Supplemental Material: 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

1. Layer Decoder Table

Table 1: Whitebox Model Layer Decoding Table

Layer	DenseNet-121	VGG19bn
0	4	128
1	6	128
2	6,2	256
3	6,4	256
4	6,6	256
5	6,8	256
6	6,10	512
7	6,12	512
8	6,12,2	512
9	6,12,4	512
10	6,12,14	512
11	6,12,16	512
12	6,12,18	512
13	6,12,20	512
14	6,12,22	FC
15	6,12,24	-
16	6,12,24,2	-
17	6,12,24,8	-
18	6,12,24,10	-
19	6,12,24,12	-
20	6,12,24,14	-
21	6,12,24,16	-
22	6,12,24,16,FC	-

Table 1 is the layer number look-up-table that corresponds to the layer notation used in the paper. DenseNet-121 (DN121) and VGG19bn (VGG) appear because they are the whitebox model architectures used for the main results. The DN121 notation follows the implementation here: https://github.com/kuangliu/pytorch-cifar/blob/master/models/densenet.py [5]. In english, layer 0 shows that the output of the truncated model comes from the 4^{th} denseblock of the first denselayer. Layer 15 means the output of the truncated model comes from the 24^{th} denseblock in the 3^{rd} denselayer. Layer 22 indicates the output comes from the final FC layer of the model.

The VGG model does not have denseblocks or dense layers so we use another notation. In the implementation at https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py, the VGG19bn model is constructed from the layer array: $[64, 64, M, 128, 128, M, 256, 256, 256, 256, M, 512, 512, 512, 512, M, 512, 512, 512, 512, M]$, and we follow this convention in the table. In the array, each number corresponds to a convolutional layer with that number of filters and the M's represent max-pooling layers. Notice, in these tests we do not consider the first two layers of the model as they were shown to have very little impact on classification when perturbed.

2. Whitebox Attack Results

Table 2: Whitebox Attack Results

Base	Attack	Error	tSuc
DN	ITCM	99.42	98.86
	TPGD	99.60	99.14
	TMIFGSM	99.48	99.05
	$AA_{L=21}$	100	99.76
VGG	ITCM	97.45	89.27
	TPGD	98.28	93.46
	TMIFGSM	98.33	91.41
	$AA_{L=6}$	99.73	99.31
DN_{IN}	ITCM	99.68	97.44
	TPGD	99.54	96.57
	TMIFGSM	100	99.99
	$AA_{L=7}$	98.1	20.08

Table 2 shows the numerical attack results on the whitebox models that were used to generate the adversarial examples to be transferred. Notice, we only measure error and

tSuc here, as the other metrics (uTR and tTR) are blackbox only metrics and can only be measured on transferred examples. In this setting, error is the percentage of adversarial examples generated that fool the model, and tSuc is the percentage of adversarial examples that are misclassified by the model as the specified target class. The table shows that each attack is very effective as a whitebox attack. Here, DN and VGG represent the CIFAR-10 trained DenseNet-121 and VGG19bn models, and DN_{IN} represents the ImageNet trained DenseNet-121 model. Recall, ITCM [6], TPGD [7], and TMIFGSM [2] are all baselines. The AA's are each from the best performing layers, as observed in the primary depth experiment.

Notice, all of the attacks for all models completely degrade the performance in terms of error, driving the classification accuracy well below random. Our AA even achieves 100% error on the DN whitebox model. We acknowledge that this is not a surprise, as $\epsilon = 0.07$ is higher than necessary for a whitebox attack to achieve random accuracy on CIFAR-10. We also see that the targeted success rates for all tests but one are very high, indicating that almost all attacks can reliably generate targeted adversarial examples in the whitebox setting. Interestingly, our AA under-performs the others on the tSuc ImageNet test, achieving only 20% tSuc. This may be because the attack drives towards a particular example (i.e. single example) of the target class and assumes that if it gets close, it will be in a region of feature space that will be classified as the target class. However in ImageNet, the feature space is much larger and there are many more classes and decision boundaries. So, given the number of perturbing iterations and epsilon are fixed, in our case getting "close" to the target example is not good enough to cause targeted misclassification.

