EXPLORING LOTS IN DEEP NEURAL NETWORKS

Andras Rozsa, Manuel Günther, and Terrance E. Boult
Vision and Security Technology (VAST) Lab
University of Colorado
Colorado Springs, USA
{arozsa,mgunther,tboult}@vast.uccs.edu

ABSTRACT

Deep neural networks have recently demonstrated excellent performance on various tasks. Despite recent advances, our understanding of these learning models is still incomplete, at least, as their unexpected vulnerability to imperceptibly small, non-random perturbations revealed. The existence of these so-called adversarial examples presents a serious problem of the application of vulnerable machine learning models. In this paper, we introduce the layerwise origin-target synthesis (LOTS) that can serve multiple purposes. First, we can use it as a visualization technique that gives us insights into the function of any intermediate feature layer by showing the notion of a particular input in deep neural networks. Second, our approach can be applied to assess the invariance of the learned features captured at any layer with respect to the class of the particular input. Third, we can also utilize LOTS as a general way of producing a vast amount of diverse adversarial examples that can be used for training to further improve the robustness and performance of machine learning models. However, we show that the improvement with respect to adversarial robustness to different adversarial types varies and highly depends on the type used for training – some adversarial generation techniques can be almost completely immune to adversarial training.

1 INTRODUCTION

Due to tremendous progress over the last several years, the most advanced deep neural networks (DNNs) have managed to approach and even surpass human level performance on a wide range of challenging machine learning tasks (Parkhi et al., 2015; Schroff et al., 2015; Szegedy et al., 2015; He et al., 2016). Despite the fact that we are able to design and train learning models that perform well, our understanding of these complex networks is still incomplete. This was highlighted by the discovery of the intriguing properties of machine learning models by Szegedy et al. (2014).

Gaining intuitive insights and, thus, building a better understanding of how these models work has a long history in the literature. For visual recognition tasks, various techniques have been proposed to address the problem of understanding which kind of features are captured and used by learning models (Erhan et al., 2009; Mahendran & Vedaldi, 2016) and how the different internal representations of object classes or input images can be better visualized (Zeiler & Fergus, 2014; Yosinski et al., 2015; Simonyan et al., 2014; Mahendran & Vedaldi, 2016).

While exploring the internal details of DNNs in order to further advance their performance via visualization is particularly relevant and the subject of this paper, recent research has also been focusing on the unpleasant properties revealed by Szegedy et al. (2014). Namely, machine learning models, including the best performing DNNs, suffer from a highly unexpected instability as they can confidently misclassify adversarial examples that are formed by adding imperceptible, non-random perturbations to otherwise correctly classified inputs. As DNNs are expected to be robust to small perturbations to their inputs due to their excellent generalization capabilities, the existence of such adversarial perturbations challenges our understanding of DNNs and questions the utility of such vulnerable learning models in real-world applications. Researchers proposed various techniques to reliably find adversarial perturbations (Szegedy et al., 2014; Goodfellow et al., 2015; Sabour et al., 2016; Rozsa et al., 2016), and demonstrated that adversarial examples can serve a good purpose as

well, as they can be successfully used for training to improve both the overall performance and the robustness of machine learning models (Goodfellow et al., 2015; Rozsa et al., 2016).

In this paper, we introduce the layerwise origin-target synthesis (LOTS) that can be efficiently used for multiple purposes. First, it can be applied to visualize the internal representation of an input captured by the learning model at any particular layer. Second, LOTS is capable of forming a vast amount of diverse adversarial examples for each input and we demonstrate that such diversity can be beneficial for adversarial training. We show that the improvement achieved by adversarial training is limited to certain types of adversarial generation techniques. Finally, derived from the previous utilization, LOTS can be used to explore and assess how stable a particular internal representation of an input is with respect to the class of the input. This paper introduces our novel approach, compares it to related work, and highlights its benefits and possible directions of future work.

2 Related Work

While deep neural networks (DNNs) have achieved excellent performance on various tasks, there is an increasing interest to alleviate their vulnerability to adversarial examples. To better understand the sometimes unexpected behavior of learning models, visualizing the learned features is a common practice for gaining intuitive insights. As our work is related to the visualization of feature representations and adversarial instability, this section discusses the relationships to both areas.

2.1 Visualization

In order to design and train better performing machine learning models, it is useful to have a clearer understanding about the internal operations and behaviors of these complex models. Otherwise, research aiming to develop more advanced learning models remains restricted to trial-and-error.

Visualization of DNNs was pioneered by Erhan et al. (2009) who displayed high level features learned by various models at the unit level by finding the optimal stimulus in the image space that maximizes the activation of each particular unit. While their approach requires careful initialization, it cannot provide information about the invariance of the inspected units. To address this short-coming, Simonyan et al. (2014) demonstrated how image-specific class saliency maps can be obtained from the last fully connected layers of DNNs, showing areas of the image that are discriminative with respect to a given class. The authors also introduced a technique – image inverting – to generate artificial images that maximize the selected class score. Related to saliency maps, Girshick et al. (2014) showed that identifying regions of images yielding high activations at higher layers can be successfully used for object detection. Zeiler & Fergus (2014) extended visualization to convolutional features. Furthermore, their approach not only finds patches of input images that stimulate a particular feature map, but is also capable of revealing structures within those patches by using a top-down projection.

