Feature Space Perturbations Yield More Transferable Adversarial Examples

Nathan Inkawhich, Wei Wen, Hai (Helen) Li and Yiran Chen
Duke University
Electrical and Computer Engineering Department, Durham, NC 27708
{nathan.inkawhich,wei.wen,hai.li,yiran.chen}@duke.edu

Abstract

Many recent works have shown that deep learning models are vulnerable to quasi-imperceptible input perturbations, yet practitioners cannot fully explain this behavior. This work describes a transfer-based blackbox targeted adversarial attack of deep feature space representations that also provides insights into cross-model class representations of deep CNNs. The attack is explicitly designed for transferability and drives feature space representation of a source image at layer L towards the representation of a target image at L. The attack yields highly transferable targeted examples, which outperform competition winning methods by over 30% in targeted attack metrics. We also show the choice of L to generate examples from is important, transferability characteristics are blackbox model agnostic, and indicate that well trained deep models have similar highly-abstract representations.

1. Introduction

Many researchers have shown how to intentionally fool well trained deep learning algorithms with adversarial attacks [23, 5, 2, 20, 8]. The focus of most works is to have a maximally devastating effect on the model, while introducing minimal perturbation to the data. In the case of image-based data, the attacker works to disrupt the classification ability of the network with imperceptible perturbations to the image [23, 5, 1, 16, 15]. These attacks are of even greater concern because we as a community do not have a firm understanding of what is happening inside these largely uninterpretable deep models. However, attacks may also provide a way to study the inner-workings of such models.

One of the more difficult threat models for an adversary operating on CNNs is a blackbox targeted attack. In this scenario, the adversary only has access to inputs and outputs, with no working knowledge of the underlying weights or architecture. Even more, the adversary may choose the target class the oracle network misclassifies to. In this work, we design a black-box targeted adversarial attack for deep CNN models. We design the attack using the property of adversarial transferability, in which examples that are adversarial to one model will often be adversarial to other models [11, 18, 13]. *The unique aspect of this attack is that it explicitly perturbs the feature space of a deep model with the intentions of creating more transferable adversarial examples.* The intuition comes from the observation that intermediate features of well trained models are transferable [26]. Thus, perturbations of intermediate features may also be transferable.

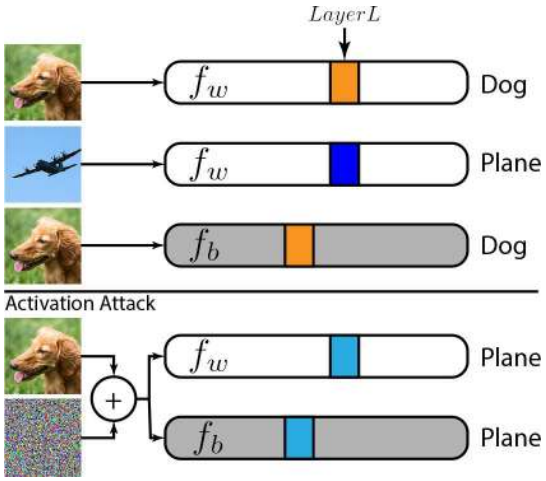

Figure 1: Illustration of Activation Attack. Given that the whitebox model (f_w) and blackbox model (f_b) are initially correct, the attack drives the layer L activations of the dog image towards the layer L activations of the plane. After attack, the dog's activations are similar to the plane's and the perturbed image is classified as a plane to f_w and f_b.

To test this hypothesis, we design the Activation Attack (AA), as shown in Fig. 1. By perturbing the source image, the algorithm drives the layer L activations of a whitebox model on the source image, towards the layer L activations of that model for a target image. The net effect is perturbations in feature space. We test the transferability of the perturbed features by feeding the perturbed image

into the blackbox model. These perturbations then have a not-directly-observable effect on the layer of the blackbox model that has learned those same features.

This work makes several contributions to adversarial attack and model interpretability research. First, we show that constructing adversarial examples with feature space perturbations yields transferable adversarial examples. Also, the layer at which features are perturbed has a large impact on the transferability of adversarial examples. Next, we use canonical CNN architectures to show the efficacy of the attack to powerful algorithms, rather than custom models for simple tasks (i.e. MNIST classifiers). For model interpretability, we show that blackbox model architecture does not affect the characteristics of layer-wise transferability. We also provide evidence for why a particular layer produces more transferable examples than another. Finally, this work gives testimony that intermediate feature representations of deep learning models with fundamentally different architectures are similar. Also, those well trained models have similar decision boundaries with similar class-wise orientations in feature space.

2. Related Work

One area of related work is adversarial attack research. The goal of many attacks is to introduce imperceptible perturbations to the input image that have devastating impacts to the classifiers performance, but have no impact to human recognition. Szegedy *et al*. [23] and Goodfellow *et al*. [5] were among the first to show that adversarial examples generated on one model may also transfer to other models. However, the transferability was not designed for, rather it was a consequence of the method and the nature of well trained models. Papernot *et al*. [18] developed a method to train a substitute model using reservoir sampling so that the substitute model generated more transferable examples. The work showed results across a spectrum of machine learning algorithms such as SVMs, decision trees, DNNs, and was less focused on the implication in modern CNNs.

