Supplementary Material: On the Robustness of Semantic Segmentation Models to Adversarial Attacks

Anurag Arnab[1] Ondrej Miksik[1,2] Philip H.S. Torr[1]
[1]University of Oxford [2]Emotech Labs
{anurag.arnab, ondrej.miksik, philip.torr}@eng.ox.ac.uk

This supplementary material details the DNN models we analysed, and experiments we omitted from the main paper since they follow similar trends. Section A1 provides further details about the experimental set-up, including the various DNNs used in the experiments. Section A2 shows qualitative examples of the adversarial attacks we studied. Section A3 presents further experimental results about "The robustness of different networks" (Sec. 5 of the main paper). Similarly, Section A4 shows more experimental results about "Multiscale Processing and Transferability of Adversarial Examples" (Sec. 6 of the main paper). Finally, Section A5 presents further experimental results on the "Effect of CRFs on Adversarial Robustness" (Sec. 7 of the main paper).

A1. Experimental setup

This section details the DNN models, additional information about the Cityscapes dataset and the software and hardware used in the experiments.

A1.1. Software and hardware setup

We use the Caffe [5] deep learning framework for all experiments, since most publicly available segmentation models are implemented using this library. Our experiments are performed on either a Nvidia M40 or P100 GPU which have 12GB and 16GB of memory respectively.

A1.2. Description of models

We detail each model in this section. Tab. A1 shows the performance of publicly available models on the Pascal VOC validation set. Tab. A2 compares the Intersection over Union (IoU) obtained by models that we have retrained compared to the original author's performance where available. Tab. A3 shows the performance of publicly available models on the Cityscapes validation set. Finally, Tab. A4 lists the number of parameters in each of the models.

FCN8s [8]. We retrained the FCN8s (VGG) network on Pascal VOC with additional annotations from SBD [3] and MS-COCO [7]. The publicly available model of FCN8s is not trained with MS-COCO, which is why we retrained it ourselves. As shown in Tab. A2, we obtain an IoU of 68.7% on the VOC validation set, whilst the original authors who did not train on MS-COCO obtained 65.5% [10].

Table A1: Networks with public models, evaluated on the VOC validation set

Model Name	IoU [%]
CRF-RNN [15]	72.8
Dilated Frontend [12]	67.1
Dilated Context [12]	70.4
SegNet [1]	43.0

For the Cityscapes dataset, we used the publicly available VGG model[1] from [11].

We trained FCN8s with a ResNet-101 backbone on Pascal VOC since no publicly available model was available. As shown in Tab. A2, the IoU on clean inputs of this version is close to the VGG version. We are not aware of any other published work to compare this number to.

Deeplab v2 [2]. We cannot use the publicly released models for the Pascal VOC dataset, since they have been trained on the entire validation set as well. Hence, we use the authors' publicly released training code[2] to retrain their networks without the VOC validation set.

We retrained the Deeplab v2 network with ResNet-101 and VGG backbones on Pascal VOC, achieving similar performance to the original authors as shown in Tab. A2. Note that the authors [2] reported results from ablation experiments on the VOC validation set, which we compare to in Tab. A2. However, these models have never been released.

For CRF post-processing, we used the hyperparameters

[1] https://github.com/shelhamer/clockwork-fcn
MD5 checksum of Caffe model: fcae4fdc759f9f461fffc7cc3baa96c6

[2] https://bitbucket.org/aquariusjay/deeplab-public-ver2.git

Table A2: Retrained models on VOC validation set. Details about FCN8, Deeplab v2 and PSPNet can be found in Sec. A1.2.

Model Name	IoU [%]	IoU of authors [%]
FCN8s (VGG) [8]	68.7	–
FCN8s (ResNet) [8]	68.8	–
Deeplab v2 ASPP (VGG) [2]	66.9	68.9
Deeplab v2 ASPP (ResNet) [2]	73.3	–
Deeplab v2 Multiscale ASPP (ResNet) [2]	73.9	76.3
Deeplab v2 Multiscale ASPP (ResNet) + CRF post-processing [2]	74.9	77.7
PSPNet [14]	75.9	–
PSPNet [14] (test set)	79.0	85.4

used by the original authors. As the weights of our trained model are different to the authors, it is possible that different CRF hyperparameters that obtain a higher IoU on the validation set exist.

PSPNet [14]. We used the publicly available model[3] for our experiments on Cityscapes. As the public VOC model has been trained on the entire validation set, we cannot use it for our experiments. Consequently, we retrained this model ourselves achieving comparable results to the original authors (Tab. A2). We followed the training procedure described in the original paper where possible. However, the original authors trained the model using 16 GPUs allowing an effective batch size of 16. Due to our limited computational resources, we could only train on a single GPU using a batch size of 1. The large batch size enabled the original authors to compute better batch statistics for batch normalisation. When using a batch size of 1, the variance in the batch statistics is too high to perform batch normalisation. As a result, we "froze" our batch normalisation layers, and used the batch statistics (mean and variance) of the ImageNet-pretrained ResNet-101 model. This is common practice in training semantic segmentation [2] and object detection [4] networks where batch sizes are typically small.

As shown in Tab. A2, our reimplementation of PSPNet on VOC achieves comparable results to the original authors, even though it has been trained on 1449 fewer images (the VOC validation set). We compared our implementation to the authors on the held-out test set (evaluation is performed on an online server) as the performance on the validation set is not reported in the original paper.

CRF-RNN [15]. We used the publicly available model for Pascal VOC (trained on MS-COCO)[4].

[3] https://github.com/hszhao/PSPNet
MD5 checksum of Caffe model: 29bbdf0ce4d2a6546ed473656db1d6e2

[4] https://github.com/torrvision/crfasrnn
MD5 checksum of Caffe model: bc4926ad00ecc9a1c627db82377ecf56

Table A3: Networks with public models on Cityscapes validation set. We have reported the IoU at 1024×512 inputs, as well as the original 2048×1024 if the network was trained using full-resolution crops.

Model name	IoU at 1024×512	IoU at 2048×1024
E-Net [9]	53.4	–
ICNet [13]	56.5	67.2
FCN8s (VGG) [11]	62.1	66.4
Dilated Frontend [12]	59.0	64.6
Dilated Context [12]	62.3	68.6
PSPNet [14]	74.4	79.7

Table A4: The number of parameters in each of the DNN models evaluated in this paper. As all the networks are stored as 32-bit/4-byte floating point numbers, we reported the number of parameters in megabytes (MB).

Model Name	Dataset	Number of parameters (MB)
E-Net	Cityscapes	1.5
ICNet	Cityscapes	30.1
PSPNet (ResNet-101)	Cityscapes	260.2
Dilated Frontend (VGG)	Cityscapes	512.4
FCN8s (VGG)	Cityscapes	512.5
Dilated Context (VGG)	Cityscapes	512.6
Segnet (VGG)	Pascal	112.4
Deeplab v2 (VGG)	Pascal	144.5
FCN8s (ResNet-101)	Pascal	162.9
Deeplab v2 (ResNet-101)	Pascal	168.4
PSPNet (ResNet-101)	Pascal	272.7
Dilated Frontend (VGG)	Pascal	512.4
FCN8s (VGG)	Pascal	513.0
CRF-RNN (VGG)	Pascal	513.0
Dilated Context (VGG)	Pascal	538.4

DilatedNet [12]. We used the public Pascal VOC and Cityscapes models[5].

[5] https://github.com/fyu/dilation.
MD5 checksum for Pascal VOC: 7a44221dbc2611529bff32029ad1f6e2

ICNet [13]. We used the public Cityscapes model[6].

E-Net [9]. We used the public Cityscapes model[7].

SegNet [1]. We used the public Pascal VOC model[8].

A1.3. Cityscapes dataset

Tab. A3 shows the performance of various publicly available models on the Cityscapes validation set consisting of 500 images. Cityscapes images are captured at a high resolution of 2048×1024, which is too large to fit into GPU memory for most networks. With the exception of E-Net [9] (which is trained on half-resolution images), the other networks we evaluated are trained on smaller crops of full-resolution images. Thereafter, at test time, authors use different tiling strategies [12, 14] to process parts of an image at full resolution before combining the partial results. To make a fairer comparison between models, we process all images at half-resolution so that tiling is not required. In Tab. A3, we show the IoU at the resolution we tested on, 1024×512. And if the model was also trained on full resolution crops, we also include the IoU of the network on full resolution inputs.

A2. Qualitative results

Figure A1 visualises adversarial perturbations of varying ℓ_∞ norms, showing how the perturbations only become visible to the naked eye when the l_∞ of the perturbation, ϵ, is 8 (when viewed on screen). Figure A2 shows the results of the four adversarial attacks considered in this paper when applied on the same image from the Pascal VOC dataset on the Deeplab v2 network. Finally, Fig. A3 compares the outputs of different networks to the Iterative FGSM ll attack for varying values of ϵ on the Cityscapes dataset.

