Adversarial and Clean Data Are Not Twins

Zhitao Gong

gong@auburn.edu

Auburn University

Auburn, AL, USA

Wenlu Wang

wenlu.wang@tamucc.edu

Texas A&M University-Corpus Christi

Corpus Chrsiti, TX, USA

ABSTRACT

Adversarial attack has cast a shadow on the massive success of deep neural networks. Despite being almost visually identical to the clean data, the adversarial images can fool deep neural networks into the wrong predictions with very high confidence. Adversarial training, as the most prevailing defense technique, suffers from class-wise unfairness and model-dependent challenges. In this paper, we propose to detect and eliminate adversarial data in databases prior to data processing in supporting robust and secure AI workloads. We empirically show that we can build a binary classifier separating the adversarial apart from the clean data with high accuracy. We also show that the binary classifier is robust to a second-round adversarial attack. In other words, it is difficult to disguise adversarial samples to bypass the binary classifier. Furthermore, we empirically investigate the generalization limitation which lingers on all current defensive methods, including the binary classifier approach. And we hypothesize that this is the result of the intrinsic property of adversarial crafting algorithms. Our experiments ascertain that adversarial and clean data are two different datasets that can be separated with a binary classifier, which can serve as a portable component to detect and eliminate adversarial data in an end-to-end data management pipeline.

ACM Reference Format:

Zhitao Gong and Wenlu Wang. 2023. Adversarial and Clean Data Are Not Twins. In Sixth International Workshop on Exploiting Artificial Intelligence Techniques for Data Management (aiDM ’23), June 18, 2023, Seattle, WA, USA. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3593078.3593935

1 INTRODUCTION

Deep neural networks have been successfully adopted in many critical areas, e.g., skin cancer detection [3], auto-driving [13], traffic sign classification [2], natural language processing [8, 16, 17], genomics [20], etc. A recent study [15], however, discovered that deep neural networks are susceptible to adversarial data. Figure 1 shows an example of adversarial images generated via fast gradient sign method [6, 7] on MNIST. As we can see that although the adversarial and original clean images are almost identical from the perspective of human beings, the deep neural network will produce wrong predictions with very high confidence. Similar techniques can easily fool the imaging system into mistaking a stop sign for a yield sign, a dog for an automobile, for example. When leveraged by malicious users, these adversarial images pose a great threat to deep neural network systems.

There have been defensive methods such as adversarial training [5, 7], which was shown to greatly enhance the model’s robustness to adversarial samples. However, adversarial training introduces a disparity of accuracies among different classes of data. This unfairness can occur even in balanced datasets but is absent in models trained on clean data [1, 19]. In other words, there is a trade-off between model robustness and model fairness when using adversarial training. To address this issue, we propose an adversarial data detection approach as an alternative to adversarial training in supporting robust and secure AI systems. Our proposed approach is orthogonal to the AI applications and can be easily integrated into an end-to-end data management pipeline.

Although adversarial and clean images appear visually indiscernible, their subtle differences can successfully fool deep neural networks. This means that deep neural networks are sensitive to these subtle differences. Inspired by the aforementioned observations, we have two intuitive research questions: (1) does all adversarial training effectively improve the robustness of deep models? and (2) can we leverage these subtle differences to detect adversarial data? Our experiment suggests the answers are negative and positive. In this paper, we demonstrate that not all adversarial retraining contributes to the robustness of deep models, and a binary classifier can separate the adversarial from the original clean images with very high accuracy (over 99%). We also experimentally investigate the generalization limitations of our binary classifier approach, which is comparable to adversarial training. We empirically investigate the limitation and propose the hypothesis that the adversarial and original datasets are, in effect, two completely different datasets, despite being visually similar. Based on the experimental observations, it is feasible to deploy a portable binary classifier to detect adversarial data.