More recently, Tramér *et al*. [25] explored transferable adversarial examples in more detail, and studied why transferability occurs through small custom MNIST models and untargeted attacks. Specifically, they found model decision boundaries are similar in arbitrary directions and adversarial examples span a low-dimensional subspace of the input. They also discuss perturbations of feature maps in directions of class-means, but results on CNNs are poor and not conclusive. Finally, the authors provide evidence that in some cases, well trained models for the same task, that both exhibit vulnerability to simple attacks, are likely to transfer adversarial examples. Our work differs in that the attacks are targeted, we perform extensive experimentation of layer-wise transferability on canonical CNNs for non-trivial problems, and we further analyze why examples transfer. Also related, Sabour *et al*. [21] were among the first to explicitly describe a whitebox only attack in feature space. They use an expensive L-BFGS perturbation method and show that examples of different classes can be forced very near each other in feature space but still maintain their original image structure. However, the authors briefly mention that the perturbed examples do not transfer well to blackbox models as targeted attacks and the attack is quite expensive.

In 2017 there was a competition for adversarial attacks and defenses [11] in which the goal of the attackers was to defeat a blackbox model with both non-targeted and targeted-attacks. Most of the top performing attacks [4, 11, 13, 14] generated and transferred adversarial examples from an ensemble of whitebox models, as introduced in [13, 24]. The winners also used a momentum gradient denoising technique [4], which is incorporated in this work. Most of the attacks were direct extensions of existing attacks to operate with an ensemble. We will not use ensembles here, but will rather focus on a new technique for perturbation that may be extended with ensembles as future work.

Another area of related work is in model interpretability, specifically in terms of shared feature representations of deep models. Yosinski *et al*. [26] showed that deep feature representations of models trained with data from similar distributions are transferable. They also measured the transition from generalized features to highly class specific (i.e. specialized) features as a function of layer depth. This now serves as a seminal work in transfer learning. Although [26] is not an adversarial attack, it does provide intuition as to why feature space attacks may transfer.

3. Transferability Metrics

In this work we will define success through four metrics: *error rate* (error), *untargeted transfer rate* (uTR), *targeted success rate* (tSuc), *targeted transfer rate* (tTR). Importantly, we assume that all of the examples are correctly classified by the whitebox and blackbox model, and an attack strength of $\epsilon = 0$ means no attack. We will denote the whitebox model as the function f_w and the blackbox model as f_b, both of which output a classification prediction. We define our original dataset $D_{orig} = \{(x^{(1)}, y^{(1)}_{true}), \ldots, (x^{(N)}, y^{(N)}_{true})\}$ as a set of N data/label pairs for which $f_b(x^{(i)}) = f_w(x^{(i)}) = y^{(i)}_{true}$. For each attack, we create an adversarial dataset $D_{adv} = \{(x^{(1)}_{adv}, y^{(1)}_{target}, y^{(1)}_{true}), \ldots, (x^{(N)}_{adv}, y^{(N)}_{target}, y^{(N)}_{true})\}$ where each data in D_{orig} has been perturbed by a targeted attack method on f_w, making it an adversarial example.

The *error rate* (error), or fooling rate of an attack is the percentage of adversarial examples generated with f_w that are misclassified by f_b. In other words, error is the percentage of examples in D_{adv} for which $f_b(x_{adv}) \neq y_{true}$.

Larger error indicates more effective attacks.

The *untargeted transfer rate* (uTR) is the rate at which the particular examples that fool the whitebox model also fool the blackbox model. Note, fooling in this context means the prediction does not equal the true label. To measure uTR, we define $D_{uTR} \subseteq D_{adv}$ which contains only the elements of D_{adv} that are misclassified by f_w. Thus,

$$uTR = \frac{1}{|D_{uTR}|} \sum_{\substack{(x_{adv}, y_{true}) \\ \in D_{uTR}}} \mathbb{1}[(f_b(x_{adv})) \neq y_{true})], \quad (1)$$

where $\mathbb{1}$ is the indicator function that is 1 if the condition is true and 0 otherwise. This metric intuitively encodes the likelihood that a successful untargeted adversarial example to the attacker's whitebox model will also be adversarial to the blackbox model.

Since the attacks considered are all targeted, we measure the *targeted success rate* (tSuc). tSuc is the rate at which adversarial examples generated with f_w are classified by f_b as the target label. In other words, tSuc is the percentage of examples in D_{adv} for which $f_b(x_{adv}) = y_{target}$. The larger tSuc, the more effective the attack at generating targeted examples for the blackbox model.

The final metric is *targeted transfer rate* (tTR), which measures the rate at which successful targeted adversarial examples measured on the whitebox model are also successful targeted examples on the blackbox model. For this, we define $D_{tTR} \subseteq D_{adv}$ (also a subset of D_{uTR}) which contains all elements of D_{adv} that are misclassified by f_w as the specified target label. Formally,

$$tTR = \frac{1}{|D_{tTR}|} \sum_{\substack{(x_{adv}, y_{target}) \\ \in D_{tTR}}} \mathbb{1}[(f_b(x_{adv})) = y_{target})]. \quad (2)$$

tTR encodes a likelihood that a successful targeted example observed on the whitebox model will be a successful targeted example to the blackbox model. Current targeted attack literature commonly measures error and tSuc. We introduce uTR and tTR as new metrics for attacks that are useful if the attacker wants to maximize his chance of success in a limited number of attempts. Also note, even though the attacks are targeted, we also measure untargeted statistics as they are still relevant to the power of the attacks.

