Detecting Adversarial Examples Using Data Manifolds

Susmit Jha
Computer Science Laboratory
SRI International
Menlo Park, USA
susmit.jha@sri.com

Uyeong Jang
Computer Science Department
University of Wisconsin
Madison, USA
wjang@cs.wisc.edu

Somesh Jha
Computer Science Department
University of Wisconsin
Madison, USA
jha@cs.wisc.edu

Brian Jalaian
Computational and Information Science Directorate
U.S. Army Research Laboratory (ARL)
Adelphi, USA
brian.a.jalaian.civ@mail.mil

***Abstract*—Models produced by machine learning, particularly deep neural networks, are state-of-the-art for many machine learning tasks and demonstrate very high prediction accuracy. Unfortunately, these models are also very brittle and vulnerable to specially crafted adversarial examples. Recent results have shown that accuracy of these models can be reduced from close to hundred percent to below 5% using adversarial examples. This brittleness of deep neural networks makes it challenging to deploy these learning models in security-critical areas where adversarial activity is expected, and cannot be ignored. A number of methods have been recently proposed to craft more effective and generalizable attacks on neural networks along with competing efforts to improve robustness of these learning models. But the current approaches to make machine learning techniques more resilient fall short of their goal. Further, the succession of new adversarial attacks against proposed methods to increase neural network robustness raises doubts about a foolproof approach to robustify machine learning models against all possible adversarial attacks. In this paper, we consider the problem of detecting adversarial examples. This would help identify when the learning models cannot be trusted without attempting to repair the models or make them robust to adversarial attacks. This goal of finding limitations of the learning model presents a more tractable approach to protecting against adversarial attacks. Our approach is based on identifying a low dimensional manifold in which the training samples lie, and then using the distance of a new observation from this manifold to identify whether this data point is adversarial or not. Our empirical study demonstrates that adversarial examples not only lie farther away from the data manifold, but this distance from manifold of the adversarial examples increases with the attack confidence. Thus, adversarial examples that are likely to result into incorrect prediction by the machine learning model is also easier to detect by our approach. This is a first step towards formulating a novel approach based on computational geometry that can identify the limiting boundaries of a machine learning model, and detect adversarial attacks.**

***Index Terms*—trusted machine learning, robustness, adversarial examples, manifolds**

I. Introduction

Deep neural networks have emerged as an ubiquitous choice of representation in machine learning due to the relative ease and computational efficiency of training these models in the presence of large amounts of data. The massive increase in computational power fueled by Moore's law and the emergence of architectures supporting parallel processing at a large scale have made it possible to train these highly nonlinear deep learning networks with thousands of parameters using millions of samples in a reasonable amount of time. This has led to a quantum leap in the prediction accuracy of machine learned models, and encouraged their rapid adoption in different aspects of our social, economic and military infrastructure. Deep neural networks currently provide state-of-the-art results in various applications ranging from computer vision, network security, natural language processing to automatic control.

Unfortunately, these models have been shown to be very brittle and vulnerable to specially crafted adversarial perturbations to examples: given an input x and any target classification t, it is possible to find a new input x that is similar to x but classified as t. These adversarial examples often appear almost indistinguishable from natural data to human perception and are yet incorrectly classified by the neural network. Recent results have shown that accuracy of neural networks can be reduced from close to 100% to below 5% using adversarial examples. This creates a significant challenge in deploying these deep learning models in security-critical domains where adversarial activity is intrinsic, such as Internet of Battle Things, cyber-networks, and surveillance. The use of neural networks in computer vision and speech recognition have brought these models into the center of security-critical systems where authentication depends on these machine learned models. How do we ensure that adversaries in these domains do not exploit the limitations of machine learning models to go undetected or trigger a non-intended outcome? This paper aims at addressing this challenge by detecting adversarial examples.

The authors acknowledge support from the National Science Foundation (NSF) Cyber-Physical Systems #1740079 project, NSF Software & Hardware Foundation #1750009 project, and US ARL Cooperative Agreement W911NF-17-2-0196 on Internet of Battle Things (IoBT).