3. Interpretation of Attack Figure

It is important to fully understand the attack visualization Fig. 1 to comprehend what the Activation Attack is doing. We start with a source image (i.e. dog) and a target image (i.e. plane) both of which are correctly classified by the whitebox model (f_w), and the source image is also correctly classified by the blackbox model (f_b). In the whitebox model, the layer L activations for the dog are orange and dark blue for the plane. Although we cannot directly observe the activations in f_b, there is some layer that has learned similar dog features and is hence vulnerable to transferring perturbations. The AA then drives the dog's layer L features (orange) towards the planes's features (dark blue) by perturbing the input image on the whitebox model. The resulting features for the perturbed image (light blue) are not exactly the same as the plane, but are similar enough as to cause misclassification to the plane class. These perturbed features then transfer to f_b and cause misclassification to the plane class in f_b.

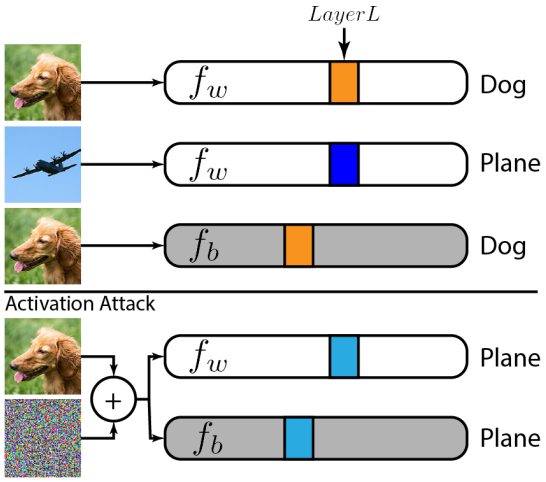

Figure 1: Illustration of Activation Attack. Given that the whitebox model (f_w) and blackbox model (f_b) are initially correct, the attack drives the layer L activations of the dog image towards the layer L activations of the plane. After attack, the dog's activations are similar to the plane's and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

4. Visual Intuitions in Feature Space

This section will show some visualizations of feature space not shown in the main manuscript. It is meant to solidify intuition and verify some findings in the paper regarding feature space. Fig. 2 shows an example of a source, adversarial, and target image triplet in both the image domain and in (flattened) feature space. In the image domain, the adversarial image maintains the spatial features of the source image. In feature space, the adversarial example has the dominant features of the target image. This figure further verifies the functionality of the AA loss function which explicitly drives the source image features towards the target image features.

Fig. 3 shows a t-SNE [11] visualization of CIFAR-10 features for several layers of VGG19bn and DenseNet121 models. This figure is meant to supplement Fig. 4 of the main manuscript. Namely, Fig. 4 of the manuscript alludes to the presence of class-wise clusters in feature space. This visualization confirms that in layers with high separability, clusters do exist.

5. SVHN Analysis First Tests

In the main experiment, to understand if feature space perturbations *could* produce transferable examples we performed an expensive layer sweep to find the best layer. We also found that transferability characteristics are blackbox model agnostic. Thus, a motivated attacker could setup a sandbox environment with a whitebox and blackbox model and perform a layer sweep to find the best transferring layer.

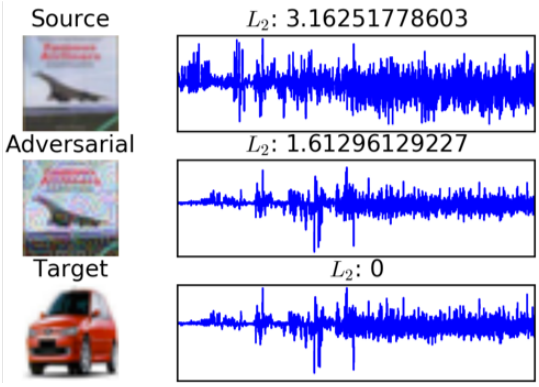

Figure 2: Source, Adversarial, and Target image triplet in the image domain and in feature space. The features of the adversarial image clearly resemble the target image features.