4. Activation Attack Methodology

The Activation Attack (AA) is a blackbox targeted attack designed for transferability. We can think of the Activation Attack methodology as being split into two segments, which mimic that of many deep learning attack practices. First, we specify a loss function to optimize for. Then, we establish the attack algorithm and perturbation method to adjust the data based on the loss.

4.1. Loss Function

The AA loss function is defined as the Euclidean distance between the vectorized source image activations and vectorized target image activations at some layer L. Let f_L be a truncated version of f_w (the whitebox model) that takes an image as input and outputs the activations at layer L. So $A_s^L = f_L(I_s)$ are the source image (I_s) activations and $A_t^L = f_L(I_t)$ are the target image (I_t) activations at L. The loss function J_{AA} between two images is then,

$$J_{AA}(I_t, I_s) = \|f_L(I_t) - f_L(I_s)\|_2 = \left\|A_t^L - A_s^L\right\|_2. \quad (3)$$

The intuition behind the AA loss function is to make the source image closer to an image of the target class *in feature space*. The implications/assumptions of this loss are three-fold. First, *adjustments of deep feature space representations have a sizable impact on the classification result*. Since we are not explicitly optimizing on classification loss, we rely on a byproduct of feature space perturbations being significant classification disruption. Because the feature space representation is so large and uninterpretable, it is not immediately obvious that this will be the case. Also, due to the size (i.e. number of parameters) of the feature space, we must assume that constrained image domain perturbations will be able to move the original sample to be close enough to the target sample (in feature space), as to be within regions of the target class.

The second major assumption made with this loss function is: *since intermediate layer features of deep models have shown to be transferable [26], explicit attacks in feature space will yield transferable adversarial examples*. Because modern deep models are unintelligible, there is no way of measuring exactly what features are captured and learned in each layer of a model. Thus, there is no way of definitively knowing if two models have learned similar feature sets, especially in deep and highly abstract layers. This attack principally assumes that deep layers of different deep models *have* learned similar features, and thus perturbing these highly abstract features in one model will perturb the same features in the other model. This is a reasonable assumption because in transferability attacks, we train the whitebox model on data from the same distribution as the blackbox model's training data. Thus, we expect that the models have learned similar hierarchical feature sets in order to properly model the classes of data. Since each model architecture differs in complexity, number of layers, and overall architecture, finding the layers at which features are most transferable is left to experimentation.

The third major assumption is in regards to the learned decision boundaries, and class orientations in feature space.

Specifically, we assume that *two different models trained with data from the same distribution learn similar decision boundaries and class orientations*. This is particularly relevant for the targeted attacks because in order for a transferred targeted example to be successful, the regions of that target class in feature space must have the same orientation w.r.t. the source image. In other words, if we start with the source example's feature space representation in the whitebox model and move it in the direction of the target sample, this assumption states that the direction of movement is the same (or at least similar) in the blackbox model.

4.2. Attack Algorithm

The perturbation mechanism is similar to that of the L_∞ constrained iterative gradient sign attack with momentum (TMIFGSM)[4]. Our attack algorithm iteratively perturbs the source image using the sign of the momentum term, where momentum is calculated as the weighted accumulation of gradients. The difference here is that gradients are not computed against classification loss. Rather, they are computed against (3). Also, gradients flow backward starting from layer L. Thus, momentum is calculated as

$$m_{k+1} = m_k + \frac{\nabla_{I_k} J_{AA}(I_t, I_k)}{||\nabla_{I_k} J_{AA}(I_t, I_k)||_1}, \tag{4}$$

where $m_0 = \mathbf{0}$ and I_k is the perturbed source image at iteration k. Note, $I_0 = I_s$. The perturbation method for this targeted L_∞ constrained Activation Attack is

$$I_{k+1} = Clip(I_k - \alpha * sign(m_{k+1}), 0, 1). \tag{5}$$

Notice, the perturbed image is always clipped to range $[0, 1]$, as to maintain the distribution of the original image. The intrinsic meaning of (5) is that we are slightly adjusting each pixel of the image in the direction that will minimize our J_{AA} loss. The momentum term is used to intuitively denoise or smooth the gradient directions, and is described in [4] as an accumulation of a velocity vector in the gradient direction. Also, keep in mind, we are perturbing the image with the explicit intentions of altering the feature space representation at some layer L. Thus any effects on classification are implicit, as we are not specifically accounting for classification loss.

There are also some hyperparameters to set for the algorithm. Since this is an iterative algorithm, we must choose the number of iterations K to perturb for, the total perturbation amount ϵ, and the per-iteration perturbation amount α. In all tests, we set $K = 10$, vary the ϵ, and set $\alpha = \epsilon/K$.

