CHARACTERIZING ADVERSARIAL SUBSPACES USING LOCAL INTRINSIC DIMENSIONALITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep Neural Networks (DNNs) have recently been shown to be vulnerable against adversarial examples, which are carefully crafted instances that can mislead DNNs to make errors during prediction. To better understand such attacks, the properties of subspaces in the neighborhood of adversarial examples need to be characterized. In particular, effective measures are required to discriminate adversarial examples from normal examples in such subspaces. We tackle this challenge by characterizing the intrinsic dimensional property of adversarial subspaces, via the use of *Local Intrinsic Dimensionality* (LID). LID assesses the space-filling capability of the subspace surrounding a reference example, based on the distance distribution of the example to its neighbors. We first provide explanations about how adversarial perturbation affects the LID characteristic of adversarial subspaces. Then, we explain how the LID characteristic can be used to discriminate adversarial examples generated using the state-of-the-art attacks. We empirically show that the LID characteristic can outperform several state-of-the-art detection measures by large margins for five attacks across three benchmark datasets. Our analysis of the LID characteristic for adversarial subspaces not only motivates new directions of effective adversarial defense but also opens up more challenges for developing new attacks to better understand the vulnerabilities of DNNs.

1 INTRODUCTION

Deep Neural Networks (DNNs) are highly expressive models that have achieved state-of-the-art performance on a wide range of complex problems, such as speech recognition (Hinton et al., 2012) and image classification (Krizhevsky et al., 2012). However, recent studies have found that DNNs can be fooled by adversarial examples (Szegedy et al., 2013; Goodfellow et al., 2014; Nguyen et al., 2015). These are intentionally-perturbed inputs that can fool the network into making incorrect predictions at test time with high confidence, and they are transferable across different networks (Liu et al., 2016; Carlini & Wagner, 2017b; Papernot et al., 2016b). The amount of perturbation required is often small and imperceptible to human observers in the case of images. This undesirable property of deep networks has become a major security concern for applying DNNs to real-world applications such as self-driving cars and identity recognition (Evtimov et al., 2017; Sharif et al., 2016). In this paper, we aim to further understand adversarial attacks by characterizing adversarial subspaces.

An adversarial subspace is the region immediately surrounding an adversarial example. It exists not just in the input space, but also in the activation space of different layers (Szegedy et al., 2013). Developing an understanding of the properties of adversarial subspaces is a key requirement for adversarial defense. Several works have attempted to characterize the properties of adversarial subspaces, but no definitive method yet exists which can discriminate adversarial subspaces from normal data subspaces. Szegedy et al. (2013) argued that adversarial subspaces have low probability (are not naturally occurring), but are densely scattered in the high dimensional representation space of DNNs. However, a linear formulation argues that adversarial subspaces span a multidimensional contiguous space, rather than being scattered randomly in small pockets (Goodfellow et al., 2014; Warde-Farley et al., 2016). Tanay & Griffin (2016) further highlight that adversarial subspaces lie close, yet off the data submanifold. It has also been found that the boundaries of adversarial subspaces are similarly close to legitimate data points in adversarial directions, and the higher the number of orthogonal adversarial directions of these subspaces, the more transferable they are to other models (Tramèr

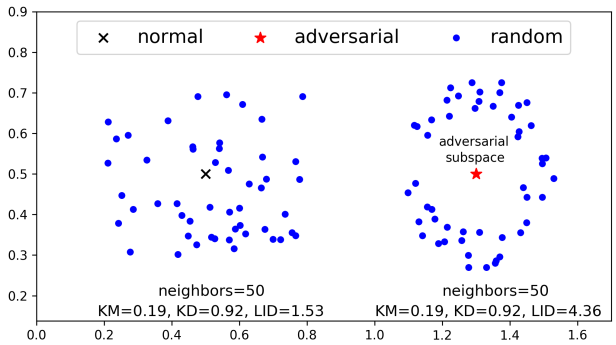

Figure 1: This example shows when density measures fail to characterize the spatial property of adversarial subspace. Gaussian kernel with bandwidth 0.2 is used for KD.

et al., 2017). Generally, we can summarize the known properties of adversarial subspaces as: (1) they are of low probability, (2) they span a contiguous multidimensional space, (3) they lie off but are close to the data submanifold, and (4) they have different class distributions from their closest data submanifold.

Some properties of adversarial examples have been analyzed and applied to build up detection methods. For instance, *Kernel Density* (KD) has been proposed as a measure to discriminate adversarial subspaces (Feinman et al., 2017). Carlini & Wagner (2017a) empirically showed that a kernel density based detector is useful, since adversarial subspaces generally have lower density than normal data subspaces, due to their low probability. In addition to kernel density, there are other density type measures, such as number of nearest neighbors and mean distance to k nearest neighbors (k-mean distance). However, these have some limitations for characterizing local adversarial subspaces. E.g., in Figure 1, the three density based measures fail to differentiate an adversarial example (red star) from a normal example (black cross), as the two examples are locally surrounded by the same number of neighbors (50) and have the same k-mean distance (KM=0.19), as well as the same kernel density (KD=0.92).