This article is organized as follows. In Section 2, we give an overview of the current research in adversarial attack and defense. Then, it is followed by a brief summary of the state-of-the-art adversarial crafting algorithms and our proposed binary approach.
in Section 3. Section 4 presents our experimental validation of the limitations of retraining adversarial samples, the possibility of detecting adversarial data, and the binary classifier being robust towards second-round adversarial attack. And we conclude in Section 5 that we certify the potential of detecting adversarial samples prior to data processing and the proposed binary classifier that is orthogonal to the AI application can be integrated into an end-to-end data management pipeline.

2 RELATED WORK
The adversarial image attack on deep neural networks was first investigated in [15]. The authors discovered that when added some imperceptible carefully chosen noise, an image may be wrongly classified with high confidence by a well-trained deep neural network. They also proposed an adversarial crafting algorithm based on optimization. We will briefly summarize it in section 3. They also proposed the hypothesis that the adversarial samples exist as a result of the high nonlinearity of deep neural network models.

However, [4] proposed a counter-intuitive hypothesis explaining the cause of adversarial samples. They argued that adversarial samples are caused by the models being too linear, rather than nonlinear. They proposed two adversarial crafting algorithms based on this hypothesis, i.e., fast gradient sign method (FGSM) and least-likely class method (LLCM) [4]. The least-likely class method is later generalized to target class gradient sign method (TGSM) in [6], [11] proposed another gradient based adversarial algorithm, the Jacobian-based saliency map approach (JSMA) which can successfully alter the label of an image to any desired category.

Adversarial training [5, 7] was also shown to greatly enhance the model’s robustness to adversarial. However, adversarial training also introduces a disparity of accuracies among different classes of data – i.e., class-wise discrepancy. This type of unfairness happens in balanced datasets and does not exist in clean data trained models [1, 14, 19]. There is a trade-off between model robustness and model fairness. The adversarial samples have been shown to be transferable among deep neural networks [6, 15]. This poses a great threat to current learning systems in that the attacker needs not the knowledge of the target system. Instead, the attacker can train a different model to create adversarial samples which are still effective for the target deep neural networks. What’s worse, [10] has shown that adversarial samples are even transferable among different machine learning techniques, e.g., deep neural networks, support vector machine, decision tree, logistic regression, etc. The observation inspired us to propose a general binary classifier that separates adversarial and clean data.

3 CRAFTING ADVERSARIALS
There are mainly two categories of algorithms to generate adversarial samples, model independent and model dependent. We briefly summarize these two classes of methods in this section. By conventions, we use $X$ to represent input image set (usually a 3-dimension tensor), and $Y$ to represent the label set, usually one-hot encoded. Lowercase represents an individual data sample, e.g., $x$ for one input image. Subscript to data samples denotes one of its elements, e.g., $x_i$ denotes one pixel in the image, $y_i$ denotes probability for the $i$-th target class. $f$ denotes the model, $\theta$ the model parameter, $J$ the loss function. We use the superscript $\text{adv}$ to denote adversarial related variables, e.g., $x^{\text{adv}}$ for one adversarial image. $\delta x$ denotes the adversarial noise for one image, i.e., $x^{\text{adv}} = x + \delta x$. For clarity, we also include the model used to craft the adversarial samples where necessary, e.g., $x^{\text{adv}(f_i)}$ denotes the adversarial samples created with model $f_i$. $\mathcal{D}$ denotes the image value domain, usually $[0,1]$ or $[0, 255]$. And $\epsilon$ is a scalar controlling the scale of the adversarial noise, another hyper-parameter to choose.

3.1 Model Independent Method
A box-constrained minimization algorithm based on L-BFGS was the first algorithm proposed to generate adversarial data [15]. Concretely we want to find the smallest (in the sense of $L_\infty$-norm) noise $\delta x$ such that the adversarial image belongs to a different category, i.e., $f(x^{\text{adv}}) \neq f(x)$.

$$
\delta x = \arg\min_r \|r\|_\infty + J(x + r, y^{\text{adv}})
\text{s.t. } x + r \in \mathcal{D}
$$

(1)

3.2 Model Dependent Methods
There are mainly three methods that rely on model gradients, i.e., fast gradient sign method (FGSM) [6], target class method [6, 7] (TGSM) and Jacobian-based saliency map approach (JSMA) [11]. We will see in Section 4 that despite that they all produce highly disguising adversarials, FGSM and TGSM produce compatible adversarial datasets which are complete different from adversarials generated via JSMA.