5. Experimental Setup

As observed in [25], examples tend to transfer between models that achieve low error for a source task. Thus, we choose CIFAR-10 [9] as our primary testing dataset, as it is non-trivial, but state of the art models can achieve less than 10% test error [12]. For the primary experiment, we select and train three canonical CNN model architectures of different design complexities that are capable of achieving low error on CIFAR-10. We use ResNet-50 [6] which achieves 6.62% top-1 test error, DenseNet-121 [7] which achieves 4.72% error, and VGG19bn [22] which tests at 6.48% error. All models were trained in PyTorch [19] using code from [12]. For completeness, we also extend some experiments to ImageNet [3] trained models. We use pretrained DenseNet-121 and ResNet-50 from PyTorchs Torchvision Models. These models incur significantly higher error for the source task, where DenseNet-121 has 25.35% top-1 error and ResNet-50 has 23.85% error.

For CIFAR-10 tests we measure the four primary metrics over the full 10k test set. First, we use DenseNet-121 as a whitebox model and evaluate transferability to VGG19bn and ResNet-50 blackbox models, separately. Next, we use VGG19bn as a whitebox model and evaluate transferability to DenseNet-121 and ResNet-50 blackbox models, separately. This allows us to see trends across whitebox models and blackbox models together. For ILSVRC2012 tests, we run one primary experiment on a 15k randomly sampled subset of the full 50k test set. Here, we test a DenseNet-121 whitebox model and a ResNet-50 blackbox model.

A note about the setup is in regards to the selection of the target image in the AA. For each dataset, we keep a library of examples from each class which have all been randomly sampled from the test splits. In the case of CIFAR-10, we keep 100 examples of each class and for ImageNet, 20 examples of each class. For a given source image, we randomly select a target class, then choose the target image from the library as the one with the *furthest* layer L activations (as measured by Euclidean distance) from the source image activations. Also, note that layer depth in the following experiments is relative, i.e. layer 2 refers to a layer closer to the input than layer 10. Also, in all tests the deepest layer tested is the final FC layer which produces the output class logits, and the sampled layers are evenly distributed across the model. A table decoding the layers for each model is in the Supplemental Materials.

6. Experimental Results

To gain an understanding of whether or not the AA attack is feasible, we will first do testing and gather empirical results, then analyze our findings. There are two major axis of parameters to test along: epsilon and depth. Epsilon tests treat attacks from each layer separately and measure the impacts of epsilon on transferability. Testing along the depth axis involves fixing the strength of the attack and measuring attack performance as a function of which layer we generate AA examples from. For these tests we use three baselines. The iterative targeted class method (ITCM) [10] is

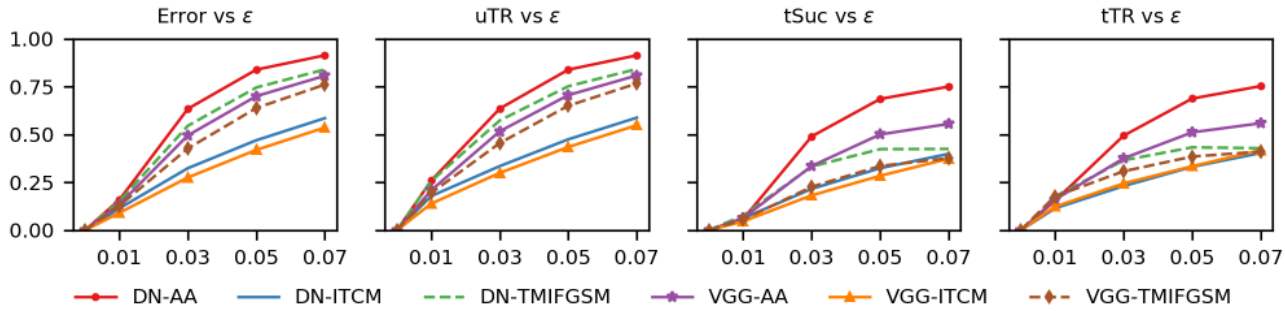

Figure 2: Transferability versus epsilon results for CIFAR-10 attacks transferring from different whitebox models to a ResNet-50 blackbox model. DN and VGG represent DenseNet121 and VGG19bn whitebox models.

a targeted variant of the basic iterative method and represents a simple method. The targeted projected gradient descent (TPGD) attack with random start from [14] represents a more complex method which routinely performs between the other baselines. And the targeted momentum iterative fast gradient sign method (TMIFGSM) [4], which won the 2017 NIPS Attacks and Defences competition [11] for both targeted and untargeted attacks, represents our most complex method. The performance of each attack on the whitebox model is shown in the supplementals.

6.1. Epsilon Results

The epsilon tests are a result of two experiments with a common goal, to fool a ResNet-50 classifier. First, we measure the four transferability metrics when transferring examples from DenseNet-121 to ResNet-50 (DN121 $\rightarrow$ RN50), then run a VGG19bn $\rightarrow$ RN50 test. For both, ϵ is swept as $[0.0, 0.01, 0.03, 0.05, 0.07]$, where $\epsilon = 0$ indicates no attack. The results are shown in Fig. 2. The represented AA attacks are from the best layers.

