Building Adversarial Defense with Non-invertible Data Transformation

Wenbo Guo[1], Dongliang Mu[1,2], Ligeng Chen[2], and Jinxuan Gai[1]

[1] College of Information Sciences and Technology, The Pennsylvania State University
wzg13@ist.psu.edu, dzm77@ist.psu.edu, jug273@ist.psu.edu
[2] National Key Laboratory for Novel Software Technology, Nanjing University
dz1733001@smail.nju.edu.cn

Abstract. Deep neural networks (DNN) have been recently shown to be susceptible to a particular type of attack possible through the generation of particular synthetic examples referred to as adversarial samples. These samples are constructed by manipulating real examples from the training data distribution in order to "*fool*" the original neural model, resulting in misclassification of previously correctly classified samples. Addressing this weakness is of utmost importance if DNN is to be applied to critical applications, such as those in cybersecurity. In this paper, we present an analysis of this fundamental flaw lurking in all neural architectures to uncover limitations of previously proposed defense mechanisms. More importantly, we present a unifying framework for protecting deep neural models using a non-invertible data transformation–developing two adversary-resistant DNNs utilizing both linear and nonlinear dimensionality reduction techniques. Empirical results indicate that our framework provides better robustness compared to state-of-art solutions while having negligible degradation in generalization accuracy.

Keywords: deep neural network · adversarial sample defense · non-invertible data transformation

1 Introduction

DNN has been applied to various critical fields such as medical imaging [2], self-driving cars [11] and malware detection [5, 18, 23]. However, recent work[15, 20] uncovered DNNs are vulnerable to *adversarial sample* – a synthetic sample generated by modifying a real example with imperceptible perturbations but causing a target DNN model to believe it belongs to the wrong class with high confidence.

To mitigate the aforementioned kind of attack, previous defenses [3, 12, 14] generally follow the basic idea of *adversarial training* in which a DNN is trained with both samples from the original data distribution as well as artificially synthesized adversarial ones. A recent unification of previous approaches [16] showed that they were all special cases of a general, regularized objective function *DataGrad*. However, because the adversarial samples space is unbounded, this framework is still vulnerable to a certain type of adversarial sample. To be specific, as we will show later in Section 5, these defense can be easily bypassed if an attacker generate adversarial samples from the network trained with *DataGrad* (*i.e.*, post-defense model).

In this paper, we present a new defense framework that increases the difficulty for attackers to craft adversarial samples from both the original DNN and the post-defense model. At a high level, we integrate an data transform layer in front of a DNN model, which transform an input sample into an latent representation before being inputted into the DNN. Technically speaking, this data transformation layer employs a non-invertible dimensionality reduction approach which increases the computational cost of mapping an adversarial sample generated from the latent space back to the input space. Evaluation results on MNIST data set demonstrate an non-invertible data transformation layer improves the robustness of a DNN and preserves the classification performance on clean testing samples. In summary, we make the following contributions:

– We propose a comprehensive framework that makes a DNN model resistant to adversarial samples by integrating an input transformation into the model.
– We develop two new defense mechanisms by injecting different dimensional reduction methods into the proposed framework.
– We theoretically and empirically evaluate the DNN models, showing that our new defense framework is resistant to adversarial samples.

2 Existing Defences

The existing defense mainly falls in to the following categories: 1) augmenting the training set and 2) enhancing model complexity. As is mentioned in Section 1, most of the defenses augment the training set with a group of adversarial samples [1, 4, 16, 22] and retrain the model with the augmented data (*i.e.*, adversarial training). These defenses can be viewed as adding a regularization term to a DNNs loss function [16], which penalizes the subspace where adversarial samples lies in. Another line of works building defences by increasing the complexity of a DNN and improve the tolerance of complex DNN models with respect to adversarial samples generated from simple DNN models. For example,[17] develops a defensive distillation mechanism, which trains a DNN from data samples that are distilled by another DNN. By using the knowledge transferred from the other DNN, the learned DNN classifiers become less sensitive to adversarial samples. Similar to [17], [9] proposed stacking an auto-encoder together with a normal DNN.