Although these visualization techniques helped to gain insights into how and why DNNs might work, researchers proposed various approaches to enhance the produced representations commonly called pre-images. Yosinski et al. (2015) visualized activations on each layer for an image or video, and introduced methods to produce qualitatively clearer and more interpretable visual representations. Similarly, Mahendran & Vedaldi (2016) presented regularized visualizations for maximized activations and inverted images in order to obtain natural looking pre-images.

Using LOTS we explore how the extracted features of an image at any given layer translate back to the input space, and how invariant those features are with respect to the class of a particular input.

2.2 Adversarial Representations

Szegedy et al. (2014) revealed that machine learning models, including state-of-the-art DNNs, are vulnerable to adversarial examples – formed by applying imperceptibly small, non-random perturbations to otherwise correctly recognized inputs – which lead to misclassifications. This discovery fundamentally challenged our understanding of DNNs, namely, the excellent performance achieved by these complex models was believed to be due to their capability of learning features from the training set that generalize well. The existence of adversarial examples highlights that machine learning models are in fact not robust, and adversarial instability needs to be addressed.

Since Szegedy et al. (2014) presented the problem and introduced the first method that is able to reliably find adversarial perturbations, various approaches were proposed in the literature. Compared to the computationally expensive box-constrained optimization technique (L-BFGS) of Szegedy et al. (2014), a more lightweight, yet effective technique was introduced by Goodfellow et al. (2015). The proposed fast gradient sign (FGS) method relies on the sign of the gradient of loss which needs to be calculated only once in order to form an adversarial perturbation. The authors also demonstrated that by using FGS examples implicitly in an enhanced objective function, both the overall performance and the robustness of the trained models can be improved. Later, Rozsa et al. (2016) demonstrated that by not using the sign, the formalized fast gradient value (FGV) approach produces significantly different adversarial samples compared to FGS.

All of the aforementioned adversarial example generation techniques rely on ascending the gradient of loss used for training the network, namely, the formed perturbation causes misclassification by increasing the loss until the particular original class does not have the highest prediction probability. Methods that do not rely on the use of the gradient of loss used for training were also proposed by researchers. The approach of Sabour et al. (2016) produces adversarial images that not only cause misclassifications but mimic the internal representations of the targeted original inputs. Again, their technique uses the computationally expensive L-BFGS algorithm. Rozsa et al. (2016) introduced the hot/cold approach which causes recognition errors by not only reducing the prediction probability of the original class of the input, but by aiming to magnify the probability of a specific target class. To do so, the hot/cold approach defines a Euclidean loss on the penultimate layer, and uses its gradients with varying target classes as directions for finding adversarial perturbations. Therefore, this approach is capable of producing multiple diverse adversarial examples for each input. The authors demonstrated that using a diverse set of such adversarial examples formed with perturbations with higher magnitude than the sufficient minimum necessary to cause misclassifications can outperform regular adversarial training. Finally, Rozsa et al. (2016) proposed a new psychometric called perceptual adversarial similarity score (PASS) to better measure adversarial quality, in other words, the distinguishability or similarity of original and adversarial image pairs in terms of human perception.

Our novel LOTS method can be considered as a general extension of the hot/cold approach to deeper layers, and it also shows similarities to the technique of Sabour et al. (2016) in terms of directly adjusting internal feature representations – without requiring the use of the L-BFGS algorithm.

3 APPROACH

One way of gaining insights into the operation of a deep neural network is by letting the network process a given image and – after modifying the output of the network – projecting this modification back to the input level, e.g., via backpropagation. While changing the output appears to be straightforward due to our clear(er) understanding of its meaning, e.g., the hot/cold approach (Rozsa et al., 2016) modifying logits, theoretically, the modification can happen at any layer or any neuron of the network. However, the interpretation of such modifications might be more difficult as the output of a given layer or neuron per se is not guaranteed to have a semantic meaning.

Formally, let us consider a network f with weights w in a layered structure, i.e., having layers $y^{(l)}, l = \{1, \dots, L\}$, with their respective weights $w^{(l)}$. For a given input x, the output of the network can be formalized as:

$$f(x) = y^{(L)}\left(y^{(L-1)}\left(\dots\left(y^{(1)}(x)\right)\dots\right)\right), \tag{1}$$

while the internal representation of the given input x at layer l is:

$$f^{(l)}(x) = y^{(l)}\left(y^{(l-1)}\left(\dots\left(y^{(1)}(x)\right)\dots\right)\right). \tag{2}$$

Our layerwise origin-target synthesis (LOTS) approach aims to adjust the internal representation of an input x_o, the *origin*, to get closer to the internal representation associated with input x_t, the *target*. To do so, we use a Euclidean loss defined on the captured internal representations of the origin and target at layer l, and apply its gradient with respect to the origin to manipulate the internal features of the origin, formally:

$$\eta^{(l)}(x_o, x_t) = \nabla_{x_o}\left[\frac{1}{2}\sum_i\left(f_i^{(l)}(x_t) - f_i^{(l)}(x_o)\right)^2\right]. \tag{3}$$

We can use the direction defined by this gradient and form adversarial perturbations using a line-search – similar to the fast gradient sign (FGS) method (Goodfellow et al., 2015) and the hot/cold approach (Rozsa et al., 2016). Compared to previous techniques, LOTS has the potential to form dramatically more, and more diverse perturbations for each input due to the large amount of possible targets and the number of layers it can be used on.