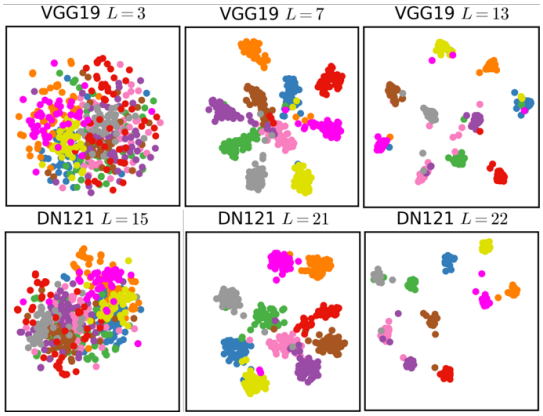

Figure 3: t-SNE plot of features from 50 examples of each class taken from CIFAR-10 trained VGG19 and DN121 models at several layers (L). Colors represent classes. Visualization meant to supplement Fig. 4 of main manuscript.

The found layer would then presumably be the best for other blackbox models. However, we also analyzed the findings and found that layers with well separated class representations that perturb examples further in two dimensions but closer in the image domain tend to produce more transferable examples. In an effort to supplement the findings of the analysis, and avoid the expensive full layer sweep of transferring to a blackbox model, we include a new experiment on the SVHN [8] dataset. Here, we perform the analysis of the whitebox model and perturbed data *first* to find candidates for well transferring layers. Then, we evaluate those layers with AA to show that the analysis techniques are sufficient for finding well transferring layers.

In this test we include additional models to further test the idea that well transferring layers do not depend on blackbox model. We use DenseNet-121 (DN121) [4], VGG19bn [10], ResNet-50 (RN50) and ResNet-152 (RN152) [3], MobileNetv2 (MNv2) [9], and a Dual Path Network (DPN92) [1]. Each model is trained on the SVHN training split for 350 epochs. The test accuracy of each model is shown in Table 3, where we see that each model is well trained for the source task.

Table 3: SVHN Model Accuracy Table

Model	SVHN Test Accuracy
DN121	96.67
VGG19bn	96.43
RN50	96.47
RN152	96.99
MNv2	96.22
DPN92	96.99

For these tests we again use DN121 and VGG19bn whitebox models. However, since we are using new data, we must reanalyze the models to find the best transferring layers. First, consider DN121 as a whitebox model. Fig. 4 shows the analysis of this SVHN trained model. Fig. 4 (top) shows the average intra-class and inter-class angular distance between features at each layer as measured over 100 examples of each class. Recall, this indicates the separability of class specific features *in feature space*. Here, we see class-specific features only become well separated at the end of the model (layers 16-21). Thus, we measure the Euclidean distance between the original and perturbed AA examples in the image domain and in two dimensions, for that part of the network as shown in Fig. 4 (bottom). We identify layers 16-20 as having favorable conditions for transferability, as the perturbed data from these layers is further in two dimensions but closer in the image domain. Also, these layers have particularly well separated features. We choose Layer 17 for these tests because it has a slight local maxima in the 2D distance measurements. With this information we can now make the informed decision to use layer 17 in the AA attack, although it appears layers 16-20 would all transfer well. The first five "Transfer Scenario" rows of Table 4 show the attack results from a DN121 whitebox. We see the $AA_{L=17}$ attack outperforms all baselines on all blackbox models tested. On average across all blackbox models, $AA_{L=17}$ performs better than the best baseline (TMIFGSM) by 9.16% error, 7.4% uTR, 10.33% tSuc, 8.38% tTR. Also on average, $AA_{L=17}$ performs better than the least powerful baseline (ITCM) by 21.58% error, 20.98% uTR, 18.08% tSuc, 18.25% tTR.

Similarly, we now analyze the VGG19bn model to find potentially well transferring layers. Fig. 5 (top) shows the feature separability of each layer and (bottom) shows the distance between original and perturbed examples. Features start to become well separated around layer 5 and remain well separated until the final (FC) layer. If we measure the