Though the above approaches, both from data augmentation and model complexity perspectives, have proven effective in handling samples generated from normal adversarial DNN models, they do not handle all adversarial samples. In light of this, we propose a framework that blocks the gradient flow from the output to input variables, a solution that prove effective even when the architecture and parameters of a given a DNN are publicly disclosed.

3 Data Transformation Enhanced DNN Framework

In this section, we introduce our frameworks design goals and choose a particular type of data transformation that will fulfill these goals.

3.1 Design Goals

As is mentioned in Section 1, we build a novel DNN framework by integrating a data transformation layer before an ordinary DNN. And we want our framework to achieving the following goals:

– It has minimal impact on the performance of a DNN model when legitimate samples are seen.
– It increases the computational cost of finding a group of adversarial samples that can bypass the post-defense model.
– it is independent from the subsequent DNN model.

3.2 Framework Overview

As is mentioned before, to generate an adversarial samples from our framework, an attack need to map an adversarial sample generated from the latent space back to the input space. This indicates that if a selected data transformation is Non-invertible, an attacker is not able to generate adversarial samples. To be specific, non-invertible data transformation stands for the following properties: (1) inverting the data transformation is computationally too complex to be tractable; or (2) inverting the data transformation will cause significant reconstruction error. Besides non-invertible, the data transformation layer should preserve the semantic meanings for an original input, which will ensure the classification of our framework. Last but not least, the transformation should also be computationally efficient and more importantly, incremental. The latter requirement is essential given that any data transformation method must be capable of handling unseen samples as they are presented. Otherwise, the data transformation will need to be retrained, and subsequently, the DNN on top of the newly retrained transform layer.

Dimensionality reduction is one particular data transformation mechanism that satisfies these design objectives. First, dimensionality reduction methods are often designed to preserve at least the most important aspects of the original data. Second, dimensionality reduction can serve as a filter for adversarial perturbations when a DNN is confronted with adversarial samples generated from the post-defence models. Third, dimensionality reduction helps reduce the dimensionality of the input distribution that is fed into the DNN. Finally, it is easier to develop non-invertible data transformation methods, since recovering higher dimensional data from lower dimensional data is difficult. The following sections introduce details of two developed defence mechanisms using different non-invertible dimensional reduction methods.

4 Data transformation enhanced DNNs

4.1 Designed Linear Mapping (DLM) DNN

We first propose a novel linear dimensional reduction method, which stems from principal component analysis (PCA). And we provide a theorem that places the lower bound on the reconstruction error.

PCA is computationally efficient and easy to implement [13]. Additionally, it preserves critical information by finding a low-dimensional subspace with maximal variance. In another word, it is convenient for an attacker to generate adversarial examples by mapping the low dimensional data back to the high one.

PCA preserves meaningful features of the original data when mapping them to a lower dimension. Given a data matrix $X \in \mathbb{R}^{n \times p}$, the transformation matrix W can be obtained by solving the optimization function as:

$$\arg\min_{Y,W} \frac{1}{2} \left|\left| X - YW^T \right|\right|_F \tag{1}$$

where $W \in \mathbb{R}^{p \times q}$, $W^T W = I_q$ and $Y \in \mathbb{R}^{n \times q}$. According to the Eckart-Young Theorem [7], the optimal solution is obtained when W consists of the q largest eigenvalues of $X^T X$. Therefore, the low dimensional mappings can be computed as follows:

$$Y = XW \tag{2}$$

Accordingly, we can approximately reconstruct the high dimensional X from the transformed data Y by:

$$\hat{X} = YW^T \tag{3}$$

which represents the process of reconstructing high dimensional approximation using only low dimensional mappings and a transform matrix. Therefore, using PCA alone for a data transformation doesn't satisfy the non-invertible criteria we introduced in Section 3.