Alternatively, LOTS can be used on the origin itself in order to magnify or reduce the internal representation of input x_o at layer l. This can be formalized as:

$$\hat{\eta}^{(l)}(x_o) = \nabla_{x_o} \left[\pm \frac{1}{2} \sum_i \left(f_i^{(l)}(x_o) \right)^2 \right]. \tag{4}$$

While the directions defined by these gradients can be utilized to form adversarial perturbations, they can also be used to visualize the captured internal representations of the input at layer l.

4 VISUALIZATION VIA LOTS

First, we focus on demonstrating the visualization capabilities of LOTS, exploring the captured internal representations of inputs on different DNNs. We investigate two publicly available deep neural networks: the 4-layer LeNet network from LeCun et al. (1995) trained on the MNIST dataset (LeCun et al., 1998), and the 16-layer VGG face recognition network from Parkhi et al. (2015).

To quantify the perturbed images, we use three metrics: L_2 and L_∞ norms of the perturbations, as well as the perceptual adversarial similarity score (PASS) (Rozsa et al., 2016) between origin x_o and the perturbed image $x_o^\pm$. While L_2 and L_∞ norms focus strictly on the perturbations regardless of how visible or hidden those are on the distorted images, PASS was designed to quantify the similarity of original and perturbed image pairs with respect to human perception. The calculation of PASS takes two steps: alignment by maximizing the enhanced correlation coefficient (ECC) (Evangelidis & Psarakis, 2008) of the image pair with homography transform $\Psi(x_o^\pm, x_o)$, followed by quantifying similarity between the aligned adversarial and original images using the structural similarity index (SSIM) (Wang et al., 2004). Consequently, PASS can be formalized as follows:

$$\text{PASS}\left(x_o^\pm, x_o\right) = \text{SSIM}\left(\Psi\left(x_o^\pm, x_o\right), x_o\right), \tag{5}$$

where $\text{PASS}(x_o^\pm, x_o) = 1$ indicates perfect similarity. Note that throughout our experiments, we form perturbed images that have discrete pixel values in $[0, 255]$.

4.1 INTERNAL REPRESENTATION OF HANDWRITTEN DIGITS

To visualize the captured internal representations of an input at a given layer of the LeNet network, we apply Equation (4) on each layer. When we use the positive sign to form a perturbed image x_o^+, the captured internal representation is magnified and the classification of the modified digit will usually not change. On the other hand, by using the negative sign in Equation (4) for forming a perturbed image x_o^-, we can display which kind of modifications would be required to inhibit the internal representation of origin x_o at that particular layer. Consequently, by increasing the magnitude of the perturbation, the network is more likely to classify x_o^- differently than the origin x_o as feature representations of the original class are fading away.

Figure 1 displays the visualized internal representations of an exemplary MNIST image with label 4, shown in Figure 1(a), captured by the LeNet network. The x_o^+ samples are displayed in Figures 1(b) - (e). Please note that magnifying the captured internal feature representations does not lead to altered classifications, therefore, these examples are optimized for visualization purposes. The perturbations shown in Figures 1(b) and 1(c) demonstrate that those convolutional layers have learned structural characteristics of the input, and increasing the captured internal representations of the origin gracefully translates to a thicker digit. Contrarily, the perturbations formed on the fully-connected layers displayed in Figures 1(d) and 1(e) seem to have slightly lost spatial focus by considering several locations in the image that appear to be unrelated for the classification of this particular digit. The x_o^- examples are presented in Figures 1(f) - (i), where perturbations have the sufficient minimal magnitude leading to a class label different than the origin's. As demonstrated by the perturbed images shown in Figures 1(h) and 1(i), inhibiting the captured internal representations of this particular digit yields plausible adversarial examples as those perturbations are hardly noticeable to us.