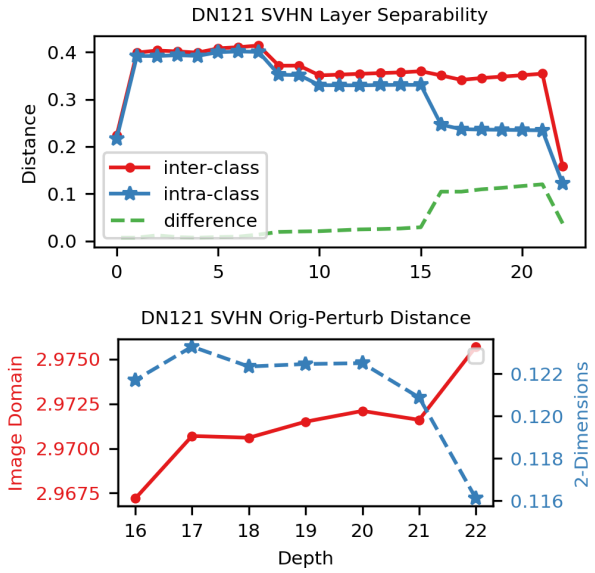

Figure 4: Analysis of SVHN trained DN121 layer-wise feature similarity (top) and distance between original and adversarial examples generated with Activation Attack from this whitebox model (bottom).

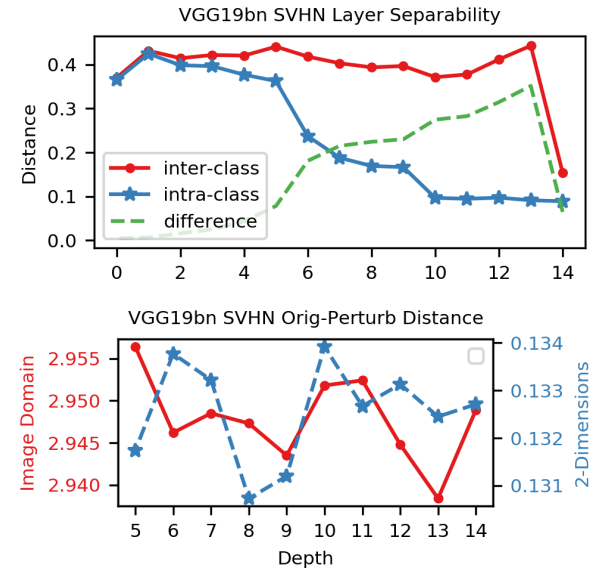

Figure 5: Analysis of SVHN trained VGG19bn layer-wise feature similarity (top) and distance between original and adversarial examples generated with Activation Attack from this whitebox model (bottom).

original/perturbed distance (Fig. 5 (bottom)) at these well separated layers we identify several candidate layers: $L = 6$ which has a local max in 2D distance and a local min in image domain distance, $L = 10$ which has high 2D perturbation but also high image domain distance, and $L = 13$ which has relatively high 2D distance and low image domain distance. Here, we select $L = 6$ because of its local minimum and maximum properties, but a diligent attacker may try several layers in their own whitebox/blackbox sandbox environment. We then run the VGG19 whitebox attacks to all blackbox models and find that $AA_{L=6}$ outperforms all other attacks in all cases. On average across all blackbox models, $AA_{L=6}$ performs better than TMIFGSM by 3.93% error, 3.84% uTR, 13.4% tSuc, 8.89% tTR. Also on average, $AA_{L=6}$ performs better than ITCM by 13.34% error, 12.35% uTR, 10.47% tSuc, 6.82% tTR.

Note, we have not done a full layer sweep of attacks to the blackbox model so we can not definitively say if the layers chosen from the analysis are the *best*. As described, there is no hard rule for choosing the best layer through the analysis, rather we analyze for trends in layer separability and original/perturbed image distances. Future work is to further refine the analysis to conclusively identify the best layer. However, also notice that all of the layers we identified were better than baselines, giving further merit to our methodology.

5.1. MobileNetV2 Whitebox Model

Finally, we can introduce a new whitebox model for our SVHN transfer setup and perform analysis first testing of the AA method. This allows us to analyze and choose a layer to transfer from with no prior knowledge from the main experiment. In this case, we will use the MobileNetV2 model which performs competitively with the other models (Table 3). The choice of model here is somewhat arbitrary, the goal of this section is to find a well transferring layer to use AA from without using prior information and without performing expensive layer sweep testing to a blackbox model.