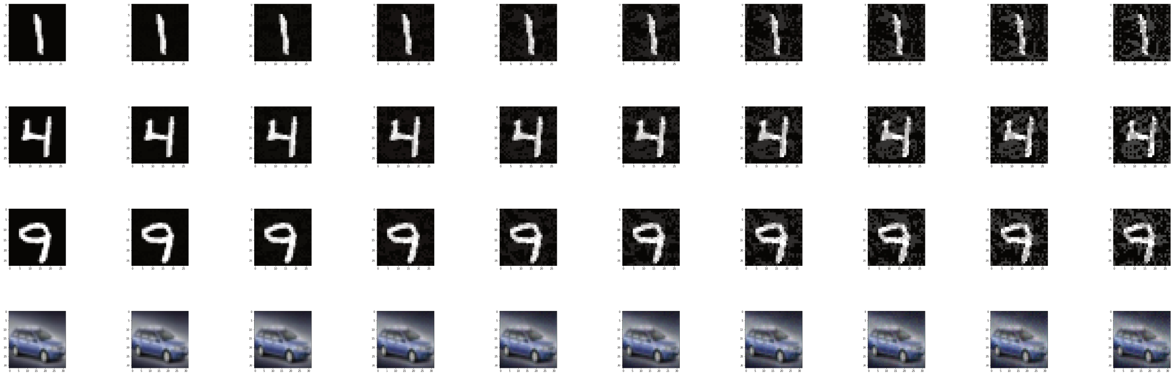

Fig. 1: Original samples (first 3 rows are from MNIST and the 4th row from CIFAR10) perturbed to create adversarial examples with increasing (left to right) norm bound in the Projected Gradient Method implemented within CleverHans system [21].

There has been a recent explosion of methods for adversarial attacks on neural network models along with techniques for making neural networks resilient to attacks. No single resilient mechanism has yet been discovered which can be used against any feasible attack method. This paper takes a different approach to resilience by focusing on the identification of suspicious adversarial examples. The overall idea is to ensure that the machine learning models can identify adversarial attacks, and not provide a prediction on them instead of providing a wrong prediction. An approach to detect these adversarial examples will act as a runtime monitor that finds the limits of the machine learning model.

The central novel contribution of this paper is an approach to detect adversarial examples by identifying a low dimensional manifold in which the training data lie, and then measuring the distance of a new sample to this manifold. Adversarial examples often rely on lying outside this manifold, and since the model was learned using data samples in the manifold, the model naturally mis-predicts on examples farther away from the manifold. In our experiments, we use the CleverHans system [21] and employ the Projected Gradient Descent (PGD) attack method implemented in it. This is an implementation of a very recent attack method described in Madry et al [17]. We control the strength of the attack using one of the parameters in this method ϵ that bounds the maximum distortion of adversarial example compared to the original input. Increasing this norm bound generates adversarial examples with higher confidence. Figure 1 illustrates how the generated adversarial examples change with increase in the norm bound. Even though the change in these images remains small in perception, the accuracy of the neural network drops to below 5%. Our empirical study on MNIST [15] and CIFAR10 [13] datasets suggests that adversarial examples not only lie farther away from the data manifold, but this distance from manifold of an adversarial example increases with the confidence of adversarial examples. Consequently, the detection approach presented in this paper can more easily detect adversarial examples generated with higher norm bound and hence, more likely to cause mis-prediction in the machine learned model. This is a first step towards formulating a computational geometric approach to identifying boundaries of a machine learning model, and using it to detect adversarial attacks.

The rest of the paper is organized as follows. In Section II, we discuss related work on attacks and defense mechanisms including methods for making machine learning algorithm robust or pre-filtering of adversarial examples. We present some background on manifold based learning in Section II, and discuss the proposed approach in Section III. We present the result of our experiments in Section IV before concluding in Section V by discussing limitations of the current approach and ongoing work in this direction.