To deal with this problem and yet preserve computational efficiency, we equip PCA with our first non-invertible characteristic. To do this, we propose a novel dimension reduction method we call a designed linear mapping (DLM). This design ensures that the PCA operation continues to preserve the critical information while the column-wise highly correlated transformation matrix guarantees that inverting the DLM will generate significant reconstruction error. To explain the consequence of this, we now introduce DLM in detail and examine its properties.

Much as in (2), we shall formally define DLM as:

$$Y = XC^T + \omega, \tag{4}$$

where $X \in \mathbb{R}^{n \times p}$, $Y \in \mathbb{R}^{n \times p_c}$. $\omega \in \mathbb{R}^{n \times p_c}$ denotes a normally distributed noise matrix, where each entry of ω generated from a normal distribution $N(0, \sigma^2)$. $C \in \mathbb{R}^{p_c \times p}$ is the transformation matrix obtained by following equation:

$$C = [B; A], \tag{5}$$

where C is constructed by combining a loading matrix $B \in \mathbb{R}^{p_b \times p}$ obtained via PCA with a designed matrix $A \in \mathbb{R}^{(p_c - p_b) \times p}$, of which all columns are highly correlated. This combination integrates PCA's information-preserving effects into our DLM. As such, the lower dimensional projection Y can provide a better representation of the original X.

Since the DLM described by (4) has a simple linear form, we estimate reconstruction $\hat{X}$ for X using high-dimensional linear regression [19] (we omit calculation details due to space constraints). According to Theorem 1 in [19], we can obtain a lower bound of the reconstruction error, which is the L_2 norm of the difference between X and $\hat{X}$ as shown in (6):

$$\left(L_2(X, \hat{X})\right)^2 \geq \kappa_0 \ \sigma^2 \frac{s \ log(p/s)}{p_c}, \tag{6}$$

where s denotes the sparsity of X. κ_0 is a constant whose value depends closely on the data set. Therefore, given a certain set of data, any linear transformation method is restricted by a constant lower bound calculated according to (6). In addition, according to Theorem 2 in [19], there also exists an upper bound of the reconstruction error as follows:

$$\left(L_2(X, \hat{X})\right)^2 \leq f(C) \frac{s \ log(p)}{p_c}, \tag{7}$$

where $f(C)$ is a function of C. According to [19], the upper bound of the reconstruction error depends on both the data transformation matrix C and noise ω. When C is a an independent correlation matrix, as in PCA, then the upper bound will approach the aforementioned lower bound. However, since we specifically design C to be highly correlated, the upper bound will be significantly larger than the lower bound [6, 8], and thus result in a larger range for the reconstruction error.

4.2 Dimensionality Reduction by Learning an Invariant Mapping (DrLIM) DNN

When adversarial samples are processed by normal DNN models, the decisions made in a lower dimensional space are completely different from those made for legitimate samples, even though adversarial samples are highly similar to legitimate ones. Therefore, we intend to employ a dimensionality reduction method that preserves the similarity of high dimensional samples in their lower dimensional mappings. Furthermore, our method needs to be capable of extracting critical information contained in the original data. Since the training of a DNN is already computationally intensive, our approach needs to be incremental in order to avoid the need for retraining the DNN.

Because of these considerations, we employ the dimensionality reduction method *DrLIM* proposed in [10]. DrLIM is specifically designed for preserving similarity between pairs of high dimensional samples when they are mapped to a lower dimensional space. As a result, there is a significantly lower chance that an adversarial sample acts as an outlier in the lower dimensional space, since its mapped location is bounded by the mapped locations of similar, legitimate samples. DrLIM can also be used in an online setting.

More importantly, we theoretically prove that inverting DrLIM is an NP-hard problem. Therefore, DrLIM is suitable for our framework in that it satisfies the second characteristic of non-invertiblity defined in Section 3. But first, we briefly review DrLIM.