Fig. 6 (top) again shows the average intra-class and inter-class angular distance between features at each layer as measured over 100 examples of each class. Interestingly, we see a few layers towards the end of the network with good class-wise feature separability (i.e. layers 13-15). We then measure the original/perturbed distance of examples from layers 12-18 which are in and around this region of separability, shown in Fig. 6 (bottom). From this data, layers 13 and 14 have higher 2D distance relative to image domain distance, and these layers are both in the region of good separability. Since both $L = 13$ and $L = 14$ appear to have very similar characteristics, we measure both attacks performance in a MNv2→RN50 test, shown in Table 5. Not surprisingly from how similar the analysis was, these layers perform very similarly, so we choose $L = 13$ and run tests to the other models from Table 3. Again, the merit of the AA is confirmed. Across all models and all metrics, $AA_{L=13}$ is better than the baselines.

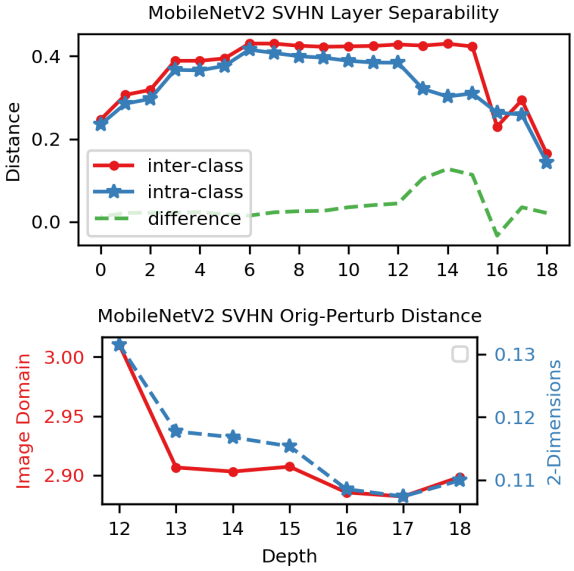

Figure 6: Analysis of SVHN trained MobileNetV2 layer-wise feature similarity (top) and distance between original and adversarial examples generated with Activation Attack from this whitebox model (bottom).

6. Sample Perturbed Images

Finally, we can visualize some perturbed CIFAR-10 images from the baseline attacks and our AA. Fig. 7 shows the baseline attacks against the $DN121_{L=21}$ and $VGG19_L = 6$ AA's on the same image being perturbed toward the same target image/class. One observation is that the AA perturbations at higher epsilons are more structured and appear to have a pattern. This is as opposed to the ITCM and TPGD examples which appear to have more random perturbations that do not form structure.

Fig. 8 shows different examples for different attacks. It also shows how AA's from different layers perturb the images. From this figure, it is not obvious exactly what features AA's from different layers are perturbing. However, it is an interesting future work to try to interpret exactly what features are being perturbed and why.

References

[1] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng. Dual path networks. In *NIPS*, 2017.

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

[3] 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.

[4] 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.

[5] kuangliu. pytorch-cifar. `https://github.com/kuangliu/pytorch-cifar`, 2017.

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

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

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

[9] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. 2018.

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

[11] L. van der Maaten and G. E. Hinton. Visualizing data using t-sne. 2008.

Table 4: SVHN Numerical Transfer Results ($\epsilon = 0.07$)