MD5 checksum for Cityscapes: 0de4d78b5f9692f2aba5e7ed88f93ccb

[6] https://github.com/hszhao/ICNet

MD5 checksum of Caffe model: c7038630c4b6c869afaaadd811bdb539

[7] https://github.com/TimoSaemann/ENet

MD5 checksum of Caffe model: d9aabd630cf6bc29c48ea55a86124e14

[8] https://github.com/alexgkendall/SegNet-Tutorial/blob/master/Example_Models/segnet_model_zoo.md

MD5 checksum of Caffemodel: 6e01077e3cda996f95b2a82ea4641a4c

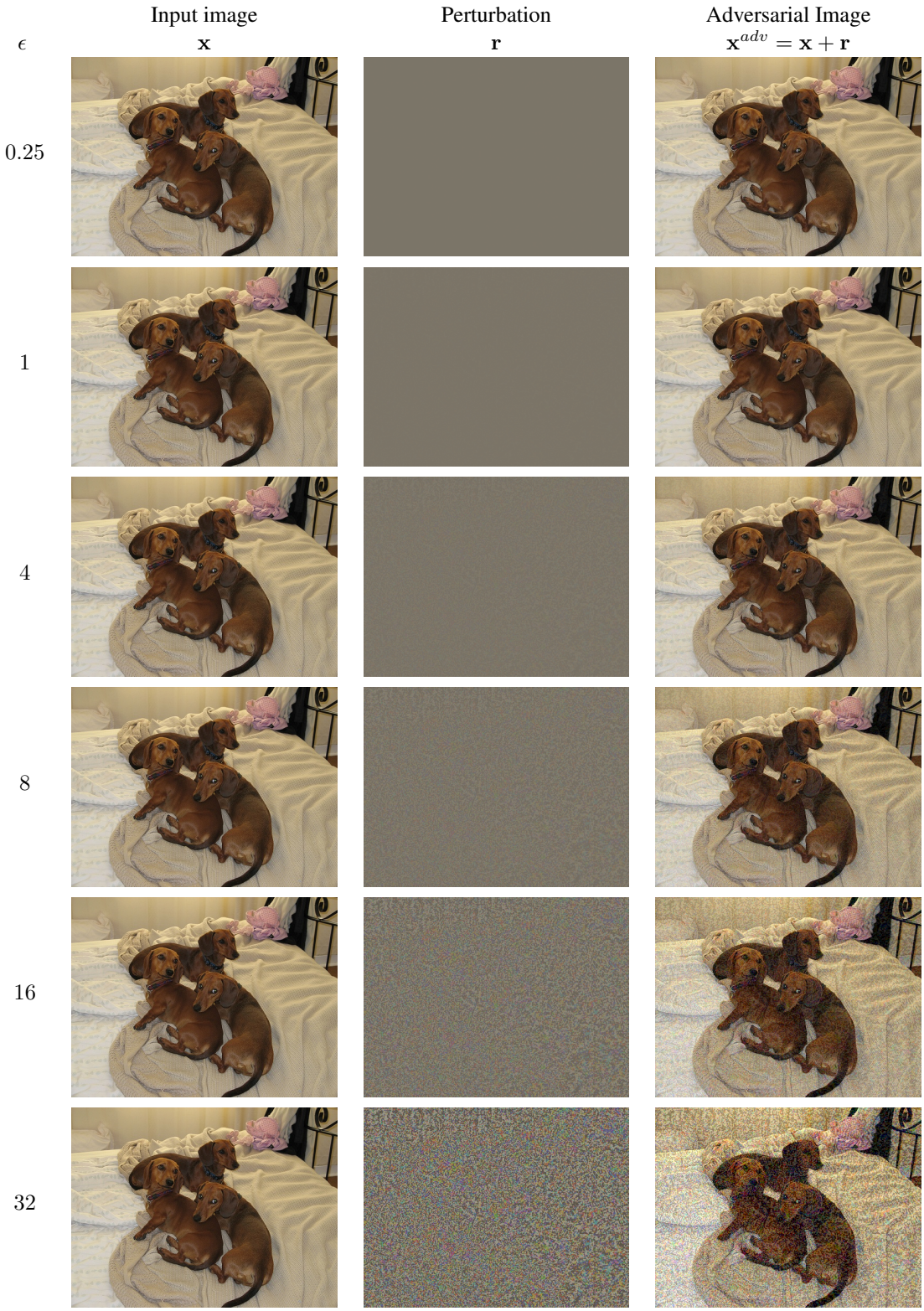


Figure A1: A visualisation of adversarial perturbations of varying ℓ_∞ norms. The perturbation, in the middle column, when added to the input, produces the adversarial example that fools neural networks. Note that the mean RGB value (of the Pascal VOC dataset) is already added to the perturbation, resulting in the grey background. This is required for visualisation as the perturbation can be negative, and RGB images are stored as positive integers $\in [0, 255]$. For $\epsilon = 0.25$, the adversarial image and input image are actually identical if rounded to integers (as RGB images are typically represented). Nevertheless, perturbations of this norm have fooled every neural network studied in this paper. Perturbations become noticeable when viewed on screen at around $\epsilon = 8$. In this figure, perturbations were created using FGSM on Deeplab v2.

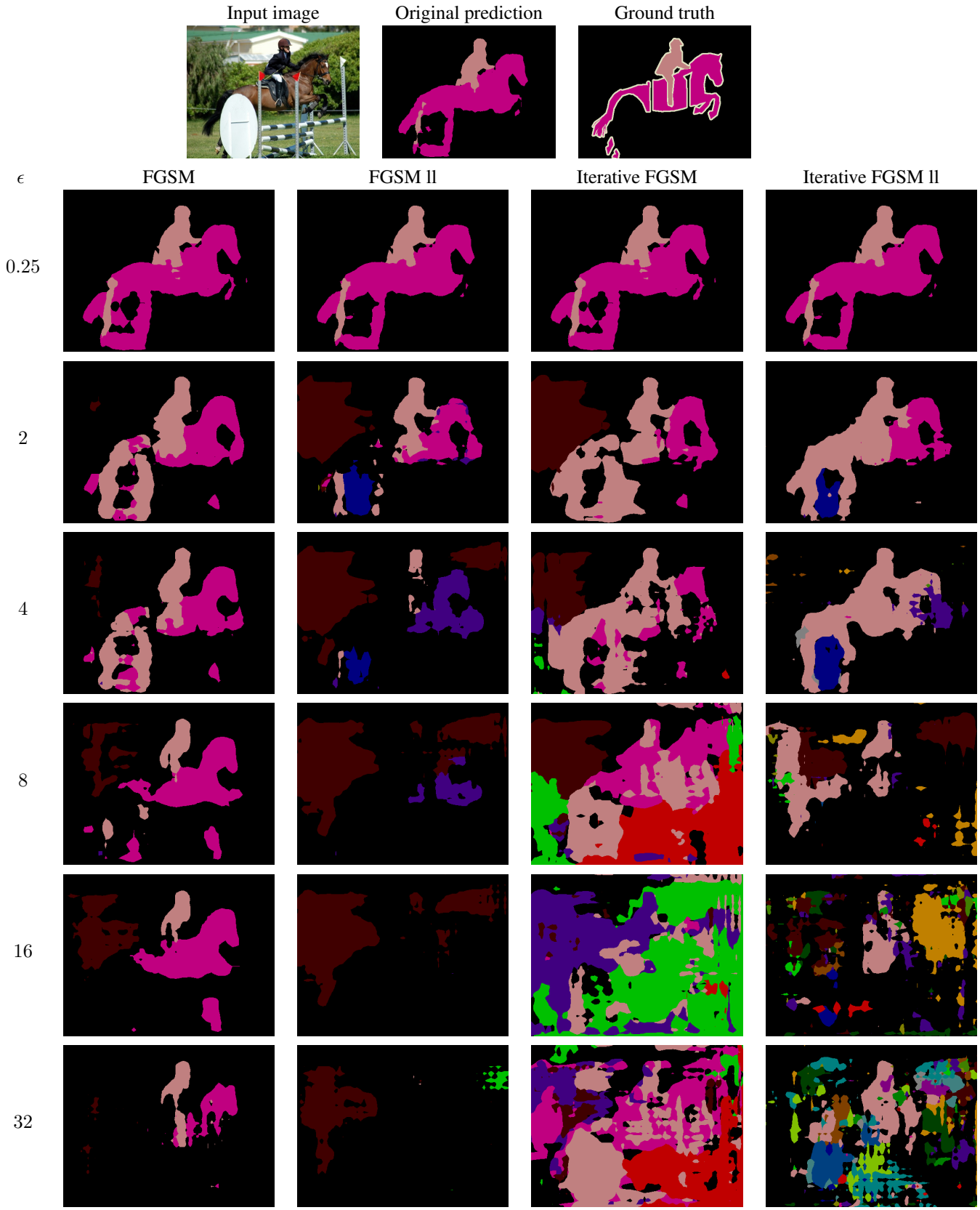


Figure A2: A comparison of different adversarial attacks on the Deeplab v2 Mulitscale ASPP network [2], on a common image from Pascal VOC. As expected, iterative attacks (last two columns) are more effective than single-step ones (first two columns). Higher l_∞ norms of the perturbation, ϵ, also degrade the network's prediction more.