DrLIM consists of a convolutional neural network (CNN) model designed for optimizing the cost function:

$$\sum_{i=1}^{P} L\left(W, (Y, X_{i_1}, X_{i_2})^i\right), \tag{8}$$

where W denotes the coefficients. X_{i_1} and X_{i_2} denote the ith pair of input sample vectors with $i = 1 \dots P$. Y is a binary label assigned to each pair of samples, with $Y = 0$ denoting a similar pair of X_{i_1} and X_{i_2}, and $Y = 1$ for dissimilar pairs. Any prior knowledge can be applied to representing dissimilarity. Let the loss function for measuring the cost for each pair be defined as:

$$L\left(W, Y, X_1, X_2\right) = (1-Y)\frac{1}{2}\big(D(X_1, X_2)\big)^2 + \frac{Y}{2}\{max\big(0, m - D(X_1, X_2)\big)\}^2, \tag{9}$$

where $D(X_1, X_2) = \|G(X_1) - G(X_2)\|_2$ is the Euclidean distance measured between the output lower dimension mapping $G(X_1)$ and $G(X_1)$ for the sample pair X_1 and X_2. Let m be a predefined constant which indicates whether all dissimilar pairs are pushed or pulled towards to maintain a constant distance m.

Since G represents a mapping by the CNN to enable the recovery of high dimensional data from the low dimensional data $G(X)$, we need to first get $G^{-1}(X)$. For the forward pass of a conventional neural network, it is not guaranteed that the weight matrices are invertible [24], implying that information lost during pooling cannot be recovered. Thus, it is very difficult to compute $G^{-1}(X)$ and recover the original data from a low dimensional representation. Since inverting the CNN is nearly impossible, one option is to reconstruct original X according to (9) given W and Y. In the following, we demonstrate that even this approach can be mapped to a NP-hard problem.

As discussed before, the most important property of DrLIM that allows it to fit into our framework is that it is provably non-invertible. Assuming $G(X)$ takes a simple linear form of $G(X) = WX$, then we have $D(X_1, X_2)^2 = (X_1 - X_2)^T W^T W (X_1 - X_2)$. Here we denote $\delta X = (X_1 - X_2)$. Following this assumption, we can reformulate (9) as follows:

$$\begin{aligned} &\min_{\delta X, z} \sum (1-Y)\delta X^T W^T W \delta X + Y z^2, \\ &\text{s.t.} z \geq 0, z \geq m - \sqrt{\delta X^T W^T W \delta X}, \end{aligned} \tag{10}$$

where $z = max\big(0, m - D(X_1, X_2)\big)$. Here we reformulate the second constraint as $\sqrt{\delta X^T W^T W \delta X} \geq m - z$. Since $m - z \geq 0$, we have following:

$$\delta X^T W^T W \delta X \geq (m-z)^2. \tag{11}$$

Therefore, W is positive semi-definite. When both sides of (11) are multiplied by -1 and substituted into (10), we find that:

$$\begin{aligned}&\min_{\delta X, z} \sum (1-Y)\delta X^T W^T W \delta X + Y z^2,\\ &\text{s.t.} z \geq 0, -\delta X^T W^T W \delta X \leq -(m-z)^2.\end{aligned} \tag{12}$$

From earlier work [21], the formulation (12) implies a quadratic problem with a non-positive semi-definite constraint, which is an NP-hard problem.

Note that solving (12) can yield the distance δX. There are multiple pairs of X_1 and X_2 that satisfy that $\delta X = (X_1 - X_2)$. This makes the problem even harder to solve. Additionally, since the linear relaxation (12) is already NP-hard, the original problem (9) is also NP-hard given that $G(X)$ is commonly regarded as a nonlinear function approximated by a neural network.

5 Evaluation

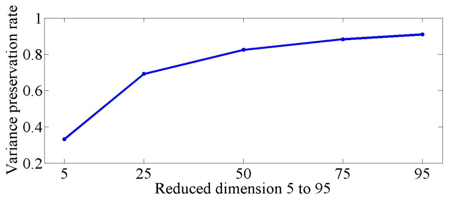

Fig. 1: Variance preservation rates with different reduced dimensions of PCA.