Figure 1 leads us to alternatively consider expansion-based models of intrinsic dimensionality, which assess the *local distance distribution* of a sample of interest to its neighbors, as a potentially effective method of characterizing adversarial examples. This is a family of dimensionality models that provide a local view of the dimensional structure of the data and which has been successfully used in a wide range of applications, such as manifold learning, dimension reduction, similarity search and anomaly detection (Amsaleg et al., 2015; Houle, 2017a). *Local Intrinsic Dimensionality* (LID) is one such expansion model that characterizes the space-filling capability of the local subspace of a reference sample (Houle, 2017a;b). In Figure 1, the estimated LID of the adversarial example (LID=4.36) is much higher than that of the referenced normal data sample (LID=1.53), illustrating how estimated LID can be used to successfully discriminate the adversarial subspace from the normal subspace by capturing intrinsic dimensional properties of adversarial subspaces. In this paper, we aim to study the LID properties of adversarial examples generated using the state-of-the-art attack methods, and to show that LID can differentiate them from normal examples. In particular, our contributions are:

- We propose the use of LID for characterizing adversarial subspaces of deep networks. We discuss how adversarial perturbation can affect the LID characteristics of an adversarial subspace, and we empirically show that with LID characteristics, a minibatch of training data is sufficient to differentiate them.
- Our study reveals that the estimated LID of adversarial examples is significantly higher than that of normal data examples, and that this difference becomes more pronounced in deeper layers of DNNs.
- We empirically demonstrate that the LID characteristics of adversarial examples generated using five state-of-the-art attack methods can be easily discriminated from that of normal

examples, and a classifier using estimated LID features can outperform some existing detection measures on five attacks across three benchmark datasets.

- We show that the LID based detector is robust to the optimization based attack of (Carlini & Wagner, 2017a).

2 Related Work

In this section, we briefly review two key related areas: adversarial attacks and adversarial defenses.

Adversarial Attacks: A wide range of attacks have been proposed to craft adversarial examples to fool deep networks; here, we mention a selection of such works. The fast Gradient Method (FGM) (Goodfellow et al., 2014) perturbs normal input once by adding a small magnitude of perturbation along the gradient direction. The basic Iterative Method (BIM) is an iterative version of FGM (Kurakin et al., 2016). One variant of BIM stops immediately at misclassification (BIM-a), and another iterates a fixed number of steps (BIM-b). The Jacobian-based Saliency Map Attack (JSMA) iteratively selects the two most effective pixels to perturb based on the adversarial saliency map until misclassification (Papernot et al., 2016c). The Optimization-based attack (Opt) is arguably the most effective to date and addresses the problem via an optimization framework (Liu et al., 2016; Carlini & Wagner, 2017b).

Adversarial Defenses: A number of defense techniques have been introduced, including adversarial training (Goodfellow et al., 2014), distillation (Papernot et al., 2016d), gradient masking (Gu & Rigazio, 2014), and feature squeezing (Xu et al., 2017). However, these defenses can either be evaded by Opt attacks or only provide marginal improvements (Carlini & Wagner, 2017a; He et al., 2017). Given the inherent challenges for adversarial defense, recent works have instead focused on detecting adversarial examples. These works attempt to discriminate adversarial examples (positive class) from both normal and noisy examples (negative class), based on features extracted from different layers of a DNN. Detection subnetworks based on activations (Metzen et al., 2017), cascade detector based on the PCA projection of activations (Li & Li, 2016), augmented neural network detector based on statistical measures, logistic regression detector based on KD, and Bayesian Uncertainty (BU) features (Grosse et al., 2017) are a few such works. However, a recent study by Carlini & Wagner (2017a) has shown that these detection methods can be vulnerable to attack as well.

3 Local Intrinsic Dimensionality

In dimensionality theory, expansion models (such as the expansion dimension and local intrinsic dimensionality (Houle, 2017a;b)) measure the rate of growth in the number of data objects encountered as the distance from the reference sample increases. Expansion models work with the distance distributions of a sample to its neighbors. Intuitively, in Euclidean space, the volume of an m-dimensional ball grows proportionally to r^m, when its size is scaled by a factor of r. From this rate of volume growth with distance, the expansion dimension m can be deduced as:

$$\frac{V_2}{V_1} = \left(\frac{r_2}{r_1}\right)^m \Rightarrow m = \frac{\ln(V_2/V_1)}{\ln(r_2/r_1)} \tag{1}$$

Expansion models of dimensionality provide a *local view* of the dimensional structure of the data, as their estimation is restricted to a neighborhood around the sample of interest. Transferring the concept of expansion dimension in the Euclidean space to a statistical setting leads to the formal definition of the LID (Houle, 2017a).

Definition 1 (Local Intrinsic Dimensionality)**.**
Given a data sample $x \in X$, let $R > 0$ be a random variable denoting the distance from x to other data samples. If the cumulative distribution function $F(r)$ of R is positive and continuously differentiable at distance $r > 0$, the LID of x at distance r is given by:

$$\mathrm{LID}_F(r) \triangleq \lim_{\epsilon \to 0} \frac{\ln\big(F((1+\epsilon)\cdot r)/F(r)\big)}{\ln(1+\epsilon)} = \frac{r \cdot F'(r)}{F(r)} \tag{2}$$

whenever the limit exists.