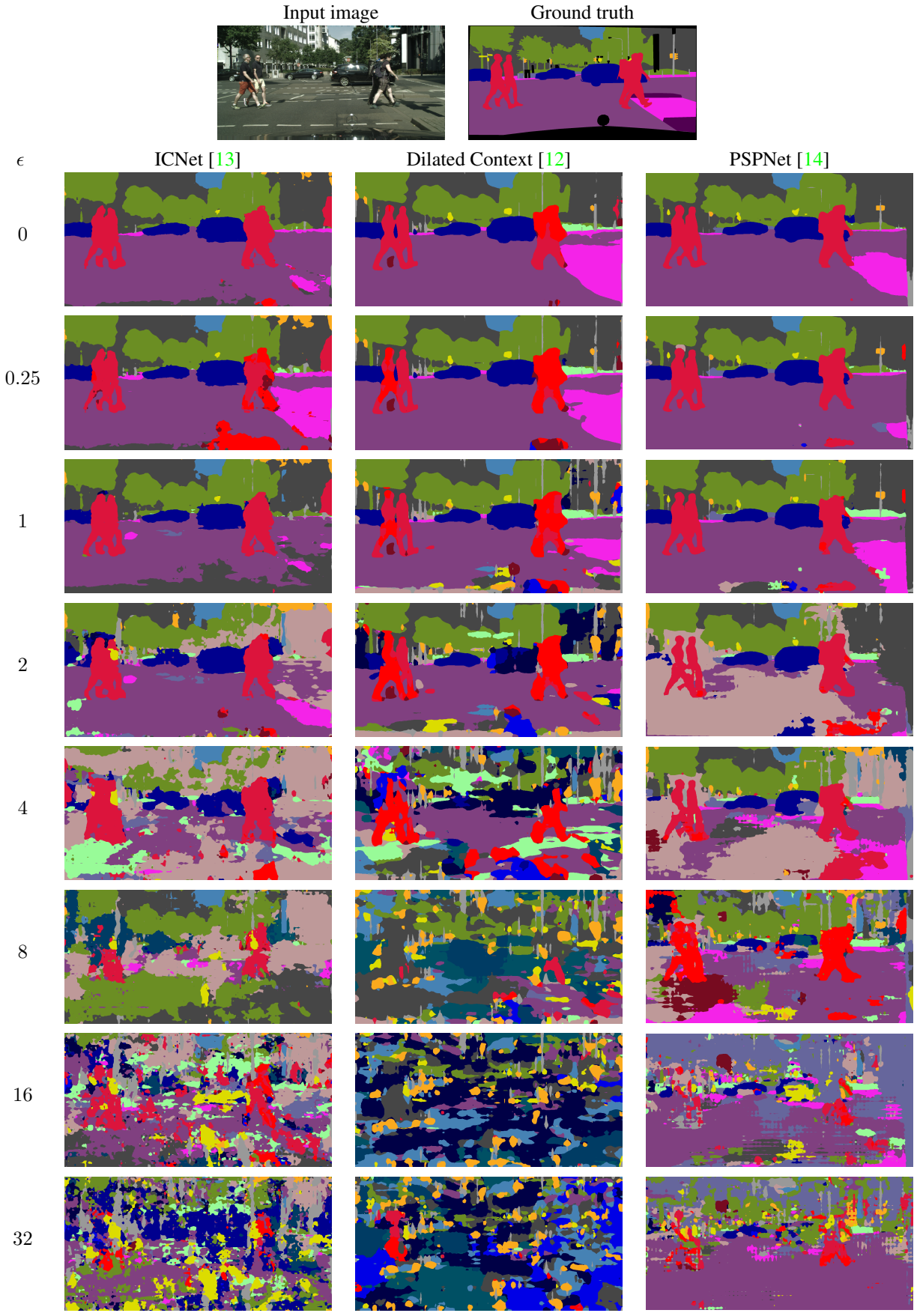


Figure A3: Comparison of ICNet, Dilated Context and PSPNet when attacked by Iterative FGSM ll, for different values of the l_∞ norm, ϵ. Note how each network is affected differently, with PSPNet the most robust. $\epsilon = 0$ is the original prediction of the network, since no perturbation is added here.

A3. Robustness of Different Architectures

The main paper presented results using the FGSM and Iterative FGSM ll attacks for both Pascal VOC and Cityscapes datasets. In this section, we present results for the targeted, single-step FGSM ll and untargeted Iterative FGSM attacks as well. Furthermore, we also include the Absolute IoU scores for each attack for different l_∞ perturbations.

A3.1. Results of other attacks

Figures A4 and A5 show results of the FGSM ll and Iterative FGSM attacks on the VOC and Cityscapes datasets respectively. Our primary observations from the main paper are mostly consistent on these attacks as well:

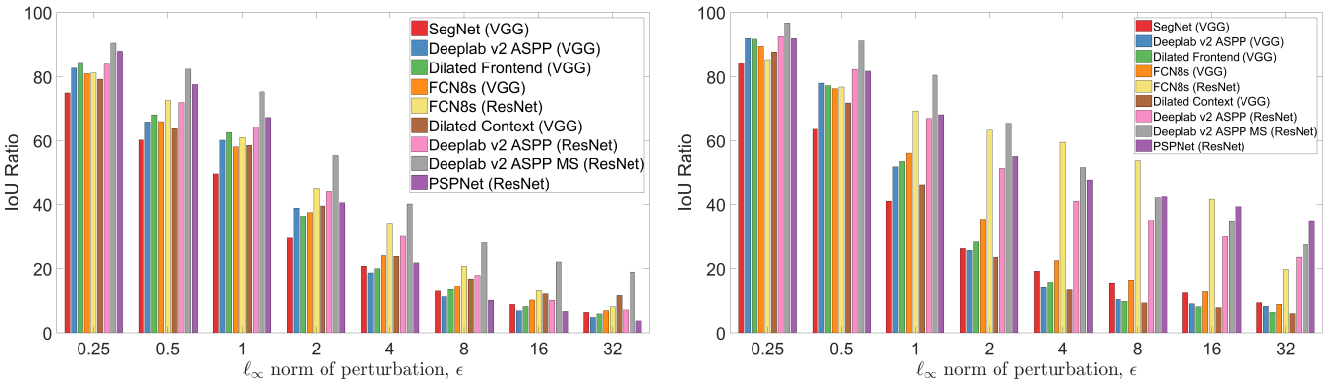
- ResNet based networks are more robust than models based on VGG.
- DilatedNet [12] without its "Context" module is typically more robust than the full, more accurate network.
- E-Net and ICNet show similar robustness to DilatedNet on the Cityscapes dataset. It is only for the FGSM ll attack for $\epsilon \geq 4$ that DilatedNet is robust than both of these lightweight networks.
- Single-step attacks (FGSM ll) are particularly effective on Cityscapes at high ϵ values. They are more effective at fooling networks than iterative methods as well. This was unexpected, and not observed on Pascal VOC.
- PSPNet, which achieves the highest IoU on clean inputs, is typically not the most robust network on Pascal VOC.

A3.2. Result tables of Absolute IoU

In contrast to the main paper that showed the IoU Ratio for various attacks, Tables A5 through A8 show the absolute IoU for different models for each of the FGSM, FGSM ll, Iterative FGSM and Iterative FGSM ll attacks on the Pascal VOC dataset. Additionally, Tables A9 through A12 show the absolute IoU for different models on the Cityscapes dataset.

Note that PSPNet, which achieves the highest IoU on clean inputs, does not usually achieve the highest absolute IoU when attacked on the Pascal VOC dataset. When considering 4 adversarial attacks, and 8 ϵ values, PSPNet achieves the highest absolute IoU in only 2 out of 32 cases. Moreover, it never achieves the highest absolute IoU for imperceptible perturbations ($0 < \epsilon \leq 4$).

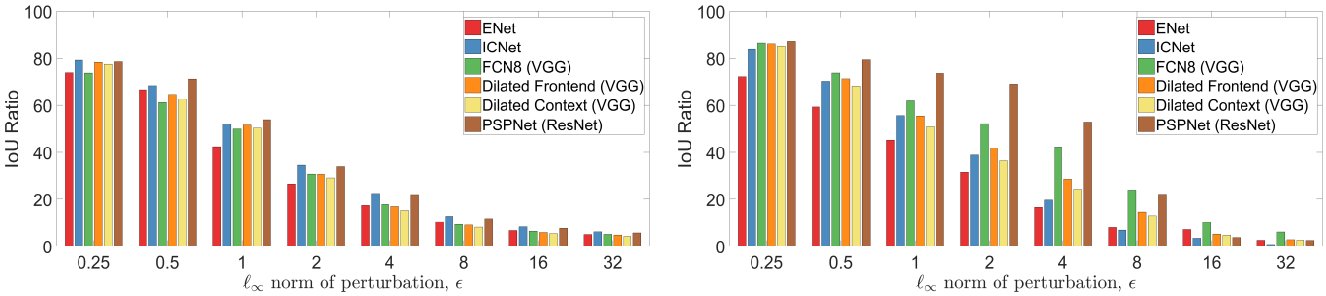
Additionally, the highest absolute IoU for any ϵ value is always from a ResNet-based model (Deeplab v2, FCN8s (ResNet) or PSPNet) on the Pascal VOC dataset. On Cityscapes, FCN8s (VGG) is sometimes the most robust network at high ϵ values. However, the performance of all the networks is severely degraded at this point.



(a) Untargeted Iterative FGSM attack

(b) Targeted FGSM ll. attack

Figure A4: Adversarial robustness of state-of-the-art models on the Pascal VOC dataset. As with the FGSM and Iterative FGSM ll attacks in the main paper, models based on the ResNet backbone are more robust. Deeplab v2 is generally the most robust network, except on the Targeted FGSM attack for $\epsilon \geq 4$. The Iterative FGSM attack is also more effective at fooling the networks than the single-step Targeted FGSM attack, as shown by the lower IoU ratios.