Transfer Scenario	Attack	Error	uTR	tSuc	tTR
DN121 $\rightarrow$ RN50	itcm	72.35	73.79	67.65	70.10
	tpgd	69.57	70.81	64.87	66.76
	tmifgsm	83.10	85.56	73.25	77.74
	$AA_{L=17}$	91.16	91.78	82.96	85.34
DN121 $\rightarrow$ VGG19	itcm	59.49	60.67	54.63	56.71
	tpgd	53.63	54.50	48.72	50.15
	tmifgsm	77.60	79.90	68.22	72.50
	$AA_{L=17}$	88.02	88.74	79.16	81.80
DN121 $\rightarrow$ MobileNetv2	itcm	76.82	77.61	70.01	72.53
	tpgd	75.64	76.69	69.14	71.05
	tmifgsm	84.19	86.52	72.57	77.06
	$AA_{L=17}$	92.24	92.81	82.85	85.39
DN121 $\rightarrow$ RN152	itcm	67.43	68.84	62.99	65.45
	tpgd	63.42	64.61	58.86	60.73
	tmifgsm	79.80	82.29	70.94	75.54
	$AA_{L=17}$	89.39	90.05	81.19	83.83
DN121 $\rightarrow$ DPN92	itcm	65.45	66.74	60.85	63.11
	tpgd	61.57	62.63	57.21	58.96
	tmifgsm	78.92	81.31	69.90	74.45
	$AA_{L=17}$	88.59	89.21	80.37	82.81
VGG19 $\rightarrow$ RN50	itcm	80.84	82.56	74.05	81.11
	tpgd	82.53	84.08	77.15	82.63
	tmifgsm	89.00	89.78	70.48	78.39
	$AA_{L=6}$	92.34	92.99	83.42	86.74
VGG19 $\rightarrow$ DN121	itcm	78.88	80.57	72.51	79.28
	tpgd	81.59	83.13	76.07	81.14
	tmifgsm	87.33	88.09	69.31	76.91
	$AA_{L=6}$	91.50	92.12	82.69	85.97
VGG19 $\rightarrow$ MobileNetv2	itcm	76.43	78.12	68.34	75.25
	tpgd	78.63	80.11	71.40	76.66
	tmifgsm	85.52	86.23	64.27	71.70
	$AA_{L=6}$	89.72	90.49	78.90	82.09
VGG19 $\rightarrow$ RN152	itcm	74.11	75.70	68.22	74.81
	tpgd	76.20	77.83	71.15	76.53
	tmifgsm	85.33	86.15	67.02	74.69
	$AA_{L=6}$	89.40	90.11	79.74	82.75
VGG19 $\rightarrow$ DPN92	itcm	76.21	77.87	69.68	76.63
	tpgd	78.10	79.58	72.95	78.36
	tmifgsm	86.35	87.12	67.08	74.57
	$AA_{L=6}$	90.23	90.85	80.44	83.62

Table 5: SVHN MobileNetV2 Transfer Results

Transfer Scenario	Attack	Error	uTR	tSuc	tTR
MNv2 → RN50	itcm	47.93	48.90	42.78	43.91
	tpgd	49.49	50.31	43.61	44.53
	tmifgsm	67.49	69.09	58.23	60.04
	$AA_{L=13}$	83.76	84.68	72.66	75.55
	$AA_{L=14}$	83.76	84.62	72.62	75.57
MNv2 → VGG19	itcm	35.70	36.44	30.94	31.75
	tpgd	34.76	35.32	29.25	29.84
	tmifgsm	61.04	62.58	52.10	53.83
	$AA_{L=13}$	78.28	79.19	67.85	70.49
MNv2 → DN121	itcm	52.25	53.32	47.83	49.04
	tpgd	56.92	57.83	51.86	52.92
	tmifgsm	67.62	69.27	60.23	62.17
	$AA_{L=13}$	85.00	85.84	75.11	78.06
MNv2 → RN152	itcm	41.05	41.88	36.73	37.69
	tpgd	42.37	43.09	37.43	38.26
	tmifgsm	63.81	65.48	55.79	57.65
	$AA_{L=13}$	80.74	81.62	70.13	73.00
MNv2 → DPN92	itcm	43.31	44.17	39.16	40.15
	tpgd	44.00	44.75	38.96	39.77
	tmifgsm	64.29	65.87	56.17	57.94
	$AA_{L=13}$	80.89	81.66	70.35	73.12