$F(r)$ is analogous to the volume V in Equation (1). The last equality of Equation (2) follows by applying L'Hôpital's rule to the limits (Houle, 2017a) and $\text{LID}_F(r)$ can be estimated using the distances of x to its k nearest neighbors (Amsaleg et al., 2015). The local intrinsic dimension at x is in turn defined as the limit, when the radius r tends to zero:

$$\text{LID}_F = \lim_{r \to 0} \text{LID}_F(r) \tag{3}$$

LID_F describes the relative rate at which its cumulative distance function $F(r)$ increases as the distance r increases from 0. From the dimensionality point of view, LID_F measures the dimension of the submanifold embedded in X at the locality of x. If LID_F is high, the proportional expansion in the number of data samples in the neighborhood of x is expected to be large. That is, the local submanifold that x lies on has high dimensionality. Conversely, if LID_F is low, then the dimension of the submanifold embedded at x is low. It should be noted that the expansion dimension defined by LID is different to the physical dimension that is defined as the minimum number of coordinates needed to specify any sample within it. We refer readers to Houle (2017a;b) for more details concerning the LID model.

Estimation of LID: In extreme value theory, the smallest k nearest neighbor distances could be regarded as extreme events associated with the lower tail of the underlying distance distribution. Under very reasonable assumptions, the tails of continuous probability distributions converge to the Generalized Pareto Distribution (GPD), a form of power-law distribution (Coles et al., 2001). From this, Amsaleg et al. (2015) developed several estimators of LID to heuristically approximate the true underlying distance distribution by a transformed GPD; among these, the Maximum Likelihood Estimator (MLE) exhibited a useful trade-off between statistical efficiency and complexity. Given a reference sample $x \sim \mathcal{P}$, where $\mathcal{P}$ represents the data distribution, the MLE estimator of LID makes use of its distances to the first k nearest neighbors as:

$$\widehat{\text{LID}}(x) = -\left(\frac{1}{k} \sum_{i=1}^{k} \log \frac{r_i(x)}{r_k(x)} \right) \tag{4}$$

where $r_i(x)$ is the Euclidean distance between x and its i-th nearest neighbor; that is, $r_1(x)$ is the minimum distance, while $r_k(x)$ is the maximum distance. Here, the k nearest neighbors of x are searched among all available training data samples randomly drawn from $\mathcal{P}$. The LID defined in Equation (3) is the *theoretical* LID and $\widehat{\text{LID}}$ defined in Equation (4) is its *estimate*. In the remainder of this paper, we will refer to Equation (4) to calculate the LID estimates.

4 Characterizing Adversarial Subspaces

Our aim is to understand adversarial subspaces and thereby derive potential defenses and provide new directions for more efficient attacks. We begin by providing some motivation as to how adversarial perturbation might affect the LID characteristic of adversarial subspaces. We then show how a detector can be designed using LID estimates to discriminate between adversarial and normal examples.

LID of Adversarial Subspaces: Consider a sample $x \in X$ lying within a data submanifold S, where X is a randomly sampled dataset from $\mathcal{P}$ consisting only of normal (unperturbed) examples. Adversarial perturbation of x typically results in a new sample x' whose coordinates differ from those of x by very small amounts. Assuming that x' is indeed a successful adversarial perturbation of x, the theoretical LID value associated with x is simply the dimension of S, whereas the theoretical LID value associated with x' is the dimension of the adversarial subspace within which it resides.

Since perturbation schemes generally allow the modification of all data coordinates, they exploit the full degrees of freedom afforded by the representational dimension of the data domain. As pointed out by (Goodfellow et al., 2014; Warde-Farley et al., 2016; Tanay & Griffin, 2016), x' is very likely to lie outside S (but very close to S — in a high-dimensional contiguous space). In applications involving high-dimensional data, the representational dimension is typically far larger

than the intrinsic dimension of any given data submanifold, which implies that the theoretical LID of x' is far greater than that of x.

In practice, however, the values of LID must be estimated from local data samples. This is typically done by applying an appropriate estimator (such as the MLE estimator shown in Equation (4)) to a k-nearest neighborhood of the test samples, for some appropriate fixed choice of k. Typically, k is chosen large enough for the estimation to stabilize, but not so large that the sample is no longer local to the test sample. If the dimension of S is reasonably low, one can expect the estimation of the LID of x to be reasonably accurate.

For the adversarial subspace, the samples appearing in the neighborhood of x' can be expected to be drawn from more than one manifold. The proximity of x' to S means that the neighborhood is likely to contain neighbors lying in S; however, if the neighborhood were composed mostly of samples drawn from S, x' would not likely be an adversarial example. Thus, the neighbors of x' taken together are likely to span a subspace of intrinsic dimensionality much higher than any of these submanifolds considered individually, and the LID estimate computed for x' can be expected to reveal this.

Efficiency through Minibatch Sampling: Computing neighborhoods with respect to the entirety of the dataset X can be prohibitively expensive, particularly when the (global) intrinsic dimensionality of X is too high to support efficient indexing. For this reason, when X is large, the computational cost can be reduced by estimating the LID of an adversarial example x' from its k-nearest neighbor set within a randomly-selected sample (*minibatch*) of the dataset X. Since the LID model regards the distances from x' to the members of X as independently-drawn samples from a distribution, any random minibatch induces a smaller sample of distances drawn independently from the same distribution.