Fast Gradient Sign Method (FGSM). FGSM tries to modify the input towards the direction where $J$ increases, i.e., $\frac{dJ(x, y^{\text{adv}})}{dx}$, as shown in Equation 2.

$$
\delta x = \epsilon \text{sign} \left( \frac{dJ(x, y)}{dx} \right)
$$

(2)

Originally [6] proposes to generate adversarial samples by using the true label i.e., $y^{\text{adv}} = y^{\text{true}}$, which has been shown to suffer from label leaking problem [7]. Instead of true labels, [7] proposes to use the predicted label, i.e., $\hat{y} = f(x)$, to generate adversarial examples.

This method can also be used iteratively as shown in Equation 3. Iterative FGSM has a much higher success rate than the one-step FGSM. However, the iterative version is less robust to image transformation [6].

$$
x_k^{\text{adv}} = x_k^{\text{adv}} + \epsilon \text{sign} \left( \frac{dJ(x_k^{\text{adv}}, \hat{y})}{dx} \right)
$$

(3)

Target Class Gradient Sign Method (TGSM). This method tries to modify the input towards the direction where $p(y^{\text{adv}} | x)$ increases.

$$
\delta x = -\epsilon \text{sign} \left( \frac{dJ(x, y^{\text{adv}})}{dx} \right)
$$

(4)

Originally this method was proposed as the least-likely class method [6] where $y^{\text{adv}}$ was chosen as the least-likely class predicted by the
model as shown in Equation 5.

\[ y^{\text{adv}} = \text{OneHotEncode}(\text{argmin } f(x)) \]  \hspace{2cm} (5)

And it was extended to a more general case where \( y^{\text{adv}} \) could be any desired target class [7].

**Jacobian-based Saliency Map Approach (JSMA).** Similar to the target class method, JSMA [11] allows to specify the desired target class. However, instead of adding noise to the whole input, JSMA changes only one pixel at a time. A saliency score is calculated for each pixel and with the highest score is chosen to be perturbed.

\[ s(x_i) = \begin{cases} 0 & \text{if } s_f < 0 \text{ or } s_o > 0 \\ 1 & \text{otherwise} \end{cases} \]

\[ s_f = \frac{\partial y_f}{\partial x_i}, \quad s_o = \sum_{j \neq i} \frac{\partial y_j}{\partial x_i} \]  \hspace{2cm} (6)

Concretely, \( s_f \) is the Jacobian value of the desired target class \( y_f \) w.r.t an individual pixel, \( s_o \) is the sum of Jacobian values of all non-target classes. Intuitively, the saliency score indicates the sensitivity of each output class w.r.t each individual pixel. And we want to perturb the pixel towards the direction where \( p(y_f | x) \) increases the most.

### 3.3 Proposed Detection Approach

Generally, we follow the steps below to test the effectiveness and limitation of the binary classifier approach.

1. **Train a deep neural network** \( f_1 \) on the original clean training data \( X_{\text{train}} \), and craft adversarial dataset from the original clean data, \( X_{\text{train}} \rightarrow X_{\text{train}}^{\text{adv}(f_1)} \), \( X_{\text{test}} \rightarrow X_{\text{test}}^{\text{adv}(f_1)} \). \( f_1 \) is used to generate the attacking adversarial dataset which we want to filter out.

2. **Train a binary classifier** \( f_2 \) on the combined (shuffled) training data \{\( X_{\text{train}}, X_{\text{test}}^{\text{adv}(f_1)} \}\), where \( X_{\text{train}} \) is labeled 0 and \( X_{\text{test}}^{\text{adv}(f_1)} \) labeled 1.