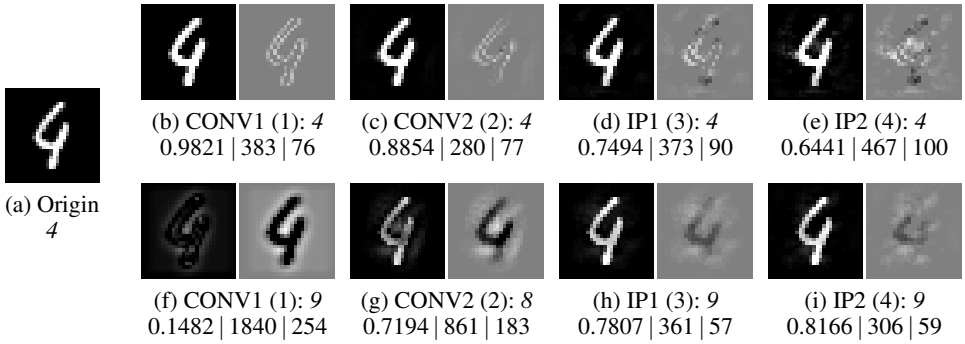


Figure 1: VISUALIZATION VIA LOTS ON LENET. *This figure shows perturbed images generated using LOTS by modifying the internal representations of an MNIST image x_o, shown in (a), captured at various layers in a trained LeNet model. Each subfigure shows the formed image $x_o^{\pm}$ and the perturbation, while the sub-captions show the name of the layer followed by an index in parenthesis indicating its position in the network architecture, the classification of $x_o^{\pm}$, the PASS score between origin and the perturbed image, and the L_2 and L_∞ norms of the perturbation. (b)–(e) show x_o^+ images having magnified internal representations with perturbations scaled such that $L_\infty = 128$. Note that as pixels are constrained to $[0, 255]$, actual L_∞ norms can be smaller. (f)–(i) contain reduced internal representations x_o^- formed by perturbations having the smallest magnitude that changes the class label.*

4.2 INTERNAL REPRESENTATION OF FACES

LOTS is capable of displaying more difficult objects, and can be used to visualize more complex classes than the digits of MNIST. To demonstrate this, we generate x_o^- and x_o^+ images for various layers of the VGG Face network (Parkhi et al., 2015). Figure 2 shows some distorted samples with the corresponding perturbations generated by modifying the internal representations captured at various layers of the VGG Face network, for an exemplar image taken from the VGG Face Dataset (Parkhi et al., 2015) shown in Figure 2(a). Figures 2(b) - (f) contain x_o^- samples, while Figures 2(g) - (i) show x_o^+ representations.

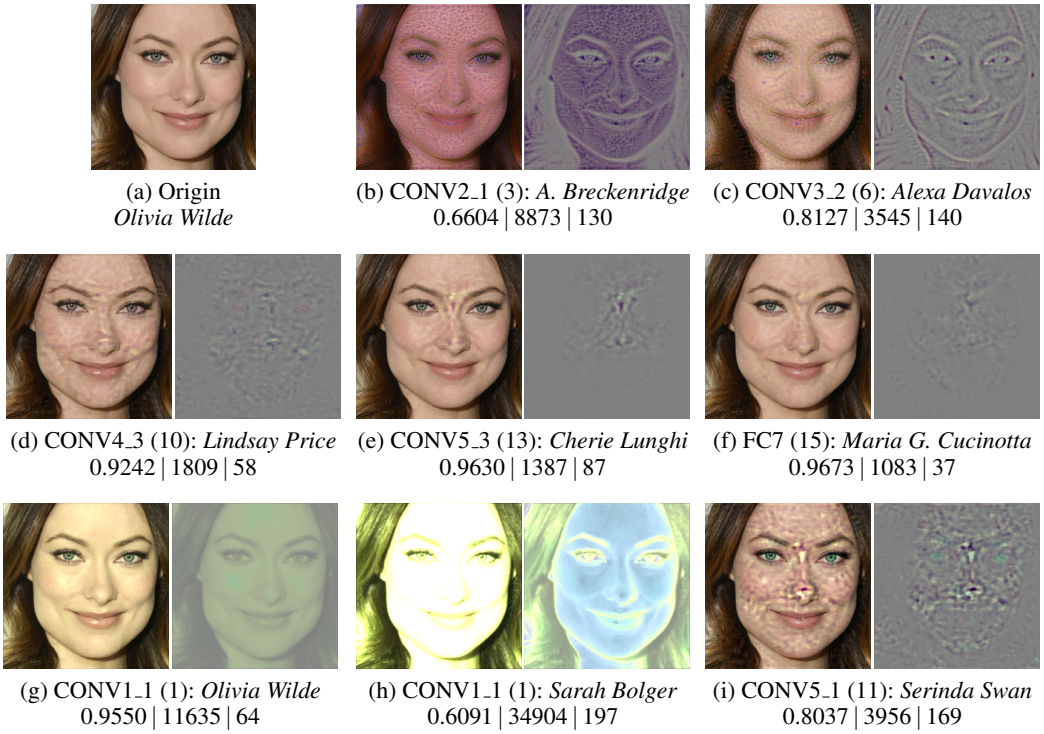
We can observe that due to the large number of identities (classes) present in the VGG Face Dataset, perturbations lead to various class labels. Also, in opposition to the previously described MNIST samples, both x_o^- and x_o^+ representations can yield altered classifications, which means that overemphasizing the captured features of a given identity also changes the classification of the network, however, these types of samples require perturbations with higher magnitudes.

