 4 refer to the different orientations of pixe2phase within the network. The traditional implementation of the PPD model is where pixel2phase is in orientation 0. As can be inferred from Table 2, moving the pixel2phase layer to position 1 and 2 outperforms the performance of the original PPD model when MLP is used on MNIST and CIFAR-10 respectively. On the clean examples, the improvement is only 2.6% and 5.1% for each dataset. On the other hand, for adversarial examples, there is a substantial increase in the accuracy by 16.6%and 9.1% on MNIST and CIFAR-10.

For the CNN based PPD model, we only investigated the effects on CIFAR-10. We noticed that introducing pixel2phase in between 2 convolution layers result in quite low accuracies both on clean and adversarial data. The best configuration of pixel2phase seems to be in position 5 for the clean images, with an accuracy of 73.8%. However, to get the best performance against adversarial attack, pixel2phase should be positioned at 0 which achieves an accuracy of 21.4%. When

compared to the MLP versions, CNN based models underperform against adversaries when coupled with pixel2phase. The highest accuracy reached with a CNN based model is 21.4% while an MLP based model achieved 23.8% in its worst configuration of pixel2phase for CIFAR-10.

Another novelty defined in Section 3.3 is to make use of the color information via pixel2magnitude. As the accuracy of MNIST decreased on both clean and adversarial data after the addition of magnitude, no further test were carried out on CIFAR-10. The performance decrease was 1.2% and 6.3% for clean and adversarial data respectively.

5 DISCUSSION

Importance of hidden permutation seed and the effect of ensemble methods: The paper reports that a single PPD + 3 layer-MLP model was able to achieve 45% classification accuracy on clean CIFAR-10, and the 50 ensemble models were able to achieve 31.1% accuracy under attack. They did not report the performance of a single PPD + MLP model under attack, but suggested that their experimental setup was almost white-box except for the concealment of the permutation seed, and hiding the seed was crucial to the defense. But our experiments in figures 2a and 2b shows that both MLP and CNN models with seed unknown to the adversary (squares) perform almost identically to the models with seed known to the adversary (circles).

This raises questions as to whether the paper is making fair claims, i.e. whether their experimental design properly decouples the effect of ensemble learning from the effect of PPD, or whether their attack setting is truly white-box except for the permutation seed. In order to properly consider ensemble methods, we believe that the adversary should be either 1) given an ensemble model and their 50 seeds, or 2) be informed that there are 50 models, and be allowed to create smart adversarial examples that are "averaged" over many different random permutations.

Accuracy on clean images: The PPD model is a DNN designed and trained to be robust in image classification tasks under adversarial attack. When evaluating a defensive model like PPD, it is important to consider its performance in two aspects: robustness against adversarial attacks, and classification accuracy on clean images. In that regard, the authors seem to gloss over the fact that the classification accuracy on clean CIFAR-10 is very poor, when Benenson (2016) and Graham (2014) has reported accuracy of 96.53% on the same dataset already in 2016. Sacrificing the accuracy on clean images in order to gain robustness in an adversarial setting is expected to some degree, and is often a conscious design choice. However, an accuracy as low as 45% on CIFAR-10 casts doubt into whether the PPD method would actually be useful in practice.

Extension of PPD to CNN: This was our first attempt to address the PPD's low accuracy on clean images. In Figure 2a, note that without PPD (stars), there is a huge gain in the classification accuracy on clean images (horizontal axis) from using CNN over MLP, as expected. However, this gain is clearly absent in the PPD models, which affirms our concerns about the practicality of vanilla PPD. We believe that this is because permuting the pixels invalidates the shift-invariance assumption exploited by CNNs, and because we are only using the phase information.

This point is reinforced by Figure 2b, in the comparison between naive models (stars) and models with just permutation (triangle), and between naive models (stars) and models with just phase (crosses). For both MLP and CNN, working with just phase information increases the adversarial accuracy while decreasing performance in clean images. This may be because the perturbation is distributed to nearby pixels, while disregarding magnitude leads to the loss of color information in CIFAR-10. It is possible that this decrease in performance will not be observed in the black and white MNIST images, but we have not tested this. The effect of permuting pixels on CNN is similar to that of pixel2phase, but the decrease in clean image accuracy is almost negligible in MLP. This is as expected, since we concatenate the pixels into a vector in MLP anyways. In CNN, however, the position of pixels are essential to detection of meaningful local features.