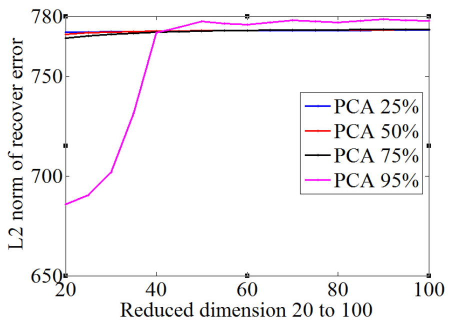

Fig. 2: Reconstruction errors with varying parameters for inverting DLM.

Going beyond theoretical analysis the non-invertible ability of the proposed defenses, we empirically demonstrate these defenses achieve the goals mentioned in Section 3. To be specific, We evaluate our framework using the widely-used MNIST data set [18]. MNIST contains a training split with 60000 greyscale images of handwritten digits and a test set containing 10000 images. Each image has a dimensionality of $28 \times 28 = 784$ pixels.

In the following experiments, we evaluate the proposed approaches under two types of adversarial samples. In order to first demonstrate that our mechanisms do indeed preserve the classification performance of the DNN, we test them with the original test set. We then test our methods with adversarial samples generated from the post-defence models to show that we achieve our secondary design goal

5.1 Limitations of Adversarial Training

In this section, We demonstrate the limitations of widely adopted defense mechanism: adversarial training. We build two different DNNs (model A and B) that share the same purpose–image recognition. Furthermore, we utilize adversarial training in learning both models A and B, which we denote as models A_{ADT} and B_{ADT}. Note that all following experiment results are the result of evaluating model A_{ADT} using different samples.

The results appear in Table 1. The second row '*Legitimate*' presents the classification error rates achieved by model A_{ADT} when testing with normal samples. In the next third and fourth row, we show classification error rates using adversarial samples generated from model A and model B respectively. The error rate obtained when testing with adversarial samples generated from model A itself is higher than the error rate found when testing with adversarial sample generated from a different model B. This is because adversarial samples generated from a specific model are more powerful for attacking that specific model. The result showed below demonstrates that adversarial samples generated from enhanced DNN models maintain their cross-model efficacy.

Different testing sets	Classification error rates of model A_{ADT}
Legitimate	0.0213
Adversarial samples from A	0.2506
Adversarial samples from B	0.1633
Adversarial samples from A_{ADT}	0.7810
Adversarial samples from B_{ADT}	0.5715

Table 1: Classification performance of testing an adversarial training enhanced model with various adversarial samples

5.2 Classification Performance

Classification Performance of DLM-DNN In this experiment, we fix the reduced dimensionality to 100. These mappings are found by DLM and PCA. In order to better explore the effect of combining DLM with PCA, we vary the percentage P_{pca} of PCA mappings used in the fixed 100 dimension. Meanwhile, the percentage of DLM mappings used varies according to $100-P_{pca}$. In addition, we also change the level of noise added to study its influence on classification performance.

We first show the classification performance when testing with legitimate samples in the column named as '*Legitimate*' in Table 2. The noise coefficient is set to be either 0.1 or 0.3, while P_{pca} varies from 5% to 95%. This performance degradation is due to the increase of noise injected into the lower dimensional mappings. Therefore, we conclude that if properly set, DLM-DNN can result in performance comparable to adversarial training.

We further examine the influence of varying P_{pca} on classification performance. As shown in Table 2, the classification performance slightly improves

Trained Model			Classification error rates with different testing sets	
			Legitimate	Adversarial
Normal DNN			0.0198	0.8981
Adversarial Training Enhanced DNN			0.0213	0.2506
DLM-DNN	Noise coefficient of 0.1	PCA(95%)	0.0226	0.3591
		PCA(75%)	0.0247	0.3211
		PCA(50%)	0.0258	0.2893
		PCA(25%)	0.0268	0.2735
		PCA(5%)	0.3101	0.5212
	Noise coefficient of 0.3	PCA(95%)	0.0386	0.2869
		PCA(75%)	0.0403	0.2685
		PCA(50%)	0.0427	0.2609
		PCA(25%)	0.0452	0.2699
		PCA(5%)	0.3710	0.5529
DrLIM-DNN			0.0384	0.1380