In summary, the visualized internal feature representations of the origin suggest that lower convolutional layers of the VGG Face model have managed to learn and capture features that provide semantically meaningful and interpretable representations to human observers. Although we can still recognize facial features and parts on visualized feature representations captured at higher layers, closer to the top layer those become harder to interpret. This can be due to the fact that closer to the last layer the model needs features that allow for differentiating identities from one another, hence, those features can be small for visualization, and even semantically meaningless for us.

5 LOTS OF ADVERSARIAL EXAMPLES

Now, let us exploit the instability of DNNs and show how LOTS can be applied to generate adversarial examples. For a given origin image x_o, we aim to form an imperceptibly small perturbation which makes the DNN classify the perturbed image differently than the origin. In order to do so, we can exploit Equation (3) to compute the gradient of the Euclidean loss using the feature difference of the origin and target captured at any particular layer, and use that direction to form the adversarial image such that the perturbation has the lowest magnitude necessary for altering the classification or, if desired, reaching the targeted class.

For adversarial example generation, we use the same networks as before: we commence our experiments forming adversarial perturbations for MNIST digits that exploit the vulnerability of the LeNet model, we compare LOTS with other adversarial example generation techniques including the use



(a) Origin
Olivia Wilde

(b) CONV2_1 (3): *A. Breckenridge*
0.6604 | 8873 | 130

(c) CONV3_2 (6): *Alexa Davalos*
0.8127 | 3545 | 140

(d) CONV4_3 (10): *Lindsay Price*
0.9242 | 1809 | 58

(e) CONV5_3 (13): *Cherie Lunghi*
0.9630 | 1387 | 87

(f) FC7 (15): *Maria G. Cucinotta*
0.9673 | 1083 | 37

(g) CONV1_1 (1): *Olivia Wilde*
0.9550 | 11635 | 64

(h) CONV1_1 (1): *Sarah Bolger*
0.6091 | 34904 | 197

(i) CONV5_1 (11): *Serinda Swan*
0.8037 | 3956 | 169

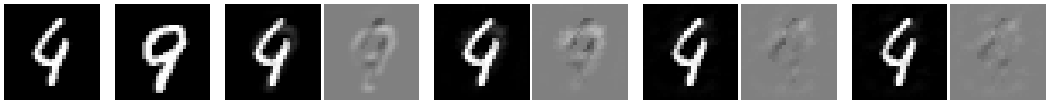
Figure 2: VISUALIZATION VIA LOTS ON VGG FACE. *This figure shows perturbed images generated using LOTS by modifying the internal representations of a face image x_o shown in (a), captured at various layers of the trained VGG Face model. Each subfigure shows a distorted image $x_o^{\pm}$ and its perturbation. The sub-captions list the name of the layer followed by an index in parenthesis indicating its position in the network architecture, the classification of $x_o^{\pm}$, the PASS score between origin and the distorted image, and the L_2 and L_∞ norms of the perturbation. (b)–(f) contain reduced internal representations x_o^- with perturbations having the sufficient minimal magnitudes to change the classification. (g) serves purely visualization purposes as its corresponding perturbation is scaled to $L_\infty = 64$. (h) and (i) show x_o^+ images that magnify the captured internal representations with sufficiently large perturbations that result in altered classifications by the model.*

for adversarial training, and then we turn to the more challenging task of manipulating internal representations of faces to cause misclassifications on the VGG Face model.

5.1 ADVERSARIAL EXAMPLES OF HANDWRITTEN DIGITS

Let us consider an image of digit 4 as the origin that we would like to turn to a 9 – at least, its LeNet classification. After selecting an applicable image of 9 for being the target, we can use the extracted feature representations of the target and the origin to form the adversarial perturbation utilizing the direction specified by Equation (3). As shown in Figure 3, we use the same origin image as in Figure 1(a) and a selected target image (cf. Figure 3(b)) that looks relatively similar to the origin, but is clearly from another class.

When generating the adversarial example x_o^t by manipulating the internal feature representations captured at the first convolution layer (CONV1), we can clearly see that the perturbation focuses on the difference between the origin 4 and the target 9, as shown in Figure 3(c). Moving to higher layers (CONV2, IP1 and IP2), we can observe that perturbations lose the focus more and more on the global structure, i.e., they are not focusing on the digit. Interestingly, the perturbations for this particular origin-target pair at higher layers have smaller L_2 and L_∞ norms, but also lower PASS scores, which indicates that the perturbed images generated at lower layers are less distinguishable from the origin, in other words, they are better in terms of adversarial quality. This might be due to the fact that the origin-target pair is indeed very similar with respect to the shapes of those digits – in other words, they are gracefully aligned – even though they perfectly represent their distinct classes.