Provided that the minibatch is chosen sufficiently large so as to ensure that the k-nearest neighbor sets remain in the vicinity of x', estimates of LID computed for x' within the minibatch would resemble those computed within the full dataset X. Conversely, as the size of the minibatch is reduced, the variance of the estimates would increase. However, if the gap between the true LID values of x and x' is sufficiently large, even an extremely small minibatch size and / or small neighborhood size could conceivably produce estimates whose difference is sufficient to reveal the adversarial nature of x'. As we shall show in Section 5.1, detection turns out to be possible even for minibatch sizes as small as 100, and for neighborhood sizes as small as 20.

Using LID to Characterize Adversarial Examples: We next describe how LID estimates characterize adversarial examples, and we explore such estimates by applying LID to train a detector to distinguish adversarial examples. This methodology requires that training sets be comprised of three types of examples: adversarial, normal and noisy. This replicates the methodology used in (Feinman et al., 2017; Carlini & Wagner, 2017a), where the rationale for including noisy examples is that DNNs are required to be robust to random input noise (Fawzi et al., 2016) and noisy inputs should not be identified as adversarial attacks. A classifier can be trained by using the training data to construct features for each sample, based on its LID within a normal example minibatch across different layers, where the class label is assigned positive for adversarial examples and assigned negative for normal and noisy examples.

Algorithm 1 describes how the LID features can be extracted for training an LID based detector. Given an initial training set of only normal examples and a pre-trained DNN, the algorithm outputs a detector trained using LID features. The extraction of LID features first begins with generating adversarial and noisy counterparts to normal examples (step 3 and 4) in each minibatch. One minibatch of normal examples (B_{norm}) is used for generating 2 counterpart minibatches of examples: one adversarial (B_{adv}) and one noisy (B_{noisy}). The adversarial examples are crafted by applying an adversarial attack on normal examples (step 3), while noisy examples are crafted by adding the same amount of *random* noise to normal examples as that of adversarial examples (step 4). The LID of each example (either normal, adversarial or noisy) is estimated based on Equation (4) by its k nearest neighbors in the *normal* minibatch (steps 12-14). An implication here is that for any new unknown test example, only a minibatch of normal training examples is used to estimate its LID. For each example and each transformation layer in the DNN, an LID estimate is calculated. The distance function needed for this estimate uses the activation values of the neurons in the given layer as inputs (step 7). We use all transformation layers, including conv2d, max-pooling, dropout, ReLU

Algorithm 1 Train a LID based detector

Input:
X: a dataset of normal examples
$H(x)$: a pre-trained DNN with L transformation layers
k: the number of nearest neighbors for LID estimation
Output:
Detector(LID) ▷ a detector
1: LID$_{neg}$=[], LID$_{pos}$=[]
2: **for** B_{norm} in X **do** ▷ B_{norm}: a minibatch of normal examples
3: B_{adv} := adversarial attack B_{norm} ▷ B_{adv}: a minibatch of adversarial examples
4: B_{noisy} := add random noise to B_{norm} ▷ B_{noisy}: a minibatch of noisy examples
5: $N = |B_{norm}|$ ▷ number of examples in B_{norm}
6: LID$_{norm}$, LID$_{noisy}$, LID$_{noisy}$ = zeros$[N, L]$
7: **for** i in $[1, L]$ **do**
8: $A_{norm} = H^i(B_{norm})$ ▷ i-th layer activations of B_{norm}
9: $A_{adv} = H^i(B_{adv})$ ▷ i-th layer activations of B_{adv}
10: $A_{noisy} = H^i(B_{noisy})$ ▷ i-th layer activations of B_{noisy}
11: **for** j in $[1, N]$ **do**
12: LID$_{norm}[j, i]$ = $-\left(\frac{1}{k}\sum_{i=1}^{k}\log\frac{r_i(A_{norm}[j],A_{norm})}{r_k(A_{norm}[j],A_{norm})}\right)$
13: LID$_{adv}[j, i]$ = $-\left(\frac{1}{k}\sum_{i=1}^{k}\log\frac{r_i(A_{adv}[j],A_{norm})}{r_k(A_{adv}[j],A_{norm})}\right)$
14: LID$_{noisy}[j, i]$ = $-\left(\frac{1}{k}\sum_{i=1}^{k}\log\frac{r_i(A_{noisy}[j],A_{norm})}{r_k(A_{noisy}[j],A_{norm})}\right)$
15: ▷ $r_i(A[j], A_{norm})$: the L_2 distance of $A_[j]$ to its i-th nearest neighbor in A_{norm}
16: **end for**
17: **end for**
18: LID$_{neg}$.append(LID$_{norm}$), LID$_{neg}$.append(LID$_{noisy}$)
19: LID$_{pos}$.append(LID$_{adv}$)
20: **end for**
21: Detector(LID) = train a classifier on (LID$_{neg}$, LID$_{pos}$)

and softmax, since we expect adversarial subspaces to exist in each layer of the DNN representation space, as will be discussed in Section 5.1. The LID values for the example are then used as feature values (one feature for each transformation layer). Finally, a classifier (such as logistic regression) is trained using the LID features. At test time, the LID features of an unknown test example can be computed based on its k nearest neighbors found in a minibatch of normal examples randomly selected from the training set. The test example can then be classified by the LID based detector to either the positive (adversarial) or negative (non-adversarial) class based on its LID feature values.