The first trend we see is that for all attacks across all metrics, as ϵ increases the attack strength increases. We also see the DenseNet-121 AA (DN-AA) is the most powerful attack, while the ITCM attacks are the least effective, and the TMIFGSM's fall in-between. At $\epsilon = 0.07$, DN-AA achieves random accuracy on the blackbox model at 91.42% error with an improvement of 7.4% over the best baseline. It outperforms the DN-TMIFGSM baseline in terms of uTR, tSuc, and tTR by $7.2\%, 32.6\%$, and 32.5%, respectively. Also, both DN-AA *and* VGG-AA are higher than all baselines in terms of tSuc and tTR, indicating that both activation based attacks are better blackbox targeted attacks than the baselines. The fact that DN-AA bests VGG-AA indicates that whitebox model architecture does affect AA performance. However, it is not intuitively surprising that the more complex DN121 model is more transferable to the RN50 blackbox model because both models are quite deep in comparison to the relatively shallow VGG.

6.2. Depth Results

The most important results for our hypothesis are the depth results. Here, we fix $\epsilon = 0.07$ as this is where the most powerful attack was able to achieve random accuracy on the blackbox model, although our conclusions hold for all tested epsilons. We then test the AA at different depths of the models, running a full test step at each. Fig. 3 shows the results of the depth sweep tests and Table 1 shows the numerical results of the most powerful layer AA attack versus baselines. In Table 1, DN and VGG are CIFAR-10 trained whitebox models, and DN_{IN} is ImageNet trained.

The first two rows of Fig. 3 are transfers from a DN121 whitebox model, and the bottom two are from the VGG19bn whitebox model. We see that *layer-wise transferability characteristics are not dependent on blackbox model*. Meaning, the shape of the trendlines do not change with the blackbox model. This is crucial to the feasibility of the attack because it means the attacker may use their own blackbox model to find the best transferring layer, then use that knowledge to attack the true target blackbox model. This limits the amount of queries to the target model. Further, both DN121 whitebox tests show powerful transfer-

Table 1: Numerical Transfer Results (RN50 Blackbox)

Base	Attack	Error	uTR	tSuc	tTR
DN	ITCM	58.62	58.88	39.97	40.39
	TPGD	61.33	61.52	35.84	36.15
	TMIFGSM	84.12	84.32	42.53	42.90
	$AA_{L=21}$	91.49	91.48	75.13	75.38
VGG	ITCM	53.72	54.94	37.40	41.88
	TPGD	58.85	59.55	36.60	39.15
	TMIFGSM	76.19	76.75	37.72	41.14
	$AA_{L=6}$	80.81	80.97	55.64	55.98
DN_{IN}	ITCM	21.14	21.18	0.99	1.01
	TPGD	25.23	25.29	0.94	0.97
	TMIFGSM	47.75	47.76	2.25	2.25
	$AA_{L=7}$	80.58	81.77	2.57	8.63