II. Background and Related Work

Multiple methods have been proposed in literature to generate adversarial examples as well as defend against adversarial examples. Adversarial example generation methods include both white-box and black-box attacks on neural networks [7], [22], [23], [29], targeting feedforward classification networks [4], generative networks [12], and recurrent neural networks [24]. These methods leverage gradient based optimization for normal examples to discover perturbations that lead to mis-prediction - the techniques differ in defining the neighborhood in which perturbation is permitted and the loss function used to guide the search. For example, one of the earliest attacks [7] used a fast sign gradient method (FGMS) that looks for a similar image x' in the L^∞ neighborhood of x. Given a loss function $Loss(x, l)$ specifying the cost of classifying the point x as label l, the adversarial example x' is calculated as

$$x' = x + \epsilon \cdot sign(\nabla_x Loss(x, l_x))$$

FGMS was improved to iterative gradient sign approach (IGSM) in [14] by using a finer iterative optimization strategy where the attack performs FGMS with a smaller step-width α, and clips the updated result so that the image stays within the

ϵ boundary of x. In this approach, the i-th iteration computes the following:

$$x'_{i+1} = clip_{\epsilon,x}(x'_i + \alpha \cdot sign(\nabla_x Loss(x, l_x)))$$

In contrast to FGSM and IGSM, DeepFool [20] attempts to find a perturbed image x' from a normal image x by finding the closest decision boundary and crossing it. In practice, DeepFool relies on local linearized approximation of the decision boundary. Another attack method that has received a lot of attention is Carlini attack that relies on finding a perturbation that minimizes change as well as the hinge loss on the logits (pre-softmax classification result vector). The attack is generated by solving the following optimization problem:

$$min_\delta[||\delta||_2 + c \cdot \max(\ Z(x')_{l_x} - maxZ(x')_i : i \neq l_x, -\kappa\ )]$$

where Z denotes the logits, l_x is the ground truth label, κ is the confidence (raising which will force searcher for larger perturbations), and c is a hyperparameter that balances the perturbation and the hinge loss. Another attack method is projected gradient method (PGM) proposed in [17]. PGD attempts to solve this constrained optimization problem:

$$\max_{||x^{adv}-x||_\infty \leq \epsilon} Loss(x^{adv}, l_x)$$

where S is the constraint on the allowed perturbation usually given as bound ϵ on the norm, and l_x is the ground truth label of x. Projected gradient descent is used to solve this constrained optimization problem by restarting PGD from several points in the l_∞ balls around the data points x. This gradient descent increases the loss function *Loss* in a fairly consistent way before reaching a plateau with a fairly well-concentrated distribution and the achieved maximum value is considerably higher than that of a random point in the data set. In this paper, we focus on this PGD attack because it is shown to be a universal first order adversary [17], that is, developing detection capability or resilience against PGD also implies defense against many other first order attacks.

Defense of neural networks against adversarial examples is more difficult compared to generating attacks. Madry et al. [17] propose a generic saddle point formulation where $\mathcal{D}$ is the underlying training data distribution, $Loss(\theta, x, l_x)$ is a loss function at data point x with ground truth label l_x for a model with parameter θ:

$$\min_\theta E_{(x,y)\sim\mathcal{D}}[\max_{||x^{adv}-x||_\infty \leq \epsilon} Loss(\theta, x^{adv}, l_x)]$$

This formulation uses robust optimization over the expected loss for worst-case adversarial perturbation for training data. The internal maximization corresponds to finding adversarial examples, and can be approximated using IGSM [14]. This approach falls into a category of defenses that use *adversarial training* [27]. Instead of training with only adversarial examples, using a mixture of normal and adversarial examples in the training set has been found to be more effective [20], [29]. Another alternative is to augment the learning objective with a regularizer term corresponding to the adversarial inputs [7]. More recently, logit pairing has been shown to be an effective approximation of adversarial regularization [11].

Another category of defense against adversarial attacks on neural networks are defensive distillation methods [23]. These methods modify the training process of neural networks to make it difficult to launch gradient based attacks directly on the network. The key idea is to use distillation training technique [9] and hide the gradient between the pre-softmax layer and the softmax outputs. Carlini and Wagner [4] found methods to break this defense by changing the loss function, calculating gradient directly from pre-softmax layer and transferring attack from easy-to-attack network to distilled network. More recently, Athalye et al. [1] showed that it is possible to bypass several defenses proposed for the whitebox setting.