3. **Test the accuracy of** \( f_2 \) on \( X_{\text{test}}^{\text{adv}(f_1)} \) and \( X_{\text{test}}^{\text{adv}(f_1)} \), respectively.

4. **Construct second-round adversarial test data**, \{\( X_{\text{test}}, X_{\text{test}}^{\text{adv}(f_1)} \)\} \( \rightarrow \) \{\( X_{\text{test}}, X_{\text{test}}^{\text{adv}(f_1)} \)\} and test \( f_2 \) accuracy on this new adversarial dataset. Concretely, we want to test whether we could find adversarial samples 1) that can successfully bypass the binary classifier \( f_2 \), and 2) that can still fool the target model \( f_1 \) if they bypass the binary classifier. Since adversarial datasets are shown to be transferable among different machine learning techniques [10], the binary classifier approach will be seriously flawed if \( f_2 \) failed this second-round attacking test.

### 4 EXPERIMENT

#### 4.1 Efficiency and Robustness of the Classifier

We evaluate the binary classifier approach on MNIST, CIFAR10, and SVHN datasets. Of all the datasets, the binary classifier achieved accuracy over 99% and was shown to be robust to a second-round adversarial attack. The results are summarized in Table 1. Each column denotes the model accuracy on the corresponding dataset. The direct conclusions from Table 1 are summarized as follows.

1. **Accuracy on** \( X_{\text{test}} \) and \( X_{\text{adv}(f_1)} \) suggests that the binary classifier is very effective at separating adversarial from the clean dataset. Actually during our experiment, the accuracy on \( X_{\text{test}} \) is always near 1, while the accuracy on \( X_{\text{test}}^{\text{adv}(f_1)} \) is either near 1 (successful) or near 0 (unsuccessful). This means that the classifier either successfully detects the subtle difference completely or fails completely. We did not observe any values in between.

2. **Accuracy on** \( \{X_{\text{test}}^{\text{adv}(f_1)}\}^{\text{adv}(f_2)} \) suggests that we were not successful in disguising adversarial samples to bypass the classifier. In other words, the binary classifier approach is robust to a second-round adversarial attack.

3. **Accuracy on** \( \{X_{\text{test}}\}^{\text{adv}(f_1)} \) \( \text{adv}(f_2) \) suggests that in case of the second-round attack, the binary classifier has very high false negative. In other words, it tends to recognize them all as adversarial. This, does not pose a problem in our opinion. Since our main focus is to block adversarial samples.

#### 4.2 Generalization Limitation

Before we conclude too optimistic about the binary classifier approach performance, however, we discover that it suffers from the generalization limitation.

1. **When trained to recognize adversarial dataset generated via FGSMS/TGSM**, the binary classifier is sensitive to the hyper-parameter \( \epsilon \).

2. **The binary classifier is also sensitive to the adversarial crafting algorithm.**

In our experiment, the aforementioned limitations also apply to adversarial training [5, 7] and defensive distillation [12].

Table 2 summarizes our tests on CIFAR10. For brevity, we use \( f_2 | \epsilon = \epsilon_0 \) to denote that the classifier \( f_2 \) is trained on adversarial data generated on \( f_1 \) with \( \epsilon = \epsilon_0 \). The binary classifier is trained on mixed clean data and adversarial dataset which is generated via

The code to reproduce our experiment is publicly available at 1.