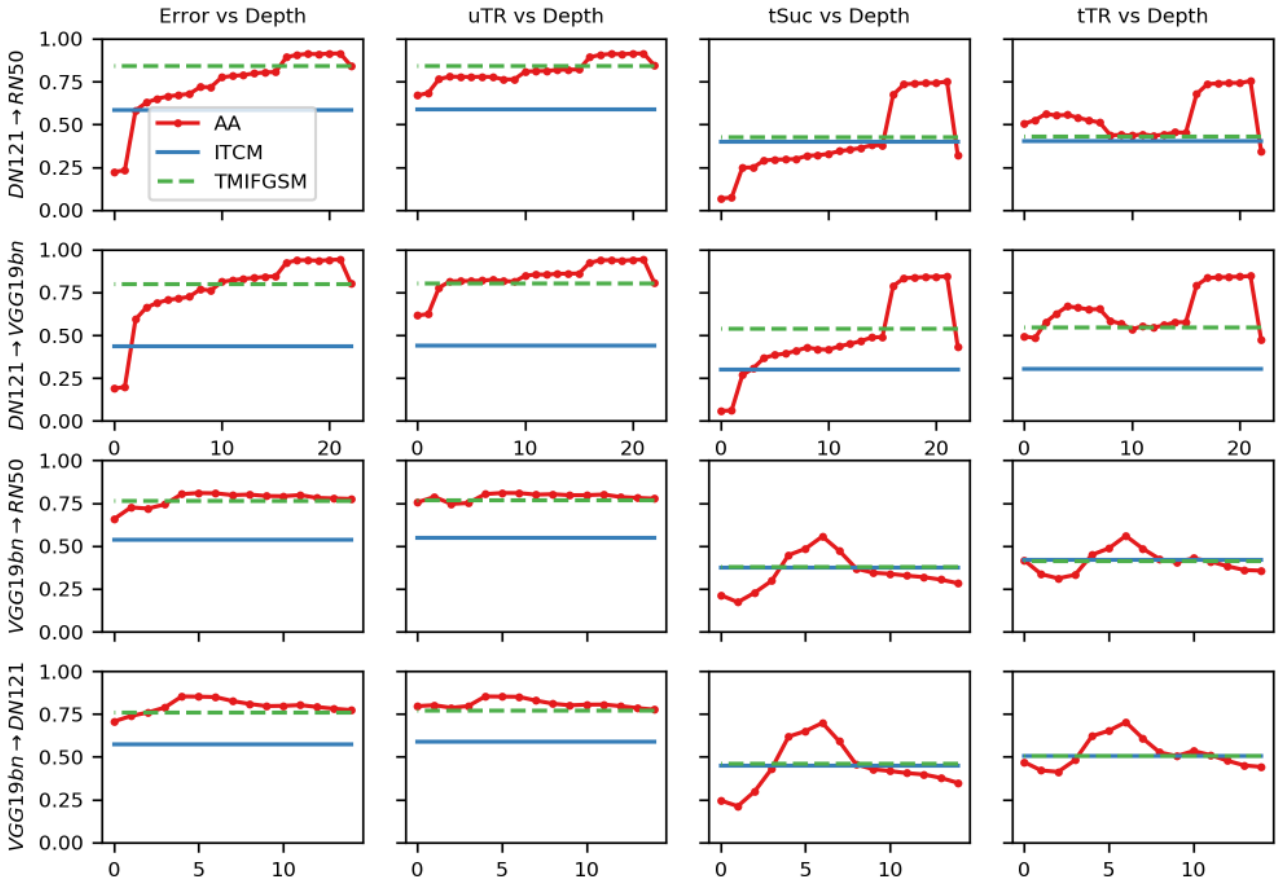

Figure 3: Error, uTR, tSuc, and tTR rates versus depth for multiple transfer scenarios. The top two rows are transfers from a DN121 whitebox model and the bottom two rows are transfers from a VGG19bn whitebox model. Dataset: CIFAR-10.

ability in the deeper layers, and both VGG19bn tests show strong transferability in the middle layers. We also see that AA's from some layers yield less powerful attacks, indicating the choice of AA layer is critical.

If we choose the single best layer to attack from, we can compare numerical performance to the best baseline, which in all cases is TMIFGSM (see Table 1). For the DN121 model the best AA is from $L = 21$, and for VGG19bn the best AA is from $L = 6$. When transferring to RN50, the $DN121_{L=21}$ attack outperforms the best baseline by 7.4% in error, 7.2% in uTR, 32.6% in tSuc, and 32.5% in tTR. Similarly, when transferring to RN50 from VGG19bn, the $VGG19_{L=6}$ attack outperforms the best baseline by $4.6\%, 4.2\%, 17.9\%, 14.8\%$ in error, uTR, tSuc, and tTR, respectively. Notice, for DN121 whitebox, error rate and untargeted transfer rate are both about 91.5% and targeted success and transfer rate are both about 75.2%. Such high numbers indicate that these deep and complex models have learned similar feature sets and similar decision boundary structures. Also, adversarial directions for one model are adversarial in other models. The targeted transfer metrics further indicate that the orientation of the decision boundaries in feature space and the directions to move from one class to another in feature space are similar across models.

Finally, we can compare performance of AA attacks directly. When transferring to RN50, $DN121_{L=21}$ outperforms $VGG19_{L=6}$ by $10.6\%, 10.5\%, 19.5\%, 19.4\%$ in error, uTR, tSuc, and tTR, respectively. Thus, we again find that the choice of whitebox model to transfer from is important.

6.3. Analysis

The next natural question to ask is why are some layers better than others, and in particular, why $DN121_{L=21}$ and $VGG19_{L=6}$? Given that the transferability trends in Fig. 3 do not change with the blackbox model, to answer this question we only consider characteristics of the whitebox model and the perturbed data. The first experiment is to measure the separability of the class representations in feature space. Intuitively, we expect that an AA with good transferability characteristics would come from a layer with well separated class representations. Fig. 4 shows the average angular distance between examples of the same class (intra-class) and examples of different classes (inter-class),

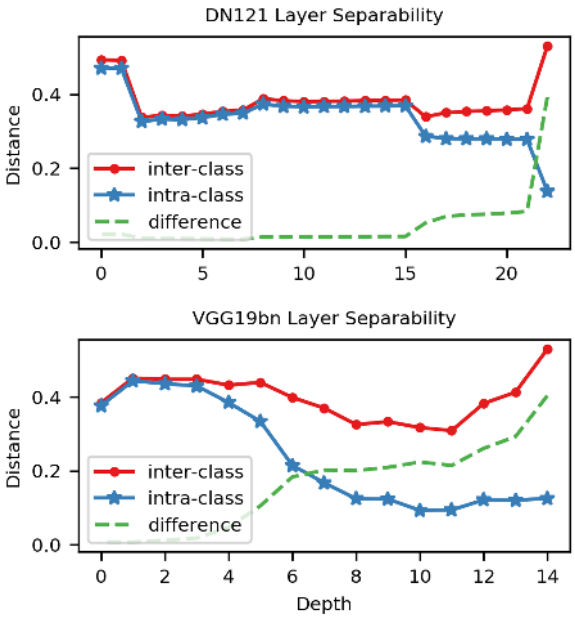

Figure 4: Average angular distance between feature maps of examples from the same class (intra-class) and examples of different classes (inter-class) for DN121 and VGG19bn CIFAR-10 trained models.