Table 2: Classification performance of DLM-DNN and DrLIM-DNN

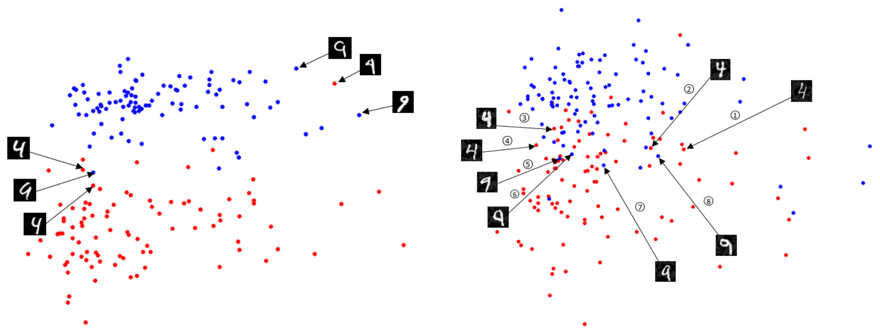

Fig. 3: 2D mapping generated by DrLIM (legitimate samples on the left and adversarial sample on the right)

with the increase of PCA dimensionality. When P_{pca} is 95%, most of critical information about original samples are preserved. However, as P_{pca} reaches 25%, enough information is preserved resulting in only a slight decrease in classification error. Meanwhile, when P_{pca} varies from 25% to 95%, the benefit of preserving any further information diminishes as with only a negligible decrease in the error rate.

We next evaluate the classification performance of DLM-DNN when confronted with adversarial samples. We list the classification error rates in the column noted as '*Adversarial*'. According to Table 2, the error rates obtained by the DLM-DNN are considerably lower than that of a standard DNN, 0.8981. Again, when P_{pca} is properly set, the DLM-DNN achieves results comparable to adversarial training. Interesting enough, as P_{pca} ranges from 25% to 95%, classification error goes up. This observation might imply that the impact of adversarial samples is mitigated to a larger degree when more random disturbances are added.

DrLIM-DNN Classification Performance In this experiment we demonstrate the classification performance of DrLIM-DNN. The training set used for evaluation includes 5 classes from the MNIST data, and each class contains 2000 samples. For testing, each of the 5 classes contains 1000 testing samples.

For training the DrLIM, we label a pair of image samples as similar when they have the same label. This simplifies the training of DrLIM utilizing strong prior knowledge. We further set the reduced dimension to 30 during the experiments. Classification performance is shown in Table 2. According to these results, using DrLIM-DNN results in a slightly higher error rate (0.0384) when testing with legitimate samples, but achieves a significant improvement in performance (0.1380) when testing adversarial samples. Especially in the latter case, DrLIM-DNN shows higher robustness when compared to adversarial training.

As previously introduced in Section 4, DrLIM is designed with the objective of preserving similarity between a pair of high dimensional samples when mapped to lower dimensional space. Fig. 3(a) shows the 2D mapping result of legitimate examples. We notice some outliers and hence highlight them and their neighbours by showing their corresponding images.

Since the point of DrLIM is to preserve the similarity in a lower dimensional space, we further visualize the 2D mapping of adversarial samples in Fig. 3(b). The 2D mapping in this case is not as clear as that for legitimate samples, but the similarity between pairs of samples are still reasonably well-preserved. This result indicates that DrLIM-DNN will not suffer as much as a normal DNN would when confronted with highly confusing adversarial samples.

In order to explore more of these outliers, in Table 3, we show the probabilities of making wrong classification decisions when testing a normal DNN and a DrLIM-DNN with these outliers. As shown in Table 3, these outliers cause a normal DNN to make wrong classification results with over 97% confidence. However, when processed with DrLIM-DNN, although these outliers are not mapped to ideal regions, the probabilities of being wrongly classified is significantly reduced to lower than 66%. This result indicates that a DrLIM-DNN is effective for responding to unfamiliar samples with lower confidence. Therefore, DrLIM-DNN will not suffer as much as a normal DNN would when confronted with highly confusing adversarial samples.