(a) Untargeted Iterative FGSM attack

(b) Targeted FGSM ll. attack

Figure A5: Adversarial robustness of state-of-the-art models on the Cityscapes dataset. As with the FGSM and Iterative FGSM ll attacks in the main paper, PSPNet is typically the most robust. Once again, DilatedNet without its "Context" module is slightly more robust than the full, more accurate network. The single-step FGSM ll attack is also more effective at higher ϵ values than the Iterative FGSM attack. This is unexpected, but was also observed in the main paper between the FGSM and Iterative FGSM ll attacks.

Table A5: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	32.3	25.9	19.5	14.8	11.7	9.7	6.9	4.0
Deeplab v2 ASPP (VGG)	66.9	55.3	44.1	31.7	22.5	17.2	13.9	11.8	9.1
Dilated Frontend (VGG)	67.1	56.7	45.7	33.8	24.2	19.2	16.1	12.2	8.2
FCN8s (VGG)	68.7	55.7	45.4	36.1	28.8	23.9	19.9	16.1	10.3
FCN8s (ResNet)	68.8	55.9	49.9	44.2	39.5	35.9	32.0	24.8	12.8
Dilated Context (VGG)	70.4	55.8	44.9	34.4	26.0	20.6	17.2	13.9	9.0
Deeplab v2 ASPP (ResNet)	73.3	61.6	52.7	43.3	35.9	30.7	27.7	24.6	18.5
Deeplab v2 ASPP MS (ResNet)	73.9	**66.9**	**60.9**	**54.1**	**47.9**	**43.2**	**39.2**	**35.7**	**28.5**
PSPNet (ResNet)	**75.9**	66.8	59.0	48.9	39.8	33.8	29.2	26.7	21.2

Table A6: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	36.2	27.4	17.6	11.4	8.3	6.7	5.4	4.1
Deeplab v2 ASPP (VGG)	66.9	61.5	52.3	34.6	17.3	9.5	7.0	6.1	5.6
Dilated Frontend (VGG)	67.1	61.6	51.9	35.8	19.1	10.6	6.6	5.5	4.4
FCN8s (VGG)	68.7	61.5	52.5	38.6	24.4	15.5	11.4	8.8	6.2
FCN8s (ResNet)	68.8	58.7	52.9	47.7	43.6	**41.0**	**36.8**	28.6	13.6
Dilated Context (VGG)	70.4	61.7	50.5	32.5	16.5	9.4	6.6	5.6	4.3
Deeplab v2 ASPP (ResNet)	73.3	67.8	60.4	49.1	37.5	30.0	25.7	22.0	17.2
Deeplab v2 ASPP MS (ResNet)	73.9	**71.5**	**67.4**	**59.5**	**48.4**	38.0	31.1	25.8	20.4
PSPNet (ResNet)	**75.9**	69.8	62.1	51.8	41.8	36.2	32.1	**29.8**	**26.6**

Table A7: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *Iterative FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	32.3	25.9	21.3	12.7	8.9	5.6	3.8	2.8
Deeplab v2 ASPP (VGG)	66.9	55.3	44.1	40.3	26.0	12.5	7.6	4.7	3.4
Dilated Frontend (VGG)	67.1	56.7	45.7	42.1	24.4	13.4	9.1	5.6	4.1
FCN8s (VGG)	68.7	55.7	45.4	39.9	25.8	16.5	10.0	7.1	4.9
FCN8s (ResNet)	68.8	55.9	49.9	42.0	31.0	23.3	14.2	9.1	5.7
Dilated Context (VGG)	70.4	55.8	44.9	41.2	27.8	16.7	11.9	8.6	8.2
Deeplab v2 ASPP (ResNet)	73.3	61.6	52.7	47.3	32.2	22.1	13.1	7.5	5.3
Deeplab v2 ASPP MS (ResNet)	73.9	**66.9**	**60.9**	**55.8**	**40.9**	**29.6**	**20.9**	**16.3**	**14.0**
PSPNet (ResNet)	**75.9**	66.8	59.0	51.1	30.8	16.5	7.8	5.2	2.8

Table A8: The absolute IoU on the *Pascal VOC* dataset for various models when attacked with *Iterative FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
SegNet (VGG)	43.0	36.2	27.4	22.0	11.4	6.7	5.3	4.1	3.7
Deeplab v2 ASPP (VGG)	66.9	61.5	52.3	49.0	28.0	12.1	6.7	5.8	4.8
Dilated Frontend (VGG)	67.1	61.6	51.9	49.1	27.8	10.8	5.4	4.0	3.7
FCN8s (VGG)	68.7	61.5	52.5	52.5	33.0	17.1	10.4	8.4	6.8
FCN8s (ResNet)	68.8	58.7	52.9	47.8	37.6	28.9	**18.2**	**12.2**	**7.9**
Dilated Context (VGG)	70.4	61.7	50.5	48.9	22.9	9.2	5.6	5.0	4.1
Deeplab v2 ASPP (ResNet)	73.3	67.8	60.4	56.9	39.6	21.1	11.3	7.7	6.3
Deeplab v2 ASPP MS (ResNet)	73.9	**71.5**	**67.4**	**65.2**	**52.6**	**30.2**	15.5	9.1	7.1
PSPNet (ResNet)	**75.9**	69.8	62.1	58.5	37.2	20.0	11.1	7.9	5.1

Table A9: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	39.6	35.6	31.0	24.0	13.2	5.8	4.1	1.4
ICNet	56.5	47.0	41.3	35.5	28.5	16.8	4.5	2.4	0.8
FCN8 (VGG)	62.1	46.0	38.0	31.9	27.8	23.9	**16.2**	**7.7**	**3.9**
Dilated Frontend (VGG)	59.0	46.3	38.1	31.1	25.7	20.7	13.3	5.0	1.7
Dilated Context (VGG)	62.3	48.4	39.0	31.6	26.0	20.8	13.3	4.8	1.8
PSPNet (ResNet)	**74.4**	**58.5**	**52.9**	**48.9**	**46.0**	**36.3**	16.0	2.8	1.9

Table A10: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	38.5	31.7	24.2	17.0	8.9	4.3	3.8	1.4
ICNet	56.5	47.2	40.5	33.2	25.1	13.4	3.4	2.3	0.8
FCN8 (VGG)	62.1	53.8	46.0	38.4	32.5	26.3	14.9	**6.4**	**3.8**
Dilated Frontend (VGG)	59.0	50.9	42.0	32.8	24.6	16.8	8.7	3.1	1.7
Dilated Context (VGG)	62.3	53.2	42.5	31.8	22.8	15.1	8.2	3.0	1.7
PSPNet (ResNet)	**74.4**	**64.9**	**59.1**	**55.0**	**51.3**	**39.5**	**16.5**	2.8	1.9

Table A11: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *Iterative FGSM*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	39.6	35.6	22.6	14.2	9.3	5.7	3.6	2.7
ICNet	56.5	47.0	41.3	30.9	22.4	13.6	7.6	4.8	3.4
FCN8 (VGG)	62.1	46.0	38.0	31.1	19.1	11.1	5.8	4.0	3.2
Dilated Frontend (VGG)	59.0	46.3	38.1	30.6	18.1	10.0	5.4	3.5	2.8
Dilated Context (VGG)	62.3	48.4	39.0	31.4	18.1	9.6	5.1	3.4	2.7
PSPNet (ResNet)	**74.4**	**58.5**	**52.9**	**40.2**	**25.4**	**16.4**	**8.9**	**5.7**	**4.3**

Table A12: The absolute IoU on the *Cityscapes* dataset for various models when attacked with *Iterative FGSM ll*. This is evaluated for eight different values of the ℓ_∞ norm of the perturbation, ϵ. $\epsilon = 0$ represents the IoU on clean inputs.

Network	ℓ_∞ norm of perturbation, ϵ								
	0	0.25	0.5	1	2	4	8	16	32
ENet	53.4	38.5	31.7	19.2	14.6	9.2	5.1	3.5	2.7
ICNet	56.5	47.2	40.5	33.8	22.4	14.5	8.9	6.8	5.5
FCN8 (VGG)	62.1	53.8	46.0	36.5	24.8	14.0	7.7	5.9	4.9
Dilated Frontend (VGG)	59.0	50.9	42.0	31.8	20.0	10.5	5.3	4.7	4.0
Dilated Context (VGG)	62.3	53.2	42.5	32.2	19.9	8.8	4.8	3.6	2.8
PSPNet (ResNet)	**74.4**	**64.9**	**59.1**	**46.1**	**36.5**	**26.1**	**16.9**	**11.5**	**8.8**

A4. Multiscale Processing and Transferability of Adversarial Examples

This section details additional results with both Deeplab v2 and FCN8s.

A4.1. Deeplab v2

Table A13 shows the performance, measured in IoU, on the VOC validation set when the input image is processed at different resolutions (50%, 75%, 100%). The fact that a different IoU is obtained for each input resolution, even though the weights of the network are the same, confirms that the network is not scale invariant. Note that the version of Deeplab which processes images at all the aforementioned resolutions, and max-pools the prediction at each pixel obtains the highest IoU. An alternative to max-pooling the predictions from each scale is to average-pool them. This method gives an insignificant improvement in accuracy, but does improve robustness as shown in Fig. A6.