5 EVALUATING LID CHARACTERISTICS

In this section, we explore the subspace characterization power of LID by applying LID to differentiate 5 current attacks. In our experiments, The attacks were selected based on their effectiveness and diversity: FGM, BIM-a, BIM-b, JSMA, Opt as introduced in Section 2. LID is compared with state-of-the-art detection measures KD and BU as discussed in Section 2, on three benchmark datasets: MNIST, CIFAR-10 and SVHN. We first introduce the experimental setup and then report and discuss the results.

Experimental Setup: For each dataset, a classification DNN was independently pretrained on the initial training images (*pre-train*) and the initial test images were set aside for test data (*pre-test*). Any *pre-test* images not correctly classified were discarded and the remainder of the *pre-test* images were sub-divided into *train* (80%) and *test* (20%). Each of these sets was randomly partitioned into minibatches of size 100. The LID, KD and BU based detectors were trained separately on the *train* set using the scheme in Algorithm 1 with the calculation of LID replaced by KD and BU in the cases of KD and BU detectors. All detectors were then evaluated against an equal number of normal, noisy and adversarial images crafted using steps 2-4 in Algorithm 1, based on images from the test set (*test*). The LID, KD and BU characteristics of those test images were extracted using steps 1-18

in Algorithm 1. It should be noted that the test images (*test*) were never seen during any training process, avoiding cross contamination. The adversarial examples for both training and test were generated by applying one of the five selected attacks. The noisy examples for the JSMA attack were crafted by randomly clipping an equal number of perturbed pixels, while noisy images for other attacks were crafted by adding the same amount of L_2 Gaussian noise, following the setting in Feinman et al. (2017). We used the logistic regression classifier as detector, and report its AUC score as the metric for performance, as suggested by Feinman et al. (2017); Carlini & Wagner (2017a).

Deep Neural Networks: The pretrained DNN used for MNIST was a 5-layer ConvNet with max-pooling and dropout. It achieved 99.29% classification accuracy on (normal) *pre-test* images. For CIFAR-10, a 12-layer ConvNet with max-pooling and dropout was used. This model reported an accuracy of 84.56% on (normal) *pre-test* images. For SVHN, we trained a 6-layer ConvNet with max-pooling and dropout. It achieved 92.18% accuracy on (normal) *pre-test* images. We deliberately did not tune the DNNs, as their performance was close to the state-of-the-art and could thus be considered sufficient for use in an adversarial study (Feinman et al., 2017).

Parameter Tuning: We tuned the bandwidth (σ) parameter for KD and the number of nearest neighbors (k) for LID based on nested cross validation within the training set (*train*) . We did not tune the number of prediction times (T) (50 in our setting) for BU as it is not sensitive provided $T > 20$ (Carlini & Wagner, 2017a). The bandwidth was tuned by a grid search over $[0, 10)$ in log-space and the k nearest neighbors were tuned by a grid search over $[10, 100)$ with respect to a minibatch size of 100, based on their detection AUC. For a given dataset, the parameter setting selected was the one with highest AUC averaged across all attacks. The optimal bandwidths chosen for MNIST, CIFAR-10 and SVHN were 3.79, 0.26, 1.0 respectively, while the value of k for LID estimation was set to 20 for MNIST and CIFAR-10, and 30 for SVHN.

Our implementation is based on the detection framework by Feinman et al. (2017). For FGM, JSMA, BIM-a, and BIM-b attacks, we used the *cleverhans* library (Papernot et al., 2016a) and used the author's implementation for the Opt attack (Carlini & Wagner, 2017b). We scaled all images to $[0, 1]$. Our code is available at: `https://www.dropbox.com/s/cc2vpqya0il78j6/LID_adv_detection.zip`.

5.1 Understanding LID Characteristics of Current Adversarial Examples

We provide empirical results showing the LID characteristics of adversarial examples generated by the most effective attack to date – the Opt attack. The left subfigure in Figure 2 shows the LIDs (for the pre-softmax layer) of 100 randomly selected normal, noisy and adversarial (Opt attack) CIFAR-10 examples. We observe that the estimated LIDs of adversarial examples are significantly higher than those of normal or noisy examples (these lines overlap each other). This supports our expectation that adversarial subspaces have higher intrinsic dimensionality than normal data subspaces as discussed in Section 4. It also suggests that the transit from normal example to adversarial example may follow directions where the complexity of the local data submanifold significantly increases (increasing estimated LID).

We further show in the right subfigure of Figure 2 that the estimated LIDs of adversarial examples are more easily discriminated from other examples at deeper layers of the network. The 12-layer ConvNet we used for CIFAR-10 consists of 26 transformation layers: the input layer (L_0), conv2d/max-pooling (L_{1-17}), dense/dropout (L_{18-24}) and the softmax layer (L_{25}). The estimated LID characteristics of adversarial examples become distinguishable (detection AUC> 0.5) from the dense layers (L_{18-24}) and significantly different after the softmax layer (L_{25}). This suggests that the fully-connected and softmax transformations are more vulnerable to adversarial perturbations than convolutional transformations. Some further estimated LID characteristics for the MNIST and SVHN datasets are provided in the Appendix A.2.