Moving pixel2phase layer: Table 2 showcases the performance of our variant of PPD when the pixel2phase layer is moved into different positions. Moving the pixel2phase further into the MLP consistently increases the accuracy on both dataset substantially (16.6% and 9.1% on MNIST and CIFAR-10). On the other hand, by eliminating the pixel permutation step before convolutional

layers, the CNN models (PhaseLayer 2, 4, 5) performed better on clean data, but became more vulnerable to adversarial attacks. Also, having pixel2phase between two consecutive convolutional layers in CNN (PhaseLayer 1, 3) resulted in NaN weights or very low classification accuracies in both clean and adversarial CIFAR-10. We tried to debug the error, but are not sure where the error in backpropagation comes from for this specific case.

When we take the phase of a signal, it tells us how the sine waves with different frequencies line up with one another. For MLP and CNN (excluding PhaseLayer 1, 3), treating the outputs of neurons at a hidden layer as a signal and taking the phase keeps information about the differences between the neuron outputs (i.e. the *edge* or *jump* in the signal). And we think that keeping the pattern information embedded in phase and discarding the magnitude acts as a regularization method, leading to improved classification accuracies. And since we cannot recover information that is lost in the earlier layers, having pixel2phase in the deeper layers performed better.

Giving more information to PPD: Our initial hypothesis was that providing additional information to PPD would increase the prediction accuracy, at least for the clean images. However, experiments on MNIST where both phase and magnitude is provided to the DNN showed a decrease in the performance when compared to using purely phase. Since MNIST is an easier dataset to work on, and phase output contains the relational information necessary for classification, magnitude mostly became a source of noise. Disregarding magnitude seemed to help the model generalize better for both clean and adversarial images.

Limitations of our experiments: Due to limited time and computational resources, each of our experiments were repeated no more than a couple of times. Therefore, although we have informally observed similar trends in our repetitions, we can not make any rigorous statistical claims.

Suggestions for future work: We did not have a chance to try this idea due to time limitations, but introducing permutation between hidden layers could be interesting. The linear operation should not interfere with network training, but hiding the encryption key from the adversary could mess up their backpropagation for gradient based attacks. Although we acknowledge that this might not be a feasible defense against adversaries capable of inferring approximate gradients, we believe it might make the approximation process harder. Furthermore, using an ensemble method that is not deterministic can improve the performance of PPD models as probabilistic methods are harder to deceive due to the inherent randomness. Potentially, employing a boosting algorithm such as AdaBoost with PPD as its weak classifiers could improve its performance. Boosting algorithms achieve better classification by finding unique combinations of weak classifiers with low performance —as is the case with PPD. Finally, better denoising techniques that work on the frequency domain or better encryption methods that keeps the certain shapes intact can be investigated.

REFERENCES

Anonymous. Ppd: Permutation phase defense against adversarial examples in deep learning. In *Submitted to International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HkElFj0qYQ`. under review.

Aurelio Campilho Pedro Costa Devesh Walawalkar Kartik Khandelwal Mostafa Mirshekari Jonathon Fagert Adrian Galdran Susu Xu Asim Smailagic, Hae Young Noh. Medal: Accurate and robust deep active learning for medical image analysis. In *Accepted to ICMLA 2018*, 2018. URL `https://arxiv.org/ftp/arxiv/papers/1809/1809.09287.pdf`.

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *CoRR*, abs/1707.07397, 2017. URL `http://arxiv.org/abs/1707.07397`.

Rodrigo Benenson. What is the class of this image? discover the current state of the art in objects classification. `http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html#43494641522d3130`, 2016.

Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *IN PROCEEDINGS OF THE 25TH IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR 2012*, pp. 3642–3649, 2012.

Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, Ali Erginay, Béatrice Charton, and Jean-Claude Klein. Feedback on a publicly distributed image database: The messidor database. *Image Analysis Stereology*, 33(3):231–234, 2014. ISSN 1854-5165. doi: 10.5566/ias.1155. URL https://www.ias-iss.org/ojs/IAS/article/view/1155.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

Benjamin Graham. Fractional max-pooling, 2014.

Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SyJ7ClWCb.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pp. 506–519, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4944-4. doi: 10.1145/3052973.3053009. URL http://doi.acm.org/10.1145/3052973.3053009.

Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6199.