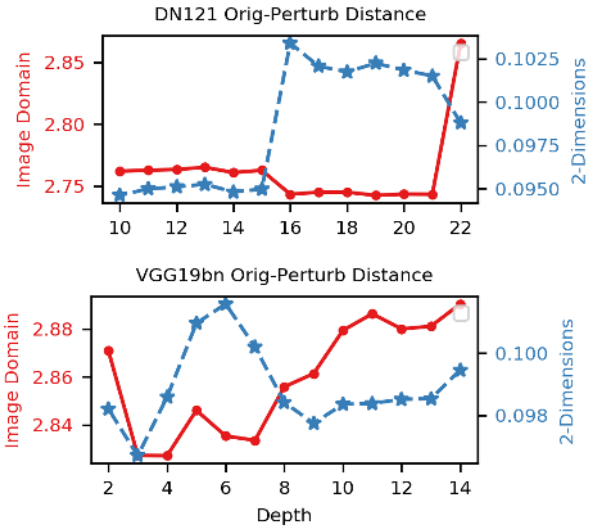

Figure 5: L_2 distance between original and AA perturbed images as measured in the image domain (red) and when projected onto the first two principal component directions of the clean data (blue). Examples generated from DN121 and VGG19bn CIFAR-10 trained models.

across layer depths. For well separated layers, we expect the inter-class distance to be high and the intra-class distance to be low, while the difference between the two is large. Fig. 4 shows that classes become well separated in later layers for DN121, and earlier in VGG19bn. Thus, the best layers to transfer from ($DN121_{L=21}$ and $VGG19_{L=6}$) have well separated class representations. However, the layer with the largest difference is not necessarily the most transferable.

Another experiment is to look purely at the data and measure the average distance between original and perturbed examples from each layer. We measure Euclidean distance in the image domain and in two dimensions. Here, we project to the first two principal component directions of the clean data to measure the attack's effects along the directions of greatest variance. Instinctively, we may expect that layers with better transferability characteristics would produce adversarial examples that are farther from the original data, as they may be more likely to have crossed a decision boundary. Fig. 5 shows the results of these experiments on the DN121 and VGG19 whitebox models for layers with some amount of feature separability. For DN121, layers 16 through 21 produce perturbed examples that are further in two dimensions from the original data, but closer in the image domain. Even though $L = 21$ doesnt produce the furthest examples, the trend in the two dimensional measurements is similar to the DN121 transferability trends from Fig. 3 where layers 16-21 have the best transferability characteristics. Similarly, for the VGG19 test, $L = 6$ produces examples that are furthest from the originals in two dimensions, while in the image domain these perturbed examples tend to be closer to the original data. The VGG19 trend in two dimensions also mimics Fig. 3 trends.

From this analysis, we observe a few tendencies that are indicative of layers that produce transferable adversarial examples. First, these layers have well separated class representations in feature space. And second, these layers produce examples that are closer to the original data in the image domain, but further in the first two principal component directions. Unfortunately, none of these results have been absolutely conclusive. Thus, we leave it to future work to more thoroughly explore the reasons for layer-wise transferability characteristics. However, with these analysis techniques, we may now make informed predictions about what layers will be most transferable when using different data and models, avoiding the expensive step of sweeping the layer. See the Supplemental Materials for analysis-first experiments on SVHN [17] trained models.

6.4. ImageNet Results

We now extend the experiments to ImageNet trained classifiers. Here, we measure the transferability of a DN121 $\rightarrow$ RN50 attack and do a similar analysis of results. Fig. 6 shows transferability results for a fixed $\epsilon = 0.07$ attack, and DN_{IN} rows of Table 1 show numerical results. Interestingly, we see several different trends here. First, early and middle layer DN121 AAs are more transferable, unlike the CIFAR-10 tests. Next, the targeted transfer

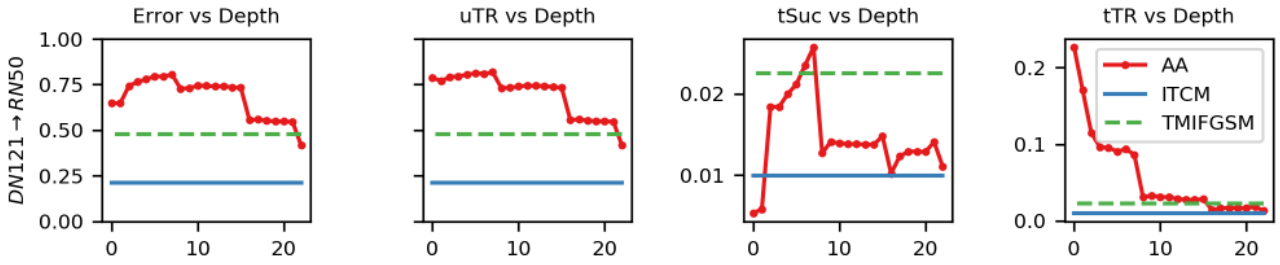

Figure 6: Error, uTR, tSuc, and tTR rates versus depth for ImageNet trained DN121 → RN50 transfer scenario.

statistics, tSuc and tTR, are lower for all attacks. Consider $DN121_{L=7}$ which is arguably the best AA layer. As compared to the best baseline, the $DN121_{L=7}$ AA has $32.8\%, 34.0\%, 0.3\%, 6.3\%$ better error, uTR, tSuc, and tTR, respectively. The drop in tSuc and tTR from the CIFAR-10 tests speaks to the dissimilarity in decision boundaries of these less accurate models.

The AA achieves a large margin over the baselines in terms of fooling rate and likelihood of misclassification (i.e. uTR). This gives an indication that the direction from an example to a nearby decision boundary is similar in some layers. However, the overall layout and orientation of classes in feature space is not the same between models. This highlights a fundamental weakness of single-model transfer-based targeted attacks. If the models are not well trained for the source task, the decision boundaries in feature space are not highly similar, so it is difficult to find a direction to move toward target class regions. An interesting future work would be to generate AA examples from an ensemble, which may find a mean direction to move.