In regard to the stability of performance based on parameter variation (k for LID, or bandwidth for KD), we can see in Figure 3 that LID is more stable than KD, exhibiting less variation in AUC as the parameter varies. From this figure, we also see that KD requires significantly different optimal settings for different types of data. For simpler datasets such as MNIST and SVHN, it requires quite high bandwidths for the best performance.

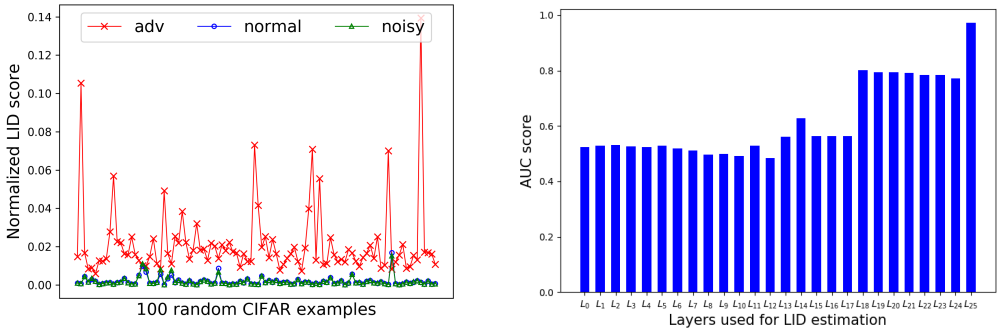

Figure 2: The left-hand figure shows the normalized LID scores (at pre-softmax layer) of 100 normal (blue), noisy (green) and Opt attack (red x-cros) CIFAR-10 examples. The noisy and adversarial examples were generated from the normal example. The blue and green lines are superimposed. The right-hand figure shows the detection performance (AUC) based on LID scores of different layers. L_i denotes i-th transformation layers.

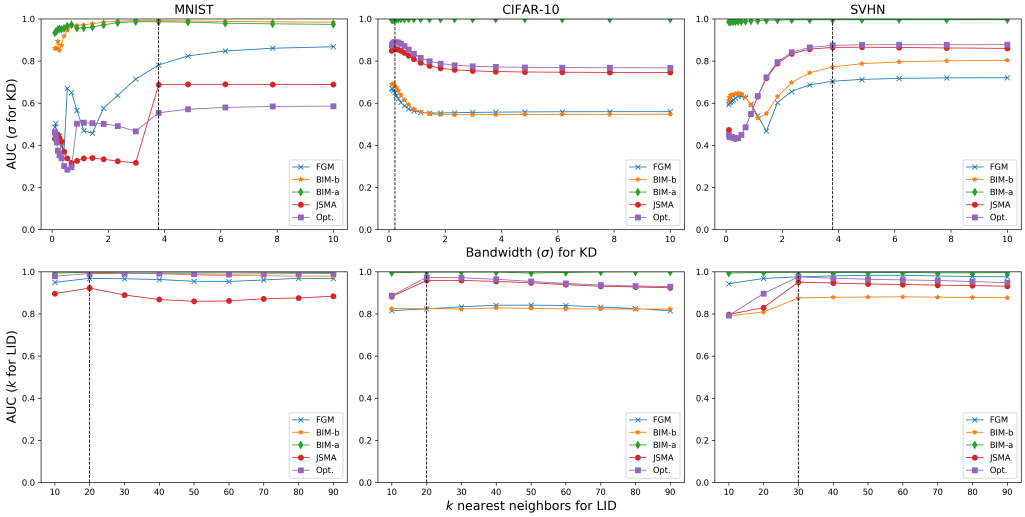

Figure 3: Top row: tuning bandwidth σ for KD by a grid search over $[0, 10)$ in log-space, separately for each dataset. Bottom row: tuning k for LID by a grid search over $[10, 100)$ for minibatch size 100, separately for each dataset. The vertical dashed lines denote the selected parameter choice.

5.2 Exploring LID on Adversarial Examples

LID Outperforms KD and BU: We compare the detection performance of LID with detectors trained with features of KD and BU, respectively, as well as the one trained with the combination of them (KD+BU). As shown in Table 1, LID outperforms the KD and BU measures (both individually and combined) by large margins on all attacks across all datasets. For the most effective attack to date, the Opt attack, the LID based detector achieved AUCs of 99.24%, 98.93% and 97.60% on MNIST, CIFAR-10 and SVHN, respectively, compared to AUCs of 95.35%, 93.77% and 90.66% for the combination of KD and BU. This strong performance suggests that LID is a highly promising characteristic for discriminating the properties of adversarial subspaces. We also that KD was not effective for some attacks, such as the FGM, JSMA and BIM-a attacks, whereas the BU measure failed to detect the FGM and BIM-b attacks on the MNIST dataset.

Effect of Larger Minibatch Size for LID: In regard to the minibatch size, a default value of 100 was used, with a view to ensuring efficiency. This can deliver state of the art performance (the results in Table 1). However, it is an interesting question as to whether the use of a larger minibatch

Table 1: A comparison of the discrimination power (AUC score of a logistic regression classifier) between LID, KD and BU. The AUC score is computed for each attack on each dataset, and the best results are highlighted in **bold**.