Table A13: Performance of Deeplab v2 (ResNet) on the VOC validation set when processing images at different resolutions

Model Name	IoU [%]
Deeplab v2 50% scale	67.8
Deeplab v2 75% scale	71.9
Deeplab v2 100% scale	73.3
Deeplab v2 100% scale (average pooling)	73.4
Deeplab v2 Multiscale (max pooling)	73.9

A4.1.1 Average-pooling instead of max-pooling

As shown in Fig. A6, average-pooling the results from each scale is also more robust to all the adversarial attacks we tested compared to the single-scale version of Deeplab v2. In fact, multiscale processing (either max- or average-pooling) achieves a higher IoU Ratio at almost all ϵ values for each attack.

Table A15 also shows that black-box attacks generated from multiscale-averaging also transfer better to single scales of Deeplab v2, for all four adversarial attacks considered in this paper. This is similar to the case of max-pooling as shown in the main paper.

A4.1.2 Transferability experiments using the FGSM ll and Iterative FGSM attacks

Table A16 shows the transferability of adversarial attacks to different scales of Deeplab v2 using the FGSM ll and Iterative FGSM attacks. The main paper presented results using the FGSM and Iterative FGSM ll attacks. However, our findings remain consistent on these different attacks. The multiscale version of Deeplab v2 is the most robust to these attacks (as also seen in Fig. A4 and A6), and black-box attacks from it transfer the best to other scales of Deeplab v2.

Table A14: Performance of FCN8s when processing images at different resolutions. As with Deeplab v2, max-pooling the predictions from multiple scales achieves the best results.

Model Name	IoU [%]
FCN8s 50% scale	60.8
FCN8s 75% scale	67.8
FCN8s 100% scale	68.7
FCN8s Multiscale	69.9

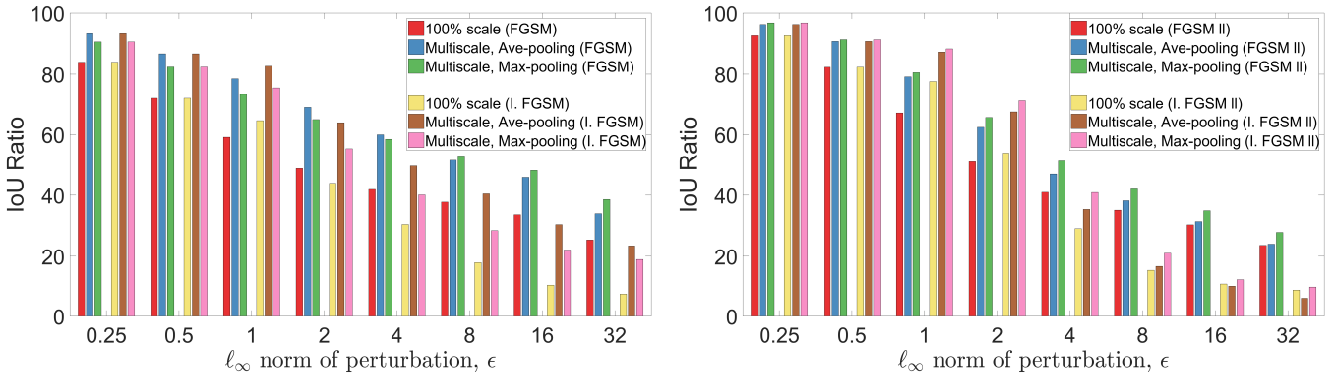
A4.1.3 Transferability experiments at multiple ϵ values

Figure A7 shows the results of black-box attacks for multiple ϵ values between different scales of Deeplab v2 for the FGSM attack. The results are largely consistent with those at $\epsilon = 8$ as reported in the main paper – the multiscale version of Deeplab v2 is the most robust to white-box attacks and black-box attacks generated from it transfer the best to other scales of Deeplab v2. Also note how the transferability from each scale to another varies greatly. For example, attacks generated from the 50% scale transfer very poorly to 100% and vice versa.

A4.2. FCN8s

Table A14 shows the IoU of FCN8s (VGG) as the input resolution of the image is varied from the VOC dataset. As with Deeplab v2, a multiscale version which max-pools the predictions from each scale achieves the highest IoU.

The transferability experiments from Section 6 of the paper are repeated on FCN8 in Tables A17 and A18. Note that FCN8s has not been trained in a multiscale manner as Deeplab v2, and it is rather done as a post-processing step. Nevertheless, the results show a similar trend as Deeplab v2: The multiscale network is more robust to white-box attacks and black-box attacks generated from it transfer better. This suggests that training the network in a multiscale manner does not confer robustness to adversarial examples. Rather it is the fact that CNNs are not scale invariant, and that adversarial examples generated at one scale are not as malignant at another. Finally Fig. A8 shows the transferability experiments at multiple ϵ values, as was done for Deeplab v2 in the previous subsection.



(a) Untargeted attacks

(b) Targeted attacks

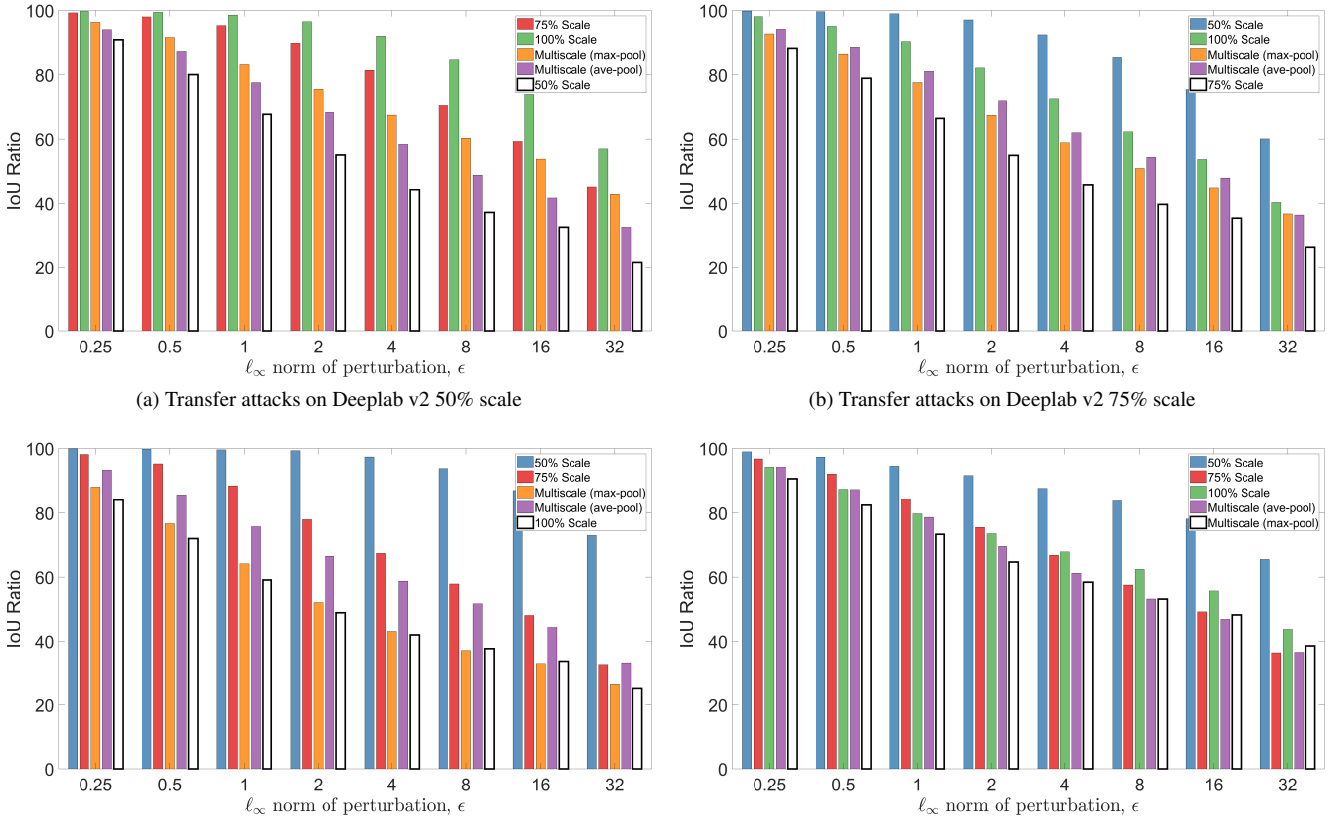
Figure A6: Adversarial robustness of Deeplab ASPP (single-scale) and Deeplab Multiscale ASPP. We compare two types of multiscale ensembling – max-pooling and average-pooling the predictions from each of the three scales of Deeplab v2 (ResNet 101). Note that both average- and max-pooling are more robust than just a single-scale model, achieving higher IoU Ratios for almost every ϵ value for each attack on the Pascal VOC dataset.

Table A15: Transferability of adversarial examples generated from different scales of Deeplab v2 (columns) and evaluated on different networks (rows). In this case, the outputs from each scale are *average-pooled* instead of max-pooled. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of Deeplab v2. In the case of Iterative FGSM ll, black-box attacks from the multiscale networks are sometimes even more effective than white-box ones.