(a) Origin *4* (b) Target *9* (c) CONV1 (1): *9* 0.9514 | 283 | 59 (d) CONV2 (2): *9* 0.9301 | 289 | 56 (e) IP1 (3): *9* 0.8942 | 173 | 40 (f) IP2 (4): *9* 0.8750 | 158 | 31

Figure 3: ADVERSARIAL PERTURBATIONS VIA LOTS ON LENET. *This figure shows adversarial images and their corresponding perturbations of handwritten digits generated with LOTS using the origin from (a) and target from (b). Each subfigure shows the adversarial image x_o^t and the perturbation, while the subcaptions show the name of layer followed by an index in parenthesis indicating its position in the network architecture, the classification of x_o^t, the PASS score between origin and the adversarial image, and the L_2 and L_∞ norms of the perturbation.*

5.2 COMPARISON OF ADVERSARIAL TYPES

In order to compare LOTS with previous adversarial generation techniques, we have conducted experiments on LeNet to measure the quantity and the adversarial quality of the produced samples. Furthermore, we have analyzed the effects of various adversarial types on error rates and adversarial robustness by using these samples for adversarial training.

First, we have trained a LeNet model on the MNIST Training dataset (60k digits), and we have also generated 2M images with InfiMNIST (Loosli et al., 2007) and trained a network for 100k iterations with the same hyperparameters distributed with Caffe (Jia et al., 2014). The former network is denoted as the Basic LeNet and the latter is listed as InfiMNIST in Table 1.

To compare LOTS with previously introduced adversarial generation techniques with respect to adversarial quality, we have formed perturbed images using the fast gradient sign (FGS) (Goodfellow et al., 2015) and the fast gradient value (FGV) (Rozsa et al., 2016) methods – both utilizing the gradient of loss used for optimization – the hot/cold (HC) approach (Rozsa et al., 2016) and our novel LOTS method on the MNIST Test dataset (10k digits). On the generated samples we have collected four metrics for each adversarial type, respectively: the success rate showing the percentage of all attempts the particular technique can produce a perturbation which changes the classification, PASS quantifying the adversarial quality in terms of imperceptibility with respect to human perception, and L_2 and L_∞ norms of the perturbations.

Using FGS, FGV, and HC, we have generated all possible perturbations having the smallest magnitudes that yield misclassifications. This translates into 1 possible perturbation with both FGS and FGV methods, and 9 possible HC samples for each image of the MNIST Test dataset. For LOTS, we have limited the targets by selecting one image per class – resulting in "only" 36 possible perturbations for each input. Utilizing all possible target images is impractical, even on a small dataset like MNIST. Considering the collected metrics on the two baseline networks – Basic LeNet and InfiMNIST in Table 1 – we can observe that LOTS maintains high success rates (99.56% and 99.07%) comparable to other adversarial generation methods, while the achieved mean adversarial quality (0.6561 and 0.6667) is better than obtained with FGS (0.4303 and 0.4553), and is worse than FGV (0.7753 and 0.7819) and HC (0.7269 and 0.7417) samples have. Interestingly, we have found that LOTS utilizing FGV or HC samples as targets can produce perturbed images with adversarial quality surpassing the originating adversarial images – indicated by higher PASS scores – but these "improved" samples do not provide any further improvements over their ancestors when used for adversarial training.

To analyze the performances of adversarial generation techniques when the formed samples are used for training, we have generated perturbed images for the MNIST Training dataset on our Basic LeNet model and then used those images for fine-tuning it. Since we have various numbers of perturbed images for each adversarial type, we have fine-tuned Basic LeNet using the regular hyperparameters for different number of iterations: 20k iterations for FGS (app. 52k images) and FGV (app. 51k images) samples, 50k iterations for HC (app. 538k images), and 100k iterations for LOTS (app. 2.1M images) samples generated on the 4 layers of LeNet. For LOTS, we have selected one image per class and used the same 10 images as targets. Considering all possible targets, the overall number of perturbed images would be beyond 10 billion samples.

Table 1: ADVERSARIAL TRAINING ON LENET. *This table contains collected metrics of adversarial images generated on the MNIST Test dataset using FGS, FGV, HC, and LOTS on corresponding models. The table contains metrics for a Basic LeNet model trained regularly, and for InfiMNIST, a LeNet model trained with 2M digits. Other models were obtained by generating adversarial examples for the MNIST Training dataset on Basic LeNet and using those samples to fine-tune Basic LeNet with the adversarial type listed as the model name. For each adversarial type success rate, PASS, and, below, L_2 and L_∞ norms are listed.*