Dataset	Feature	FGM	BIM-a	BIM-b	JSMA	Opt
MNIST	KD	78.12%	99.14%	98.61%	68.77%	95.15%
	BU	32.37%	91.55%	25.46%	88.74%	71.29%
	KD+BU	82.43%	99.20%	98.81%	90.12%	95.35%
	LID	**96.89%**	**99.60%**	**99.83%**	**92.24%**	**99.24%**
CIFAR-10	KD	64.92%	68.38%	98.70%	85.77%	91.35%
	BU	70.53%	81.60%	97.32%	87.36%	91.39%
	KD+BU	70.40%	81.33%	98.90%	88.91%	93.77%
	LID	**82.38%**	**82.51%**	**99.78%**	**95.87%**	**98.93%**
SVHN	KD	70.39%	77.18%	99.57%	86.46%	87.41%
	BU	86.78%	84.07%	86.93%	91.33%	87.13%
	KD+BU	86.86%	83.63%	99.52%	93.19%	90.66%
	LID	**97.61%**	**87.55%**	**99.72%**	**95.07%**	**97.60%**

Table 2: The failure rate of an adaptive attack targeting the LID based detector.

	MNIST	CIFAR-10	SVHN
Scenario 2:Attack Failure Rate (one-layer)	100%	95.7%	97.2%
Scenario 1:Attack Failure Rate (all-layer)	100%	100%	100%

size could further improve the performance (as measured by AUC) of the LID method. Figure 5 in the Appendix (A.3) shows the effect of using a minibatch size of 1000 for different choices of k. It illustrates that a larger batch size can improve detection AUC even further — we conjecture that this is because it facilitates a more accurate estimation of LID (as discussed in Section 4). A comprehensive investigation of the choice of minibatch size is an interesting area for future work.

Robustness to Adaptive Attack: To further evaluate the robustness of the LID based detector, we applied an adaptive Opt attack in a white-box setting to test the limits of our LID detector. Similar to the strategy used to attack the KD based detector in Carlini & Wagner (2017a), we used an Opt L_2 attack with a modified adversarial objective:

$$\text{minimize } \|x - x_{adv}\|_2^2 + \alpha \cdot \big(\ell(x_{adv}) + \ell(\text{LID}(x_{adv}))\big) \tag{5}$$

where α is a constant balancing between the amount of perturbation and the adversarial strength, and the estimated LID is that of the pre-softmax layer. We test two different scenarios for detection. In the first scenario, we use LID features as described in Algorithm 1. In the second scenario, we use just estimated LID of the pre-softmax layer (this scenario is of interest because the Opt attack is known to attack the pre-softmax layer, and such a strategy enables a fair comparison (Carlini & Wagner, 2017b;a) to be made). The optimal constant α is chosen via an internal binary search for $\alpha \in [10^{-3}, 10^6]$. The rationale for the minimization of the LID characteristic in Equation (5) is that adversarial examples have higher LID than normal examples, as we demonstrated in Section 5.1.

We applied the adaptive attack on 1000 normal images randomly chosen from the detection test set. The deep networks used were the same ConvNets as used in our previous experiments. Instead of AUC as measured in the previous sections, we report accuracy here to evaluate attack performance, following the suggestion of Carlini & Wagner (2017a). We see in Table 2 that the adaptive attack for scenario 2 fails to find any valid adversarial example 100%, 95.7% and 97.2% of the time on MNIST, CIFAR-10 and SVHN, respectively. In addition, when trained on all transformation layers (scenario 1), the LID-based detector still correctly detected the attacks 100% of the time. Based on these results, we can conclude that integrating LID into the adversarial objective (increasing the complexity of the attack) does not make detection more difficult for our method. This is in contrast to the work of Carlini & Wagner (2017a), who showed that incorporating kernel density into the objective function makes detection substantially more difficult for the KD method.

6 CONCLUSION

In this paper, we have addressed the challenge of understanding the properties of adversarial subspaces, particularly with a view to discriminate adversarial examples. We characterized the intrinsic dimensional property of adversarial subspaces via the use of Local Intrinsic Dimensionality (LID), and showed how these could be used to discriminate adversarial examples generated using the state-of-the-art attacks. From a theoretical perspective, we have provided an initial intuition as to how adversarial perturbation can affect the LID characteristic of an adversarial subspace. Our empirical results suggest that LID is a highly promising measurement for characterizing adversarial examples that can be used to deliver state-of-the-art discrimination performance. Further investigation in this direction may lead to new techniques for both adversarial attack and defense.

REFERENCES

Laurent Amsaleg, Oussama Chelly, Teddy Furon, Stéphane Girard, Michael E. Houle, Ken-ichi Kawarabayashi, and Michael Nett. Estimating local intrinsic dimensionality. In *SIGKDD*, pp. 29–38. ACM, 2015.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. *arXiv preprint arXiv:1705.07263*, 2017a.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *S&P*, pp. 39–57, 2017b.

Stuart Coles, Joanna Bawa, Lesley Trenner, and Pat Dorazio. *An introduction to statistical modeling of extreme values*, volume 208. Springer, 2001.

Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on machine learning models. *arXiv preprint arXiv:1707.08945*, 2017.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *NIPS*, pp. 1632–1640, 2016.

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

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

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

Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defenses: Ensembles of weak defenses are not strong. *arXiv preprint arXiv:1706.04701*, 2017.

Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine*, 29(6):82–97, 2012.

Michael E. Houle. Local intrinsic dimensionality I: an extreme-value-theoretic foundation for similarity applications. In *SISAP*, pp. 64–79, 2017a.

Michael E. Houle. Local intrinsic dimensionality II: multivariate analysis and distributional support. In *SISAP*, pp. 80–95, 2017b.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pp. 1097–1105, 2012.

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

Xin Li and Fuxin Li. Adversarial examples detection in deep networks with convolutional filter statistics. *arXiv preprint arXiv:1612.07767*, 2016.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

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

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, pp. 427–436, 2015.

Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. cleverhans v1. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 2016a.

Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016b.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *EuroS&P*, pp. 372–387, 2016c.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *S&P*, pp. 582–597, 2016d.

Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1528–1540, 2016.

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.

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

Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017.

David Warde-Farley, Ian Goodfellow, T. Hazan, G. Papandreou, and D. Tarlow. Adversarial perturbations of deep neural networks. *Perturbations, Optimization, and Statistics*, pp. 1–32, 2016.

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

A Appendix

A.1 Statistics of Adversarial Attacks

Table 3: The L_2 mean perturbation and model accuracy on adversarial examples.

	MNIST		CIFAR		SVHN	
	L_2	Acc.	L_2	Acc.	L_2	Acc.
FGM	7.26	11.09%	2.74	3.15%	7.09	6.17%
BIM-a	2.30	10.43%	0.48	0.00%	0.83	0.13%
BIM-b	3.42	10.42%	2.39	0.00%	5.53	0.13%
JSMA	5.40	10.00%	3.64	0.04%	3.09	0.16%
Opt	4.21	3.92%	0.37	0.01%	0.59	0.26%

A.2 LID CHARACTERISTICS OF CURRENT ADVERSARIAL EXAMPLES

Figure 4 illustrates LID characteristics of the most effective attack to date, i.e., Opt attack on MNIST and SVHN datasets. On both datasets, the LIDs of adversarial examples are significantly higher than that of normal or noisy examples. In the right subfigure, the LIDs of normal examples and its noisy counterparts were overlapped.

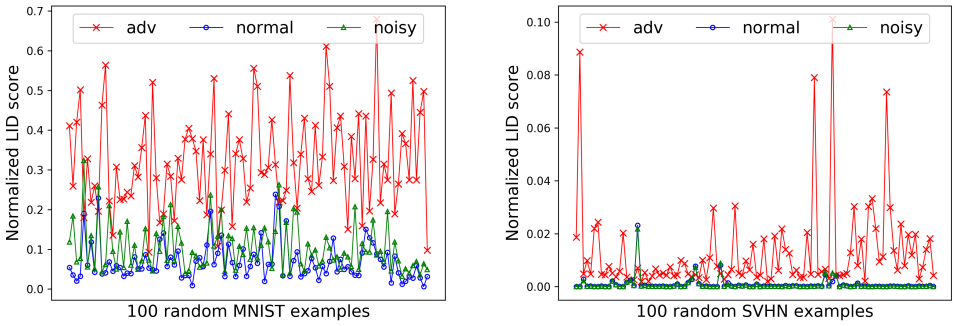

Figure 4: The plots show the normalized LID scores of 100 randomly selected normal (blue), noisy (green) and Opt attack (red x-cross) examples. The noisy and adversarial examples were generated from the normal examples. The left-hand figure shows the scores (for the pre-softmax layer) of MNIST examples, while the right-hand figure shows LID scores (after-softmax) of SVHN examples. Normal and noisy example curves are superimposed in the right-hand figure.

A.3 EFFECT OF LARGER MINIBATCH SIZE FOR LID

Figure 5 shows the discrimination power (detection AUC) of LID characteristics estimated using two different minibatch sizes (the default setting of 100, and a larger size of 1000). The horizontal axis represents different k nearest neighbors used, from 10% to 90% percent to the batch size. We that the peak AUC is higher for the larger minibatch size.

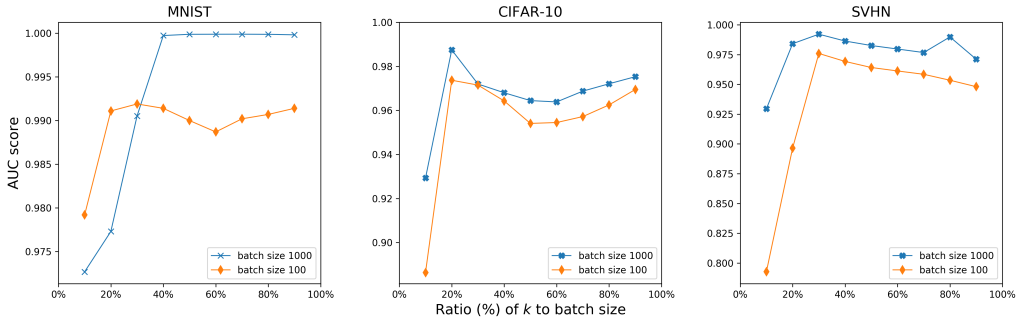

Figure 5: The detection AUC score of LIDs estimated using different k with a large batch size of 1000. The results are shown for detecting Opt attack on MNIST, CIFAR-10 and SVHN datasets.