Network evaluated	FGSM ($\epsilon = 8$)				Iterative FGSM ll ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 0.5 (ResNet)	<u>37.3</u>	70.5	84.8	**48.8**	<u>18.0</u>	92.0	96.9	**12.1**
Deeplab v2 0.75 (ResNet)	85.5	<u>39.7</u>	62.2	**54.2**	99.5	<u>17.9</u>	89.9	**17.4**
Deeplab v2 1 (ResNet)	93.6	57.9	<u>37.7</u>	**51.7**	100.0	79.0	<u>15.5</u>	**9.6**
Deeplab v2 Multiscale (ResNet)	75.1	**54.2**	59.0	<u>51.6</u>	95.2	**84.9**	87.5	<u>16.7</u>
Network evaluated	FGSM ll ($\epsilon = 8$)				Iterative FGSM ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 50% (ResNet)	<u>36.4</u>	70.1	83.7	**36.6**	<u>21.3</u>	90.9	97.0	**37.3**
Deeplab v2 75% (ResNet)	89.9	<u>37.4</u>	61.6	**39.9**	99.1	<u>20.0</u>	88.6	**44.1**
Deeplab v2 100% (ResNet)	95.1	58.3	<u>35.1</u>	**36.9**	100.2	71.9	<u>18.6</u>	**33.5**
Deeplab v2 Multiscale (ResNet)	96.0	**91.4**	94.7	<u>38.2</u>	94.5	**76.2**	86.5	<u>37.7</u>

Table A16: Transferability of adversarial examples generated from different scales of Deeplab v2 (columns) and evaluated on different networks (rows). As with the main paper, *max-pooling* is performed from the output of each scale. However, in contrast to the main paper, the FGSM ll and Iterative FGSM attacks are reported. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of Deeplab v2.

Network evaluated	FGSM ll ($\epsilon = 8$)				Iterative FGSM ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
Deeplab v2 0.5 (ResNet)	<u>36.4</u>	70.1	83.7	**46.0**	<u>21.3</u>	90.9	97.0	**39.2**
Deeplab v2 0.75 (ResNet)	89.9	<u>37.4</u>	61.6	**43.3**	99.1	<u>20.0</u>	88.6	**34.0**
Deeplab v2 1 (ResNet)	95.1	58.3	<u>35.1</u>	**33.9**	100.2	71.9	<u>18.6</u>	**22.0**
Deeplab v2 Multiscale (ResNet)	90.7	**60.8**	68.9	<u>42.1</u>	96.5	**81.9**	87.5	<u>29.2</u>
Deeplab v2 (VGG)	95.1	69.9	63.8	**61.9**	98.5	86.9	86.3	**81.2**
FCN8 (VGG)	94.5	67.7	64.7	**62.4**	98.7	86.9	86.0	**82.0**



(a) Transfer attacks on Deeplab v2 50% scale

(b) Transfer attacks on Deeplab v2 75% scale

(c) Transfer attacks on Deeplab v2 100% scale

(d) Transfer attacks on Deeplab v2 Multiscale (max-pooling)

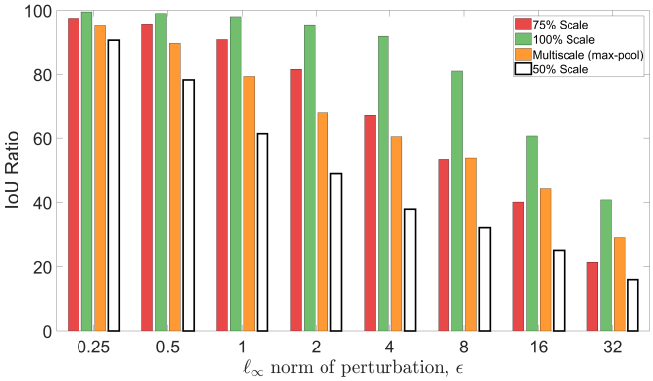
Figure A7: Black-box attacks on each scale of Deeplab v2, from each other scale, using adversarial perturbations generated by FGSM for differing values of ϵ on the Pascal VOC dataset. In each figure, the last bar shows the "white-box" attack on the network, where the attack is generated from the network that is being evaluated. This is typically the most powerful attack, as expected. Note that attacks generated from the multiscale version of Deeplab v2 (using either max- or average-pooling) produce the most effective black-box attacks across multiple ϵ values. The trend from the main paper, which only tabulated the IoU Ratio for $\epsilon = 8$, can thus be seen across all other ϵ values considered in this paper.

Table A17: Transferability of adversarial examples generated from different scales of FCN8s (VGG) (columns) and evaluated on different networks (rows) on the Pascal VOC dataset. For the multiscale network, the outputs from each scale are max-pooled. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of FCN8s.

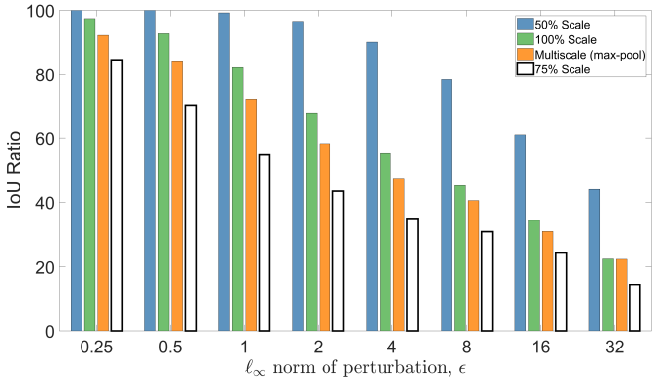
Network evaluated	FGSM ($\epsilon = 8$)				Iterative FGSM ll ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
FCN8 50%	<u>32.1</u>	**53.3**	81.0	53.7	<u>20.5</u>	87.3	96.9	**21.9**
FCN8 75%	78.4	<u>30.9</u>	45.5	**40.5**	96.3	<u>17.6</u>	77.8	**20.5**
FCN8 100%	94.0	41.7	<u>28.9</u>	**28.7**	98.2	58.6	<u>15.3</u>	**17.5**
FCN8 Multiscale	79.1	**42.8**	53.3	<u>47.8</u>	97.5	**79.3**	85.2	<u>20.0</u>

Table A18: Transferability of adversarial examples generated from different scales of FCN8s (VGG) (columns) and evaluated on different networks (rows) on the Pascal VOC dataset. For the multiscale network, the outputs from each scale are max-pooled. The underlined diagonals for each attack show white-box attacks. Off-diagonals, show transfer (black-box) attacks. The most effective one in bold, is typically from the multiscale version of FCN8s.

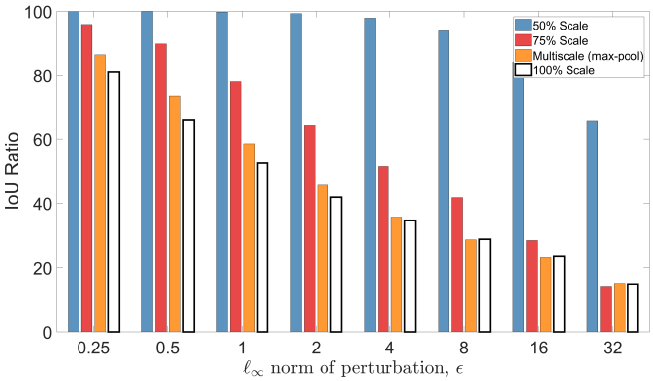
Network evaluated	FGSM ll ($\epsilon = 8$)				Iterative FGSM ($\epsilon = 8$)			
	50%	75%	100%	Multiscale	50%	75%	100%	Multiscale
FCN8 50%	<u>18.5</u>	51.4	79.2	**24.0**	<u>23.6</u>	85.7	97.1	**38.1**
FCN8 75%	80.9	<u>18.5</u>	37.0	**23.4**	97.3	<u>15.9</u>	74.7	**28.1**
FCN8 100%	93.0	33.8	<u>16.6</u>	**17.1**	99.1	54.9	<u>14.7</u>	**18.1**
FCN8 Multiscale	87.5	**40.0**	60.3	<u>21.1</u>	96.4	**74.5**	82.3	<u>25.1</u>



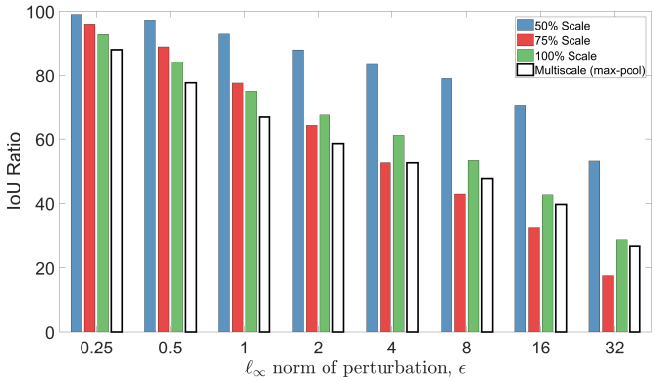
(a) Transfer attacks on FCN8s 50% scale



(b) Transfer attacks on FCN8s 75% scale



(c) Transfer attacks on FCN8s 100% scale



(d) Transfer attacks on FCN8s Multiscale (max-pooling)

Figure A8: Black-box attacks on each scale of FCN8, from each other scale, using adversarial perturbations generated by FGSM for differing values of ϵ on the Pascal VOC dataset. In each figure, the last bar shows the "white-box" attack on the network, where the attack is generated from the network that is being evaluated. The results from this experiment are very similar to Deeplab v2 – attacks generated from the multiscale network transfer the best to other scales. However, unlike Deeplab v2, the FCN8s network in this case was not trained with multiscale ensembling. This was simply done at test-time. This suggests that the increased robustness of multiscale networks to adversarial attacks, and their transferability to other networks, is not a result of the training procedure, but rather the fact that these networks are not scale invariant.