Model	Error	FGS	FGV	HC	LOTS
Basic LeNet	0.91%	85.37% | 0.4303 ± 0.1098 718.8 ± 281.2 | 34.6 ± 13.2	85.37% | 0.7753 ± 0.1050 435.3 ± 187.9 | 94.4 ± 38.7	99.92% | 0.7269 ± 0.1102 596.0 ± 275.5 | 117.5 ± 47.2	99.56% | 0.6561 ± 0.1245 1004.0 ± 456.8 | 170.0 ± 52.4
InfiMNIST	0.53%	92.89% | 0.4553 ± 0.1069 729.6 ± 260.6 | 35.2 ± 12.5	92.88% | 0.7819 ± 0.0921 450.5 ± 181.7 | 94.6 ± 36.2	99.71% | 0.7417 ± 0.0944 566.0 ± 224.9 | 107.6 ± 40.4	99.07% | 0.6667 ± 0.1236 986.8 ± 473.7 | 163.6 ± 54.1
FGS	0.77%	80.69% | 0.4835 ± 0.1019 650.9 ± 255.8 | 33.6 ± 13.8	80.69% | 0.7576 ± 0.0997 427.7 ± 186.2 | 91.0 ± 39.7	100.00% | 0.7200 ± 0.0912 594.3 ± 249.0 | 116.8 ± 49.8	91.05% | 0.6457 ± 0.1138 1032.0 ± 393.4 | 179.6 ± 51.5
FGV	0.82%	60.52% | 0.4240 ± 0.1161 701.3 ± 256.0 | 35.2 ± 13.2	60.52% | 0.7722 ± 0.1083 434.5 ± 189.8 | 96.1 ± 39.1	99.95% | 0.7140 ± 0.1079 684.8 ± 311.2 | 135.9 ± 52.4	99.69% | 0.6361 ± 0.1242 1094.3 ± 431.3 | 188.8 ± 50.8
HC	0.78%	71.92% | 0.4527 ± 0.1116 683.3 ± 263.8 | 34.5 ± 13.7	71.92% | 0.7669 ± 0.1105 449.8 ± 231.7 | 95.0 ± 42.7	99.99% | 0.7140 ± 0.1064 653.1 ± 320.5 | 126.6 ± 53.2	99.89% | 0.6401 ± 0.1213 1074.0 ± 414.0 | 183.8 ± 51.9
LOTS	0.65%	30.89% | 0.4506 ± 0.1250 597.5 ± 220.1 | 29.4 ± 10.9	30.88% | 0.8208 ± 0.1044 341.1 ± 132.9 | 83.6 ± 32.7	97.26% | 0.7004 ± 0.1127 686.1 ± 271.8 | 143.1 ± 50.3	97.61% | 0.6285 ± 0.1239 1119.6 ± 422.9 | 194.0 ± 48.2
LOTS CONV1	1.02%	47.58% | 0.4783 ± 0.1316 486.7 ± 185.4 | 23.9 ± 9.1	47.58% | 0.8144 ± 0.1030 295.3 ± 117.5 | 66.2 ± 26.7	100.00% | 0.7374 ± 0.1025 538.5 ± 231.5 | 106.4 ± 41.7	90.03% | 0.6471 ± 0.1357 1036.8 ± 538.8 | 171.6 ± 55.0
LOTS CONV2	0.90%	54.86% | 0.4679 ± 0.1263 533.6 ± 192.5 | 26.4 ± 9.5	54.85% | 0.8169 ± 0.1047 317.6 ± 120.1 | 73.3 ± 28.4	99.92% | 0.7468 ± 0.1046 535.9 ± 216.1 | 110.2 ± 41.6	98.26% | 0.6518 ± 0.1328 1044.2 ± 500.6 | 177.9 ± 53.5
LOTS IP1	0.74%	46.16% | 0.4456 ± 0.1257 659.1 ± 238.9 | 32.5 ± 11.9	46.16% | 0.8049 ± 0.1131 386.5 ± 159.5 | 90.9 ± 36.2	99.33% | 0.7137 ± 0.1204 662.2 ± 256.8 | 136.2 ± 50.0	98.72% | 0.6319 ± 0.1261 1099.4 ± 416.2 | 189.7 ± 48.9
LOTS IP2	0.78%	46.43% | 0.4263 ± 0.1275 717.8 ± 266.8 | 35.6 ± 13.6	46.42% | 0.7724 ± 0.1209 427.5 ± 172.1 | 96.3 ± 37.5	99.39% | 0.6869 ± 0.1155 727.8 ± 292.7 | 144.7 ± 53.5	99.17% | 0.6239 ± 0.1215 1110.7 ± 387.2 | 191.8 ± 48.9

By looking at the results listed in Table 1, we can see that fine-tuning with LOTS samples achieves the lowest error rate (0.65%) on the MNIST Test dataset among models trained with adversarial examples – slightly worse than InfiMNIST (0.53%) – and this network is the most robust considering the collected metrics of all four adversarial types as well. Furthermore, we have compared the effects of LOTS samples formed by manipulating the captured feature representations at various layers. We have found that, in general, fine-tuning with LOTS samples originating from higher layers requires adversarial generation techniques to form stronger, more visible perturbations as indicated by the decreasing PASS scores and increasing L_2 and L_∞ norms on models denoted as LOTS CONV1, LOTS CONV2, LOTS IP1, and LOTS IP2, respectively.

Finally, we can observe that various adversarial types have significantly different effects on adversarial robustness against samples of other adversarial generation techniques. As highlighted by the collected metrics, the HC technique is completely immune to fine-tuning with perturbed images produced using FGS. Compared to the baseline networks (Basic LeNet and InfiMNIST in Table 1), on the network fine-tuned with FGS samples (FGS in Table 1) we have managed to generate quantitatively and qualitatively very similar samples via the HC approach. In summary, our results demonstrate that the achieved adversarial robustness to different adversarial types can vary and can also heavily depend on the particular adversarial type used for training.