Outlier No.	Classification confidence of testing adversarial samples	
	Normal DNN	DrLIM-DNN
1	0.9995	0.5196
2	0.9721	0.5290
3	0.9989	0.6220
4	0.9921	0.5646
5	0.9998	0.5903
6	0.9997	0.5402
7	0.9998	0.6596
8	0.9919	0.5638

Table 3: Classification confidence obtained from normal DNN and DrLIM-DNN

Trained models		Classification error rates
Normal DNN		0.6596
Noise Coefficient of 0.1	PCA(95%)	0.2846
	PCA(75%)	0.2011
	PCA(50%)	0.1447
	PCA(25%)	0.1131
	PCA(5%)	0.3691
Noise Coefficient of 0.3	PCA(95%)	0.1864
	PCA(75%)	0.1729
	PCA(50%)	0.1449
	PCA(25%)	0.1884
	PCA(5%)	0.4766

Table 4: Classification performance of PCA-DNN and DLM-DNN testing with reconstructed adversarial samples by inverting PCA

As our experimental results show, DrLIM-DNN provides the best performance when tested against adversarial samples.

5.3 Reconstruction Performance

As previously introduced in Section 4, both DLM-DNN and DrLIM-DNN are non-invertible for different reasons. More importantly, we have proven that recovering the original data from a low dimensional space induced by DrLIM is an NP-hard problem. In this subsection, we mainly focus on inverting the proposed dimensional reduction method DLM by approximating it with a linear transformation matrix. We obtain the linear transformation matrix by solving a linear regression problem. In case the original data is sparse, we further employ a linear regression with L_1 regularization. First, we demonstrate that when configuring DLM as pure PCA, the approach is not robust given that it may be effectively inverted and thus allow for reconstruction of adversarial samples. Next, we examine the reconstruction error obtained from inverting DLM, taking a percentage of PCA mappings less than 100%.

We evaluate the reconstruction performance when inverting one extreme case of DLM, where DLM uses only PCA mappings. We refer to this method as PCA-DNN for comparison. To examine this extreme case, we first configure DLM as pure PCA and map legitimate testing samples to a 100-dimensional space. Then we reconstruct these legitimate samples by inverting PCA, as explained in Section 4.

Now we assume that an adversary has acquired the lower dimensional mappings generated by PCA. Then this adversarial can easily generate their corresponding *lower dimensional adversarial mappings*. So the adversarial example can be easily reconstructed as mentioned in Section 4. We use the reconstructed adversarial samples to test a normal DNN model and a DLM-DNN under different settings. According to the testing results shown in Table 4, the reconstructed adversarial samples maintain their attack power against a normal DNN model.

And these adversarial samples can be effectively defended by a DLM-DNN as shown in Table 2.

We finally investigate the reconstruction errors when inverting DLM-DNN. We present reconstruction errors when varying percentages of PCA mappings used and when varying sub-space dimensionality in Fig 2. Our experiment shows that inverting a DLM-DNN leads to high reconstruction errors, regardless of how many PCA mappings are used what dimensionality is used. Recall the theoretical analysis of DrLIM-DLM in Section 4, we demonstrate that our proposed methods effectively build an adversary-resistant DNN.

6 Conclusion

We proposed a new framework for constructing deep neural network models that are robust to adversarial samples, based on an analysis of both the "blind-spot" of DNNs and the limitations of previous solutions. With our proposed framework, we developed two adversary-resistant DNN architectures that leverage non-invertible data transformation mechanisms. Then we empirically showed that crafting an adversarial sample for the first architecture will incur significant distortion and thus lead to easily detectable adversarial samples. In contrast, under the second architecture, we theoretically demonstrated that it is impossible for an adversary to craft an adversarial sample to attack it. This implies that our proposed framework no longer suffers from attacks that rely on generating model-specific adversarial samples.