A5. Effect of CRFs on Adversarial Robustness

A5.1. Adversarial Robustness and Smoothing

The pairwise term of DenseCRF [6] (which is interpreted as a neural network in CRF-RNN [15]) takes the form of a weighted sum of a Bilateral and Gaussian filter.

$$\psi_p(x_i, x_j) = \mu(x_i, x_j)\left[w_1 \exp\left(\frac{|p_i - p_j|^2}{\theta_\alpha} + \frac{|I_i - I_j|^2}{\theta_\beta}\right) + w_2 \exp\left(\frac{|p_i - p_j|^2}{\theta_\gamma}\right)\right]. \quad (1)$$

Increasing θ_α, θ_β, θ_γ, w_1 and w_2 all correspond to favouring smoother predictions. The compatibility function, $\mu(x_i, x_j)$, is given by the Potts model, and is equal to 1 if $x_i \neq x_j$ and 0 otherwise [6].

Figure A9 shows the effect of varying θ_α, Fig. A10 the effect of varying θ_β and Fig. A11 the effect of varying both θ_γ and w_2. Note that in all cases, each of the other hyperparameters remains unchanged at the values from the public CRF-RNN model.

In all of these cases, we can see that increasing the smoothness does not correspond to increasing adversarial robustness to the FGSM attack. Rather, as detailed in the next subsection, there is a correlation between the confidence of the prediction and robustness to the FGSM attack.

A5.2. Results about the confidence on VOC

We empirically measured the confidence of the predictions of CRF-RNN. This was done by recording the probability (from the softmax activation function) of the predicted (highest-scoring) label, and also by calculating the entropy of the marginal distribution over labels at each pixel in the image. A lower entropy indicates a more certain or confident prediction. This was then averaged over the Pascal VOC validation set.

Figures A13 and A14 show the mean confidence and entropy respectively as a function of the IoU Ratio. This is done for the FGSM attack for all the ϵ values considered in the paper. There is a clear correlation between the IoU Ratio and the confidence of the prediction. Moreover, the results of CRF-RNN are always more confident than FCN8s. Note that multiple variants of CRF-RNN, using different θ_α, θ_β and θ_γ hyperparameter values were considered, as in Figures A9 through A11.

A5.3. Experiments on Deeplab v2 with CRF

In contrast to CRF-RNN [15], a common approach is to apply CRFs as a post-processing step, as done in Deeplab [2]. We perform adversarial attacks on this by appending the CRF-RNN layer of [15] onto the Deeplab v2 network. This allows us to compute the gradient of the loss with respect to the input image (required for all the attacks) by backpropagating through the CRF-RNN layer. The parameters of the CRF-RNN layer appended to Deeplab v2 were manually set to the parameters used by the original authors[9] (who obtained them via cross-validation). Note that appending the CRF-RNN layer to Deeplab v2 and using the same parameters as the authors produces output that is identical to the post-processing code used by the original authors. The difference is that this allows us to compute gradients as well.

Figures A12a and A12c show the results of targeted and untargeted attacks on Deeplab v2 with a CRF on the Pascal VOC dataset. As in the main paper, we also compute the adversarial attack from the Deeplab v2 part of the network (which produces "unaries"), and then use these perturbations to attack the entire Deeplab v2 with CRF network (Fig. A12b).

The results from these experiments are consistent with the ones of CRF-RNN in the main paper: Appending the CRF at the end of the network confers resistance to only untargeted attacks. For targeted attacks, there is barely any difference in robustness. Finally, untargeted adversarial perturbations generated from Deeplab v2, and then tested on Deeplab v2 + CRF, are actually more effective than white-box attacks on Deeplab v2 + CRF. This is due to the "gradient masking" effect of mean-field inference of CRFs which make the final prediction of the network more confident, and thus lead to gradients of the loss with respect to the input (in the untargeted case) which have smaller norm.

References

[1] V. Badrinarayanan, A. Handa, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. *CoRR*, abs/1505.07293, 2015. 1, 3

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915v2*, 2016. 1, 2, 5, 15

[3] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *ICCV*, 2011. 1

[4] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, 2017. 2

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 1

[9] http://liangchiehchen.com/projects/DeepLabv2_resnet.html

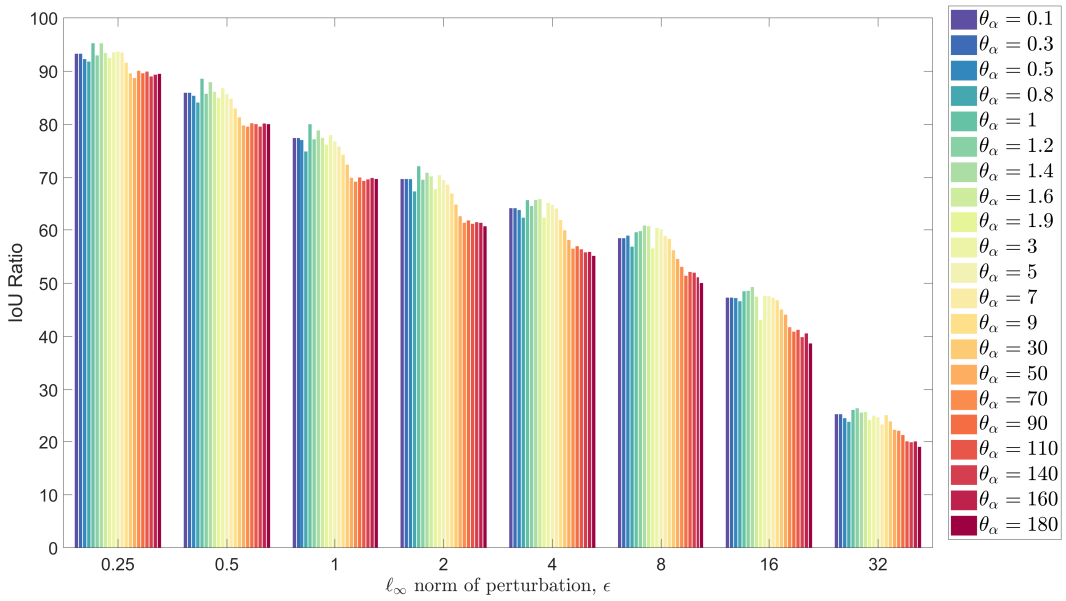


Figure A9: The IoU Ratio of CRF-RNN for various values of the θ_α (filter bandwidth) hyperparameter when attacked with FGSM on the Pascal VOC dataset. Increasing this hyperparameter visually smoothes the result further, but we can see that this does not increase adversarial robustness. In fact, lower filter bandwidths of approximately $\theta_\alpha = 1$ provide more robustness.

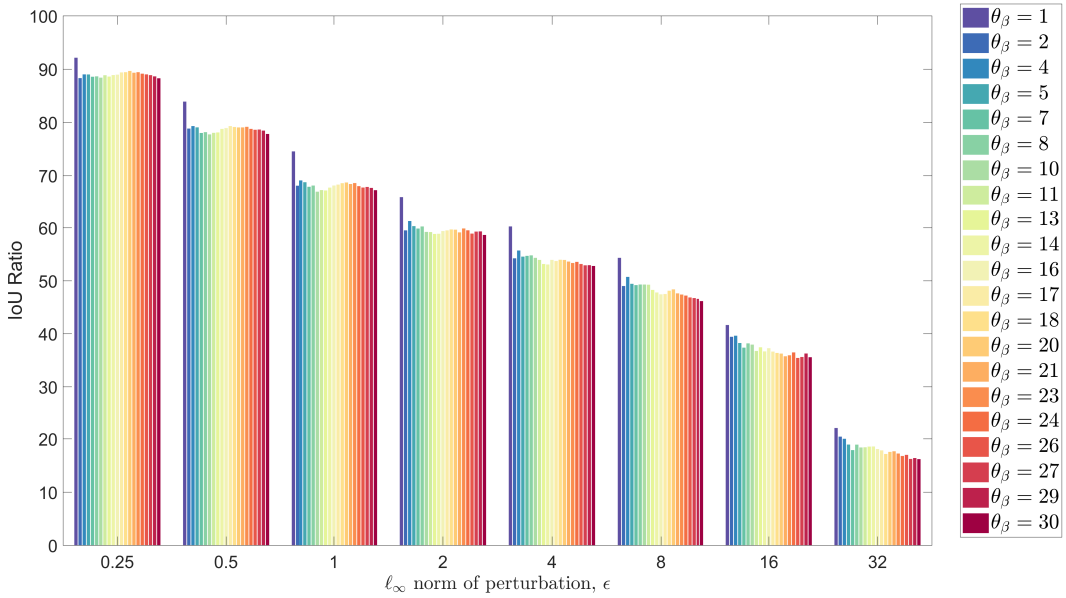


Figure A10: The IoU Ratio of CRF-RNN for various values of the θ_β (filter bandwidth) hyperparameter when attacked with FGSM on the Pascal VOC dataset. Again, we can see that larger filter bandwidths, which encourage more spatial smoothness, do not increase adversarial robustness.