5.3 ADVERSARIAL EXAMPLES OF FACES

A greater threat to automatic face recognition systems is presented by the possibility of generating adversarial examples for face images. In DNN-based face recognition systems, usually the last layer of the network (containing the identities) is disregarded, and the output of the penultimate layer (e.g., FC7 of the VGG Face model) is stored as a representation of the face. When an adversary steals this representation of a target (without requiring the possession of the target image) and also has access to the original network, he can generate an image that looks like himself to a human operator, but is identified as the target identity by the network.

Given the internal representation of a target image x_t representing identity t (*Sean Bean* in Figure 4(b)) at a given layer l of the network, and an origin image x_o including its internal representation of identity o (*Hugh Laurie* in Figure 4(a)), we can use Equation (3), and form the adversarial image x_o^t classified as identity t. As shown in Figure 4, the adversarial images can be basically indistinguishable from the original images as indicated by the very high PASS scores, yet the network classifies them incorrectly as the target.

(a) Origin *Hugh Laurie* (b) Target *Sean Bean* (c) FC6 (14): *Sean Bean* 0.9838 | 976 | 41 (d) FC7 (15): *Sean Bean* 0.9785 | 1136 | 48

Figure 4: ADVERSARIAL PERTURBATIONS VIA LOTS ON VGG FACE. *This figure shows adversarial images and their corresponding perturbations of face images generated with LOTS using the origin from (a) and target from (b). Each subfigure shows the adversarial image x_o^t and the perturbation, while the subcaptions show the name of the layer followed by an index in parenthesis indicating its position in the network architecture, the classification of x_o^t, the PASS score between origin and the adversarial example, and the L_2 and L_∞ norms of the perturbation.*

We would like to note that LOTS can be modified to perform better, however, it would be computationally more expensive. Namely, instead of calculating the gradient of the introduced Euclidean loss with respect to the origin once and using it throughout the whole line-search, we can consider recalculating and adjusting that direction multiple times. This "step-and-adjust" optimization could potentially further improve both the quality of the produced adversarial examples and the capability of LOTS to reach the target more frequently.

6 CONCLUSION

In this paper, we have presented our novel layerwise origin-target synthesis (LOTS) algorithm which can be efficiently used for multiple purposes as we have demonstrated on two well-known deep neural networks (DNNs): the hand-written digit recognition network LeNet (LeCun et al., 1995) and the face recognition network from VGG (Parkhi et al., 2015).

First, we can visualize the captured internal structure of an input at any layer of DNNs and we can also gain intuitive insights into the interpretation of a given input by DNNs by magnifying or inhibiting what the network at a particular layer "thinks" is or is not a good representation of the object. Second, the stability of the captured internal feature representations can be assessed with respect to the class of the input by exploring how robust they are to perturbations aiming at magnifying or reducing them. Third, we have demonstrated that LOTS is capable of producing a large number of diverse adversarial examples for each input by modifying the internal structure of the original input to mimic the particular target. This application of LOTS can help us get a deeper understanding about the intrinsic features that differentiate classes in various layers of DNNs.

We have conducted large-scale experiments to compare our novel LOTS approach with other adversarial generation techniques and have concluded that using the large number of diverse perturbed examples produced via LOTS for adversarial training outperforms previous methods with respect to both the achieved performance and the improved adversarial robustness. We have found that the overall performance of adversarial training with respect to improving adversarial robustness to different adversarial types varies depending on the chosen adversarial generation technique used to produce samples for training. Our results suggest that adversarial training might not be considered as a general solution to improve adversarial stability.

ACKNOWLEDGMENTS

This research is based upon work funded in part by NSF IIS-1320956 and in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

REFERENCES

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341, 2009.

Georgios D Evangelidis and Emmanouil Z Psarakis. Parametric image alignment using enhanced correlation coefficient maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(10):1858–1865, 2008.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587, 2014.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representation (ICLR)*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Interantional Conference on Multimedia*, pp. 675–678. ACM, 2014.

Yann LeCun, LD Jackel, Léon Bottou, Corinna Cortes, John S Denker, Harris Drucker, Isabelle Guyon, UA Muller, E Sackinger, Patrice Simard, et al. Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural networks: the statistical mechanics perspective*, 261:276, 1995.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits, 1998.

Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training invariant support vector machines using selective sampling. *Large scale kernel machines*, pp. 301–320, 2007. Project InfiMNIST available at `http://leon.bottou.org/projects/infimnist`. Accessed: January 5, 2017.

Aravindh Mahendran and Andrea Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision (IJCV)*, pp. 1–23, 2016.

O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference (BMVC)*, 2015.

Andras Rozsa, Ethan M. Rudd, and Terrance E. Boult. Adversarial diversity and hard positive generation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2016.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. In *International Conference on Learning Representation (ICLR)*, 2016.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823, 2015.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *International Conference on Learning Representations (ICLR) Workshop*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

Christian J. Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representation (ICLR)*, 2014.

Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. In *International Conference on Machine Learning (ICML) Workshop*, 2015.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision (ECCV)*, pp. 818–833. Springer, 2014.