Our approach falls into the category of techniques that focus on only detecting adversarial examples. Techniques based on manually identified statistical features [8] or a dedicated learning model [19] trained separately to identify adversarial examples have been previously proposed in literature. These explicit classification methods do not generalize well across different adversarial example generation techniques.

In contrast to these defensive methods, our approach does not require any augmentation of training data, modification of the training process or change in the learned model. The design and training of the neural network is independent to the manifold based filtering developed in this paper. Thus, our approach to detection is orthogonal to learning robust machine learning models and can benefit from these methods. Further, we do not require access to the adversarial example generation method, and thus this defense is likely to generalize well across different attack methods. Our approach relies on just identifying the manifold of typical data which need not be even labeled and hence, this method is more practical in contexts where labeled training data is very difficult to obtain.

A number of explanations for vulnerability of deep neural networks to adversarial samples have been put forward in literature. Szegedy et al. [29] argue that low-probability adversarial "pockets" densely populate the input space and hence, finding an adversarial perturbation is easy. Goodfellow et al. [7] link the ease of generating adversarial examples to the linear nature of deep neural networks. Tanay and Griffin [30] present a boundary tilting perspective that suggests the adversarial examples lie in regions where the classification boundary is close to the manifold of training data. Fienman et al. [5] identify three situations that give rise to adversarial examples: first, when the adversarial example is far away from the data manifold, the second when the submanifolds corresponding to different labels have pockets in them that allow the adversarial example to be on submanifold of the wrong label further away from the classification boundary, and finally when the adversarial example is near two submanifolds as well as close to the decision boundary.

Manifold based explanations for adversarial attacks have also motivated defense mechanisms that try to exploit the manifold property of training data. Bhagoji et al. [3] propose a defense comprising of transforming data to eliminate per-

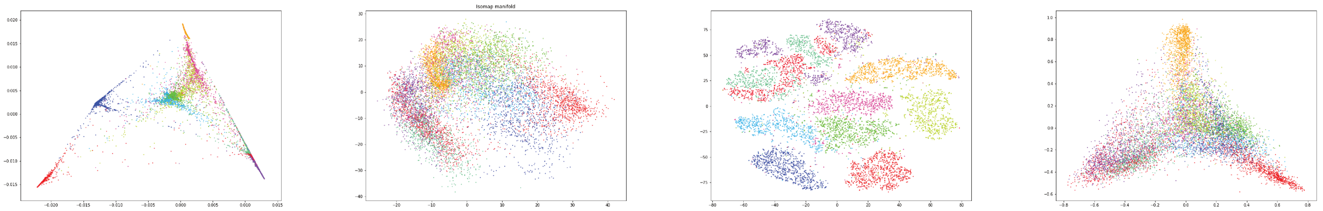

Fig. 2: Left to right: 2D manifold embedding of MNIST dataset using LLE, ISOMAP, t-SNE and Spectral Embedding.

turbations in non-principle components. Meng and Chan [18] consider using autoencoders to project an input image to low dimensional encoding and then decode it back to remove perturbations. Xu et a. [32] propose feature squeezing for images in the input space that effectively forces the data to lie in a low dimensional manifold. Song et al. [28] also note that the distribution of log-likelihoods show considerable difference between perturbed adversarial images and the training data set which can be used to detect adversarial attacks.

III. Approach

Learning manifold in which data points lie has been itself an active area of research [16], [25], [26], [31]. ISOMAP, t-SNE and spectral embedding have been proposed to learn the data manifold. Figure 2 illustrates the learned manifold for MNIST data in 2 dimensions using these methods. The spectral embedding method performs dimensionality reduction in a way that preserves dot products between data points as closely as possible by minimizing $\sum_i (x_i^T x_j - y_i^T y_j)^2$ where y_i is embedding of x_i. ISOMAP [31] embeds the data points in a low dimensional space while preserving the geodesic distances between data points. The geodesic distances are measured in terms of shortest paths between the points in a graph formed by computing k-nearest neighbors and introducing an edge between the neighbors. After computing the geodesic distances, spectral methods can be used to compute the embeddings that preserve this geodesic distance instead of Euclidean distance. t-distributed Stochastic Nearest Embedding (t-SNE) [16] is another method for computing manifold. It constructs a probability distribution over pairs of high-dimensional data points in such a way that similar objects have a high probability of being picked. This is followed by defining a similar distribution in the low dimension and minimized the KL divergence between the two distributions.