### Table 1: Accuracy on adversarial samples generated with FGSM/TGSM.

| Dataset | \( X_{\text{test}} \) | \( X_{\text{adv}(f_1)} \) | \( X_{\text{adv}(f_1)} \) | \( X_{\text{adv}(f_1)} \) | \( X_{\text{adv}(f_1)} \) |
|---------|------------------|------------------|------------------|------------------|------------------|
| MNIST  | 0.9914 | 0.0213 | 1.00 | 1.00 | 0.00 | 1.00 |
| CIFAR10 | 0.8279 | 0.1560 | 0.99 | 1.00 | 0.01 | 1.00 |
| SVHN   | 0.9787 | 0.2453 | 1.00 | 1.00 | 0.00 | 1.00 |

### Table 2: \( \epsilon \) sensitivity on CIFAR10

| \( \epsilon \) | \( X_{\text{test}} \) | \( X_{\text{adv}(f_1)} \) |
|-------------|------------------|------------------|
| 0.3 | 0.9996 | 1.0000 |
| 0.1 | 0.9996 | 1.0000 |
| 0.03 | 0.9996 | 0.9997 |
| 0.01 | 0.9996 | **0.0030** |

1https://github.com/VV123/adversarial-classifier1
We can generate adversarial data samples by a linear combination of the direction computed by FGSM and another random orthogonal direction, as illustrated in a church plot [18] Figure 2. Figure 2 visually shows the effect of adversarial training [7]. Each image represents adversarial samples generated from one data sample, which is represented as a black dot in the center of each image, the last adversarial sample used in adversarial training is represented as an orange dot (on the right of the black dot, i.e., in the direction computed by FGSM). The green area represents the adversarial samples that cannot be correctly recognized without adversarial training but can be correctly recognized with adversarial training. The red area represents data samples that can be correctly recognized without adversarial training only. The white (gray) area represents the data samples that are always correctly (incorrectly) recognized with or without adversarial training.

As we can see from Figure 2, adversarial training does make the model more robust against the adversarial sample (and adversarial samples around it to some extent) used for training (green area). However, it does not rule out all adversarials. There are still adversarial samples (gray area) that are not affected by the adversarial training. Further more, we could observe that the green area largely distributes along the horizontal direction, i.e., the FGSM direction. [9] observed similar results for fooling images with a set of experiments showing, with adversarial training with fooling images, deep neural network models are more robust against a limited set of fooling images, and can still be fooled by other fooling images.

5 CONCLUSION

We show in this paper that not all adversarial retraining contributes to the robustness of deep models. We also notice that it is possible to detect adversarial from the original clean data using a binary classifier. It can serve as an orthogonal preprocessing step without assumptions about the deep model itself. Besides, the binary classifier can be readily deployed without any modification of the underlying systems as a stand-alone detection method and avoid adversarial training that hurts the model’s fairness. However, as we empirically showed in the experiment, the binary classifier approach, defensive retraining, and distillation all suffer from the same level of generalization limitation. For future work, we plan to extend our current work to improve generalization.

Figure 2: Adversarial training [5, 7] does not work. This is a church window plot [18]. Each pixel $(i, j)$ (row index and column index pair) represents a data point $x$ in the input space and $x' = x + \hat{h}e_j + v\varepsilon_i$, where $\hat{h}$ is the direction computed by FGSM and $v$ is a random direction orthogonal to $h$. The $\epsilon$ ranges from $[-0.5, 0.5]$ and $e_{(i, j)}$ is the interpolated value in between. The central black dot $\cdot$ represents the original data point $x$, the orange dot (on the right of the center dot) $\cdot$ represents the last adversarial sample created from $x$ via FGSM that is used in the adversarial training and the blue dot $\cdot$ represents a random adversarial sample created from $x$ that cannot be recognized with adversarial training. The three digits below each image, from left to right, are the data samples that correspond to the black dot, orange dot, and blue dot, respectively. $\Box$ ($\square$) represents the data samples that are always correctly (incorrectly) recognized by the model. $\Box$ represents the adversarial samples that can be correctly recognized without adversarial training only. And $\square$ represents the data points that were correctly recognized with adversarial training only, i.e., the side effect of adversarial training.
REFERENCES

[1] Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. 2021. Robustness may be at odds with fairness: An empirical study on class-wise accuracy. In NeurIPS 2020 Workshop on Pre-registration in Machine Learning. PMLR, 325–342.