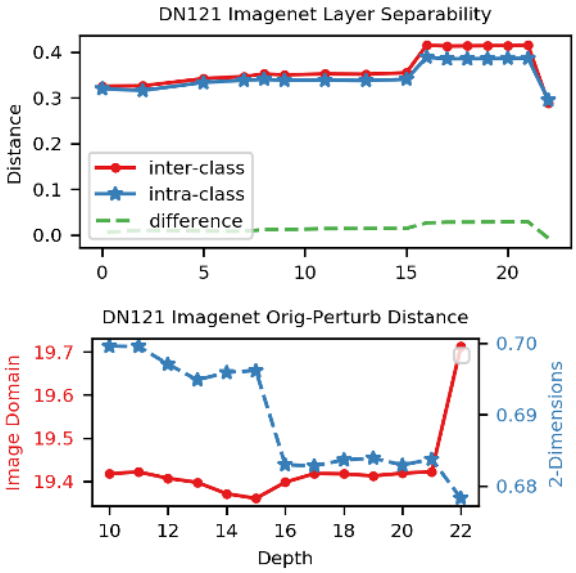

Figure 7: Analysis of ImageNet trained DN121 layer-wise feature similarity (top) and distance between original and adversarial examples generated with Activation Attack from this whitebox model (bottom).

Fig. 7 (top) shows the analysis of layer separability and distance between original (clean) and perturbed examples. Not surprisingly from the tSuc and tTR results, none of the layers in the DN121 ImageNet model appear to have well separated classes in feature space. Thus, we would expect that driving our source features towards features of a single target example does not necessarily mean we are driving towards large regions of that target class in feature space. A potential future work is to drive towards a centroid of target class examples, which may inflate the targeted performance and boost AA's tSuc rate.

We also see a changing trend in Fig. 7 (bottom) when we look at layers with some amount of separability. Rather than layers 16 through 21 producing examples that are further in two dimensions, earlier layers (10 through 15) generate further examples. This is similar to the Fig. 6 findings, where earlier layers transfer better than later ones. This is further evidence that more transferable AA layers produce perturbed examples that are further from the originals along the principal component directions, yet closer to the originals in the image domain.

7. Conclusion

This work describes an adversarial attack using feature space perturbations that also provides insights into how deep learning models make decisions. We show for well trained models, feature space perturbations are highly transferable and the layer at which perturbations are transferred has a large impact on attack effectiveness. Also, the layer-wise transferability characteristics of a whitebox model are blackbox model agnostic. Through analysis we find layers that are best to attack from have well separated class representations and produce examples that are perturbed more along the principal component directions. Towards interpretability, we indicate deep CNNs of differing architectures learn similar hierarchical representations of the data by showing that perturbing features of one model also perturbs those features of other models.

Acknowledgements We acknowledge the support of AFRL (FA8750-18-2-0057).

References

[1] N. Carlini and D. A. Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[2] N. Carlini and D. A. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7, 2018.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[4] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li. Boosting adversarial attacks with momentum. 2017.

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[7] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[8] J. Kos, I. Fischer, and D. X. Song. Adversarial examples for generative models. *2018 IEEE Security and Privacy Workshops (SPW)*, pages 36–42, 2018.

[9] A. Krizhevsky, V. Nair, and G. Hinton. Cifar-10 (canadian institute for advanced research).

[10] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *CoRR*, abs/1611.01236, 2016.

[11] A. Kurakin, I. J. Goodfellow, S. Bengio, Y. Dong, F. Liao, M. Liang, T. Pang, J. Zhu, X. Hu, C. Xie, J. Wang, Z. Zhang, Z. Ren, A. L. Yuille, S. Huang, Y. Zhao, Y. Zhao, Z. Han, J. Long, Y. Berdibekov, T. Akiba, S. Tokui, and M. Abe. Adversarial attacks and defences competition. *CoRR*, abs/1804.00097, 2018.

[12] K. Liu. pytorch-cifar. `https://github.com/kuangliu/pytorch-cifar`, 2017.

[13] Y. Liu, X. Chen, C. Liu, and D. X. Song. Delving into transferable adversarial examples and black-box attacks. *CoRR*, abs/1611.02770, 2016.

[14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *CoRR*, abs/1706.06083, 2017.

[15] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 86–94, 2017.

[16] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, 2016.

[17] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[18] N. Papernot, P. D. McDaniel, and I. J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[20] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary. Robust deep reinforcement learning with adversarial attacks. In *AAMAS*, 2018.

[21] S. Sabour, Y. Cao, F. Faghri, and D. J. Fleet. Adversarial manipulation of deep representations. *CoRR*, abs/1511.05122, 2015.

[22] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[23] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

[24] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. D. McDaniel. Ensemble adversarial training: Attacks and defenses. *CoRR*, abs/1705.07204, 2017.

[25] F. Tramèr, N. Papernot, I. J. Goodfellow, D. Boneh, and P. D. McDaniel. The space of transferable adversarial examples. *CoRR*, abs/1704.03453, 2017.

[26] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014.