LLE [25] is another graph-based dimensionality reduction method that tries to preserve the local linear structure. LLE linearly approximates each data point in the training set manifold with its closest neighbors where the approximation is learned using linear regression. LLE requires computations of the k-nearest neighbors followed by computing the weight matrix W that represents each point as a linear combination of its neighbors. W is computes such that the overall reconstruction error $\sum_i ||x_i - \sum_j W_{ij} x_j||^2$ is minimized subject to constraints that W_{ij} = 0 when x_i and x_j are not neighbors, and $\sum_j W_{ij} = 1$ for all i. The low dimensional embedding is computed in LLE by minimizing the following objective: $\sum_i ||y_i - \sum_j W_{ij} y_j||^2$, where y_i denotes the low dimensional embedding of x_i, and we can normalize the representation by requiring $\sum_i y_i = 0$ and $Y^T Y = I$. W is constructed locally for each point, but the low dimensional embeddings y_i are computed globally in a single optimization step. This enables LLE to uncover global structure. Further, the embedding discovered by LLE is scale and rotation independent due to constraints on y_i. Our experiments found LLE to be most effective because of LLE's better discovery of nonlinearity, and sharper embedding in lower dimension as illustrated in Figure 2.

Our approach relies on computing the distance of the new sample point from the manifold of training data. The kernel density estimation can be used to measure the distance $d(x)$ of x from the data manifold of training set. Specifically, $d(x) = \frac{1}{|X|} \sum_{x_i \in X} k(x_i, x)$, where X is the full data set and $k(\cdot, \cdot)$ is a kernel function such as Gaussian or a simple L_∞ or L_2 norm. In case of using Gaussian kernel, the bandwidth σ needs to be carefully selected to avoid 'spiky' density estimate or an overly smooth density estimate. A typical good choice for bandwidth is a value that maximizes the log-likelihood of the training data [10]. Further, we can restrict the set of training points to be considered from the full set X to a set of immediate neighbors of x. The neighborhood can be defined using the maximum distance or bound on the number of neighbors. In our experiments, we use L_∞ norm with bound on the number of neighbors which yielded good result.

It has been hypothesized in literature [2], [6] that the deeper layers of a deep neural network provide more linear and unwrapped manifolds in comparison to the input space. Thus, the task of identifying the manifold becomes easier as we progress from the input space to the more abstract feature spaces all the way to the logit space. But the adversarial perturbations are harder to detect at higher levels and might get hidden by the lower layers of the neural network. In our experiments, we learned manifolds in input space as well as the logit space.

IV. Experiments

We evaluated our approach on MNIST dataset [15] and CIFAR10 dataset [13]. We report the key findings in this section.

As the norm bound in the PGD method for generating adversarial examples is increased, the distance of adversarial