[2] Dan Cireşan, Ulrich Meier, Jonathan Masci, and Jurgen Schmidhuber. 2012. Multi-Column Deep Neural Network for Traffic Sign Classification. Neural Networks 32 (2012), 333–338.

[3] Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. 2017. Dermatologist-Level Classification of Skin Cancer With Deep Neural Networks. Nature advance online publication (25 Jan 2017). http://dx.doi.org/10.1038/nature21056 Letter.

[4] I. J. Goodfellow, J. Shlens, and C. Szegedy. 2014. Explaining and Harnessing Adversarial Examples. ArXiv e-prints (Dec. 2014). arXiv:1412.6572 [stat.ML]

[5] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. 2015. Learning With a Strong Adversary. CoRR abs/1511.03034 (2015). http://arxiv.org/abs/1511.03034

[6] A. Kurakin, I. Goodfellow, and S. Bengio. 2016. Adversarial Examples in the Physical world. ArXiv e-prints (July 2016). arXiv:1607.02533 [cs.CV]

[7] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. 2016. Adversarial Machine Learning At Scale. CoRR abs/1611.01236 (2016). http://arxiv.org/abs/1611.01236

[8] Jingjing Li, Wenlu Wang, Wei-Shinn Ku, Yingtao Tian, and Haixun Wang. 2019. Spatialnli: A spatial domain natural language interface to databases using spatial comprehension. In Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. 339–348.

[9] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. 2014. Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images. CoRR abs/1412.1897 (2014). http://arxiv.org/abs/1412.1897

[10] N. Papernot, P. McDaniel, and I. Goodfellow. 2016. Transferability in Machine Learning: From Phenomena To Black-Box Attacks Using Adversarial Samples. ArXiv e-prints (May 2016). arXiv:1605.07277 [cs.CR]

[11] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. 2015. The Limitations of Deep Learning in Adversarial Settings. CoRR abs/1511.07528 (2015). http://arxiv.org/abs/1511.07528

[12] Nicolas Papernot, Patrick Drew McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2015. Distillation As a Defense To Adversarial Perturbations Against Deep Neural Networks. CoRR abs/1511.04508 (2015). http://arxiv.org/abs/1511.04508

[13] Eder Santana and George Hotz. 2016. Learning a Driving Simulator. CoRR abs/1608.01230 (2016). http://arxiv.org/abs/1608.01230

[14] Korn Sooksatra and Pablo Rivas. 2021. Adversarial Training Negatively Affects Fairness. In 2021 International Conference on Computational Science and Computational Intelligence (CSCI) IEEE, 140–145.

[15] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2013. Intriguing Properties of Neural Networks. CoRR abs/1312.6199 (2013). http://arxiv.org/abs/1312.6199

[16] Wenlu Wang, Jingjing Li, Wei-Shinn Ku, and Haixun Wang. 2023. Multilingual spatial domain natural language interface to databases. GeoInformatica (2023), 1–24.

[17] Wenlu Wang, Yingtao Tian, Haixun Wang, and Wei-Shinn Ku. 2020. A natural language interface for database: Achieving transfer-learnability using adversarial method for question understanding. In 2020 IEEE 36th International conference on data engineering (ICDE). IEEE, 97–108.

[18] D. Warde-Farley and I. Goodfellow. 2016. Adversarial Perturbations of Deep Neural Networks.

[19] Han Xu, Xiaorui Liu, Yaxin Li, Anil Jain, and Jiliang Tang. 2021. To be robust or to be fair: Towards fairness in adversarial training. In International Conference on Machine Learning. PMLR, 11492–11501.

[20] Honggang Zhao and Wenlu Wang. 2022. Adversarial Removal of Population Bias in Genomics Phenotype Prediction. In 2022 IEEE International Conference on Data Mining Workshops (ICDMW). IEEE, 334–340.