Furthermore, we demonstrated that recently studied adversarial training methods are not sufficient defense mechanisms. Applying our new framework to the MNIST data set, we empirically demonstrate that our new framework significantly reduces the error rates in classifying adversarial samples. Furthermore, our new framework has the same classification performance for legitimate samples with negligible degradation.

References

1. S. Aman, N. Hongseok, and D. John, *Certifiable distributional robustness with principled adversarial training*, in International Conference on Learning Representations, 2018.
2. Y. Bar, I. Diamant, L. Wolf, and H. Greenspan, *Deep learning with non-medical training used for chest pathology identification*, in SPIE Medical Imaging, International Society for Optics and Photonics, 2015, pp. 94140V–94140V.
3. A. N. Bhagoji, D. Cullina, and P. Mittal, *Dimensionality reduction as a defense against evasion attacks on machine learning classifiers*, arXiv preprint, (2017).
4. M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, *Parseval networks: Improving robustness to adversarial examples*, in Proceedings of the 34th International Conference on Machine Learning, 2017.

5. G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, *Large-scale malware classification using random projections and neural networks*, in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 3422–3426.
6. D. L. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, 52 (2006), pp. 1289–1306.
7. C. Eckart and G. Young, *The approximation of one matrix by another of lower rank*, Psychometrika, 1 (1936), pp. 211–218.
8. M. Fornasier and H. Rauhut, *Compressive sensing*, in Handbook of mathematical methods in imaging, Springer, 2011, pp. 187–228.
9. S. Gu and L. Rigazio, *Towards deep neural network architectures robust to adversarial examples*, arXiv:1412.5068 [cs], (2014).
10. R. Hadsell, S. Chopra, and Y. LeCun, *Dimensionality reduction by learning an invariant mapping*, in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.
11. R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun, *Learning long-range vision for autonomous off-road driving*, Journal of Field Robotics, 26 (2009), pp. 120–144.
12. R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, *Learning with a strong adversary*, CoRR, abs/1511.03034, (2015).
13. I. Jolliffe, *Principal component analysis*, Wiley Online Library, 2002.
14. T. Miyato, S.-i. Maeda, M. Koyama, K. Nakae, and S. Ishii, *Distributional smoothing with virtual adversarial training*, stat, 1050 (2015), p. 25.
15. A. Nguyen, J. Yosinski, and J. Clune, *Deep neural networks are easily fooled: High confidence predictions for unrecognizable images*, in 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015.
16. A. G. Ororbia II, C. L. Giles, and D. Kifer, *Unifying adversarial training algorithms with flexible deep data gradient regularization*, arXiv:1601.07213 [cs], (2016).
17. N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, *Distillation as a defense to adversarial perturbations against deep neural networks*, arXiv preprint arXiv:1511.04508, (2015).
18. R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, *Malware classification with recurrent networks*, in 2015 IEEE International Conference on Acoustics, Speech and Signal Processing, 2015.
19. G. Raskutti, M. J. Wainwright, and B. Yu, *Minimax rates of estimation for high-dimensional linear regression over-balls*, IEEE Transactions on Information Theory, 57 (2011), pp. 6976–6994.
20. C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, *Intriguing properties of neural networks*, in International Conference on Learning Representations, 2014.
21. S. A. Vavasis, *Nonlinear Optimization: Complexity Issues*, Oxford University Press, Inc., 1991.
22. X. Wu, U. Jang, J. Chen, L. Chen, and S. Jha, *Reinforcing adversarial robustness using model confidence induced by adversarial training*, in International Conference on Machine Learning, 2018.
23. Z. Yuan, Y. Lu, Z. Wang, and Y. Xue, *Droid-sec: Deep learning in android malware detection*, in ACM SIGCOMM Computer Communication Review, 2014.
24. M. D. Zeiler and R. Fergus, *Visualizing and understanding convolutional networks*, in European Conference on Computer Vision, 2014.