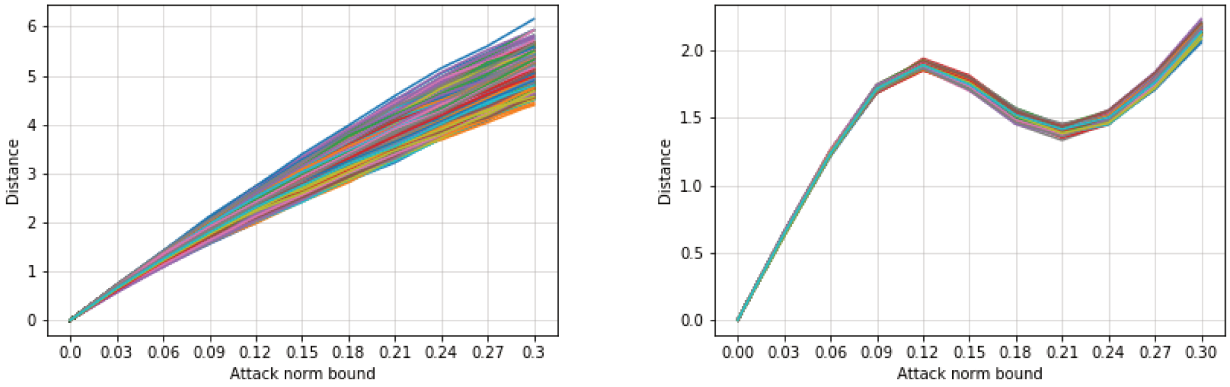

Fig. 3: Increase in adversarial distance from manifold for MNIST in input space (Left) and logit space (Right). Each line of different color shows the increase in distance with attack norm for one sample of a 1000 images. The distance monotonically increased in each of the 100 experiments in the input space. The logit space shows increase in distance with norm up to a threshold after which the distance decreases before again increasing. This is because of high norm bound allowing occasional discovery of ‘clever’ adversarial examples that are closer to the logit manifold though farther from the input manifold.

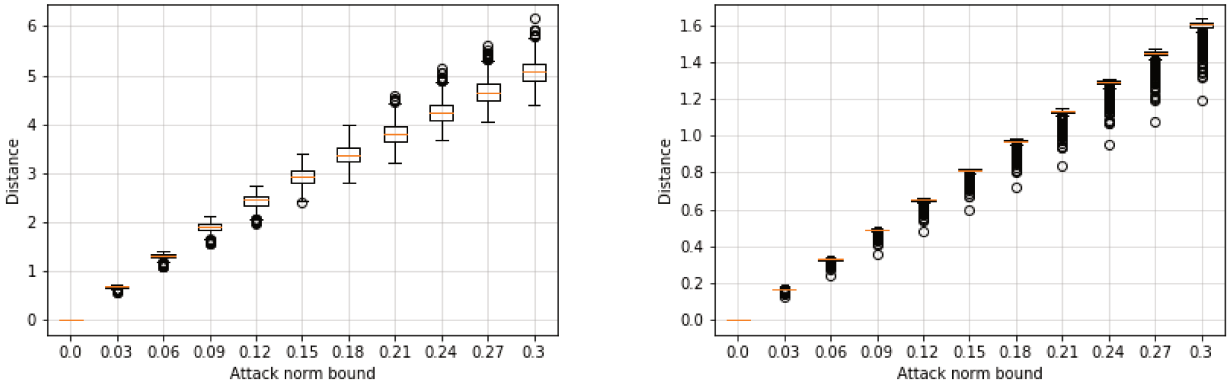

Fig. 4: Increase in adversarial example’s distance from input manifold with increase in attack norm: Left:MNIST, Right: CIFAR. The boxes in the box plot denote the first and third quartile of the distance at a given attack norm.

examples from the manifold increases. While the success of attack on the neural network increases with high norm bound, it also becomes easier to detect these adversarial examples. We observed this behavior to be common across MNIST and CIFAR10 data set as illustrated in Figure 4. The distance from manifold monotonically increases in the input space but in the logit space, higher norm bound beyond a threshold allows the attack method to find examples that decrease the distance from logit manifold even though they are farther from the input manifold. The consistent rise and fall of distance in logit space for all the 100 samples is likely a property of the used PGD method. This result is illustrated in Figure 3. The detection rate of adversarial examples for MNIST as well as CIFAR10 improves with increase in norm bound and increased distance from the manifold as illustrated in Figure 5 and 6.