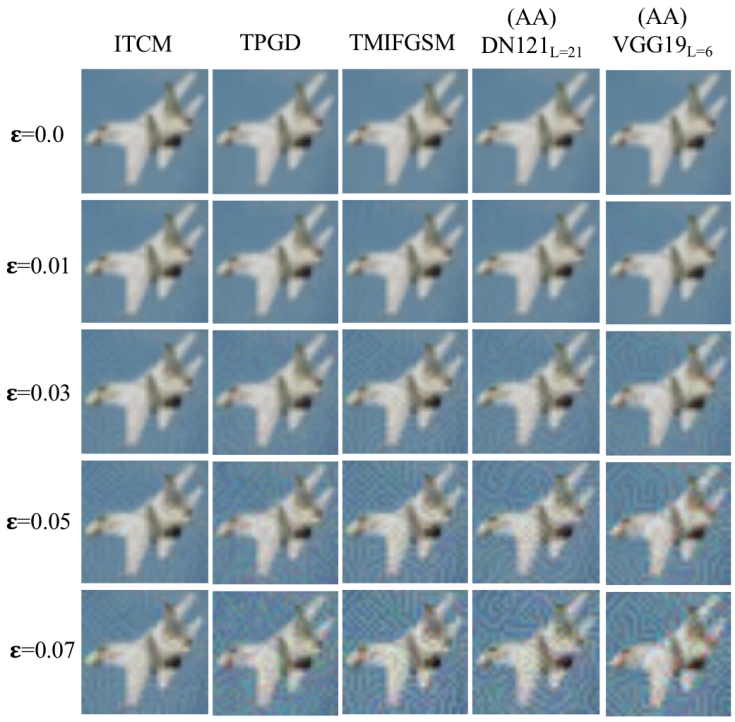

Figure 7: Perturbations of baseline attacks and most effective AA's on same source image, each being perturbed towards the same target image/class. Dataset: CIFAR-10.

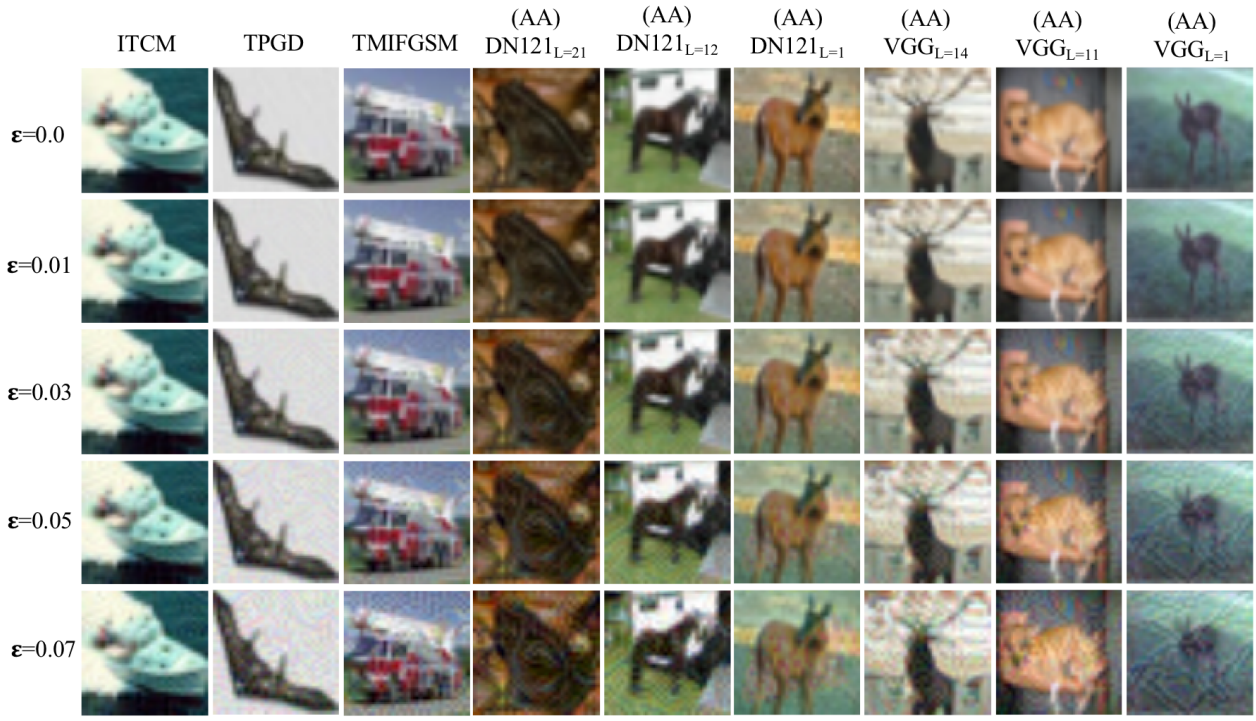

Figure 8: Perturbations of baseline attacks and AA's from different layers on different images. Dataset: CIFAR-10.