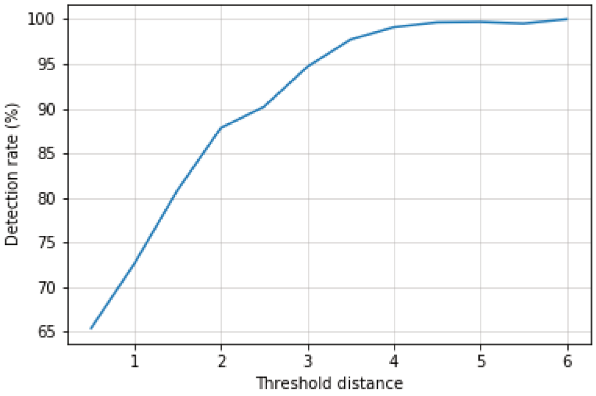

Fig. 5: Detection rate for MNIST data set

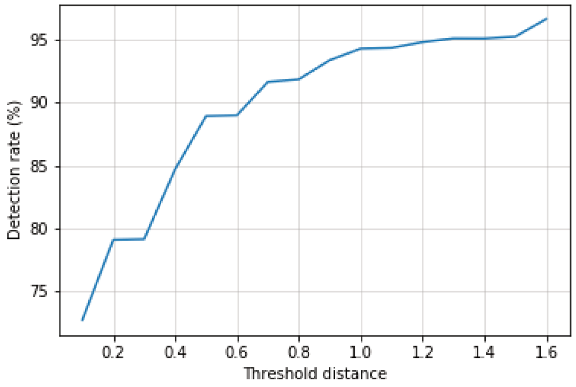

Fig. 6: Detection Rate for CIFAR

V. Conclusion

We proposed a novel approach for detecting adversarial examples using distance of a new example from the manifold of training data. We focused on PGD method for generating adversarial examples because it is shown to be a universal first order adversary. Our empirical study on MNIST and CIFAR10 datasets suggests that adversarial examples not only lie farther away from the data manifold, but this distance from manifold of an adversarial example increases with the confidence of adversarial examples. Thus, examples which are more likely to cause neural network to mis-predict are easier to detect using distance from manifold. This paper is a first step towards formulating a computational geometric approach to detecting adversarial attacks. The approach presented in this paper is not limited to only detecting adversarial examples but can be extended to learn boundaries of a machine learning model by geometrically characterizing the training data and the model parameters.

References

[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[2] Yoshua Bengio, Grégoire Mesnil, Yann Dauphin, and Salah Rifai. Better mixing via deep representations. In *International Conference on Machine Learning*, pages 552–560, 2013.

[3] Arjun Nitin Bhagoji, Daniel Cullina, Chawin Sitawarin, and Prateek Mittal. Enhancing robustness of machine learning systems via data transformations. In *Information Sciences and Systems (CISS), 2018 52nd Annual Conference on*, pages 1–5. IEEE, 2018.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *arXiv preprint arXiv:1608.04644*, 2016.

[5] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner. Detecting adversarial samples from artifacts. *arXiv preprint arXiv:1703.00410*, 2017.

[6] Jacob R Gardner, Paul Upchurch, Matt J Kusner, Yixuan Li, Kilian Q Weinberger, Kavita Bala, and John E Hopcroft. Deep manifold traversal: Changing labels with convolutional features. *arXiv preprint arXiv:1511.06421*, 2015.

[7] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples (2014). *arXiv preprint arXiv:1412.6572*.

[8] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[10] M Chris Jones, James S Marron, and Simon J Sheather. A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association*, 91(433):401–407, 1996.

[11] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018.

[12] Jernej Kos, Ian Fischer, and Dawn Song. Adversarial examples for generative models. *arXiv preprint arXiv:1702.06832*, 2017.

[13] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 2014.

[14] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[15] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[16] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[18] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 135–147. ACM, 2017.

[19] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[20] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016.

[21] Nicolas Papernot, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Fartash Faghri, Alexander Matyasko, Karen Hambardzumyan, Yi-Lin Juang, Alexey Kurakin, Ryan Sheatsley, et al. cleverhans v2. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 2016.

[22] 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*, pages 506–519. ACM, 2017.

[23] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 372–387. IEEE, 2016.

[24] Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. Crafting adversarial input sequences for recurrent neural networks. In *Military Communications Conference, MILCOM 2016-2016 IEEE*, pages 49–54. IEEE, 2016.

[25] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[26] Lawrence K Saul and Sam T Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of machine learning research*, 4(Jun):119–155, 2003.

[27] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *arXiv preprint arXiv:1511.05432*, 2015.

[28] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[29] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[30] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016.

[31] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[32] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.