[6] P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *NIPS*, 2011. 15

[7] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*. 2014. 1

[8] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 2

[9] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. In *arXiv preprint arXiv:1606.02147v1*, 2016. 2, 3

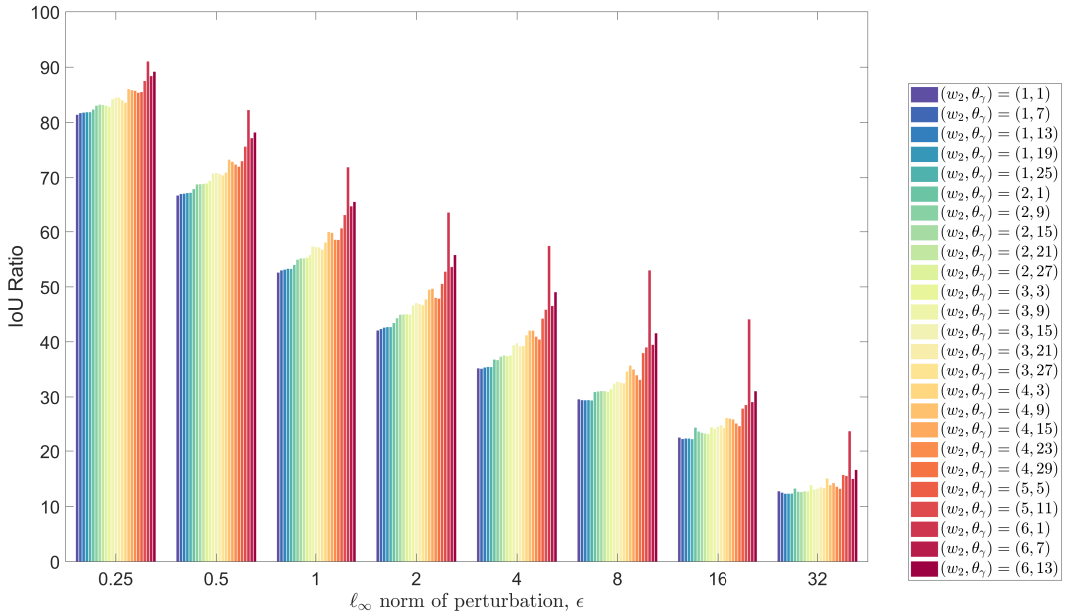
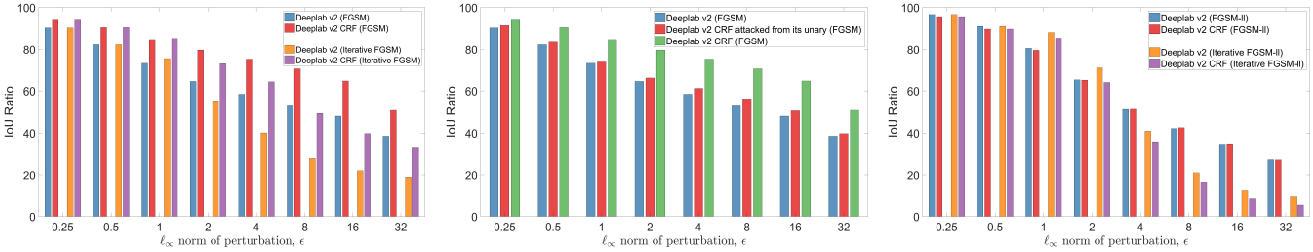


Figure A11: The IoU Ratio of CRF-RNN for various values of the w_2 and θ_γ parameters when attacked with FGSM on the Pascal VOC dataset. Increasing the weight of the Gaussian term (w_2) tends to increase robustness. However, we still see that lower filter bandwidths (θ_γ) tend to provide more robustness.



(a) (b) (c)

Figure A12: (a) On untargetted attacks, Deeplab v2 with a CRF is noticably more robust than just the Deeplab v2 network. (b) Attacks created from the base Deeplab v2 network using FGSM are more effective than those created from Deeplab v2 with CRF. This is due to the "gradient masking" effect of mean-field inference of CRFs. (c) However, the CRF does not "mask" the gradient for targeted attacks. As a result, Deeplab v2 with a CRF is no more robust than just the Deeplab v2 network.

[10] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *PAMI*, 39(4):640–651, 2017. 1

[11] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. In *ECCV 2016 Workshop*, pages 852–868. Springer, 2016. 1, 2

[12] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 1, 2, 3, 6, 7

[13] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. Icnet for real-time semantic segmentation on high-resolution images. In *arXiv preprint arXiv:1704.08545v1*, 2017. 2, 3, 6

[14] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 2, 3, 6

[15] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015. 1, 2, 15

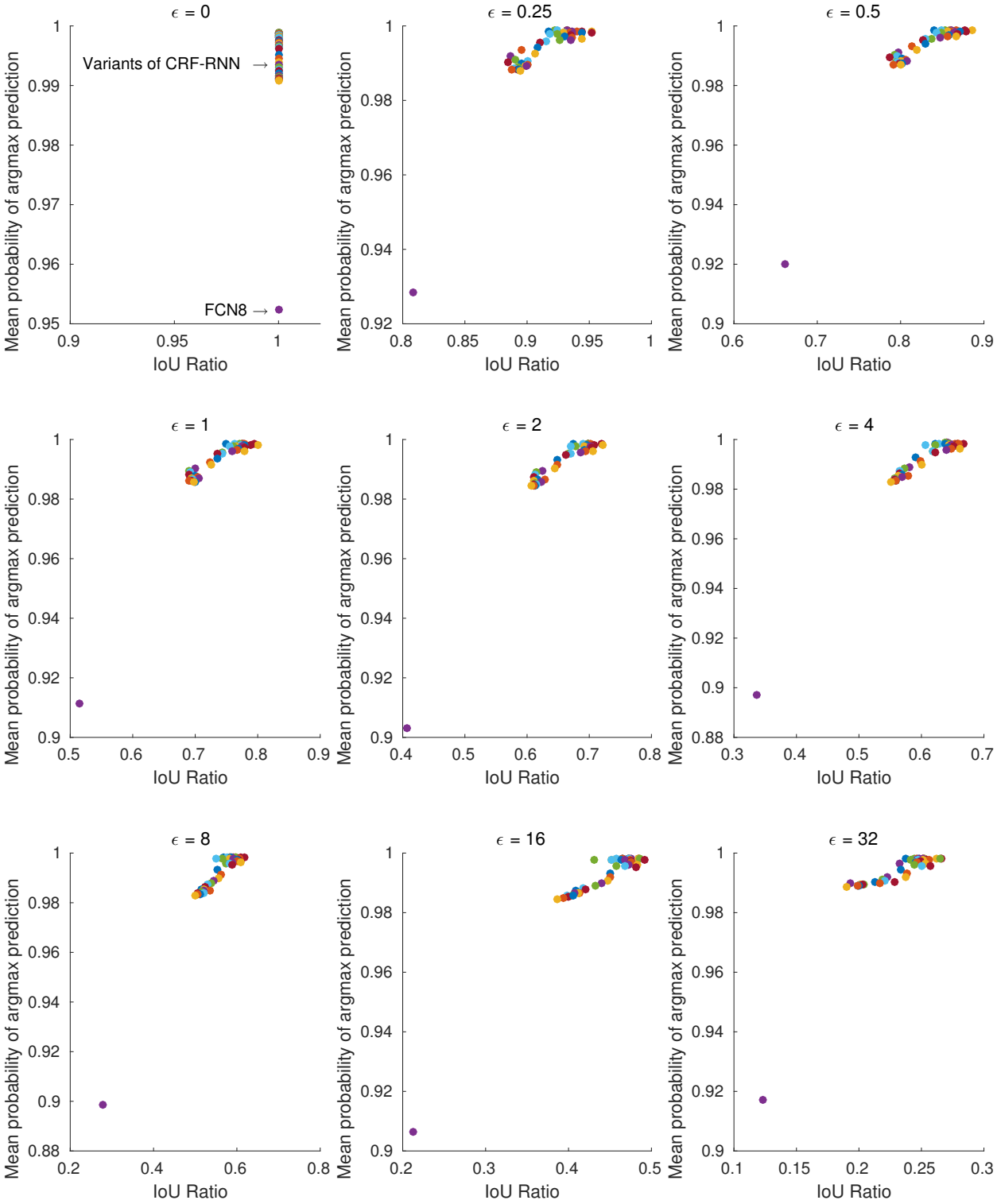


Figure A13: The mean probability of the highest-scoring class for each pixel, averaged over the Pascal VOC validation set. This is performed for the FGSM attack for multiple ϵ values. $\epsilon = 0$ corresponds to clean inputs (no adversarial attack). Note how FCN8s (the purple dot) consistently has the lowest mean probability. This probability is significantly lower than other variants of CRF-RNN (with varying θ_α, θ_β, θ_γ), shown by the other coloured dots. Moreover, note the correlation between the confidence in the prediction, and adversarial robustness to the FGSM attack. Additionally, the probability of the predicted class remains high (above 90%) for all models throughout all adversarial attacks.

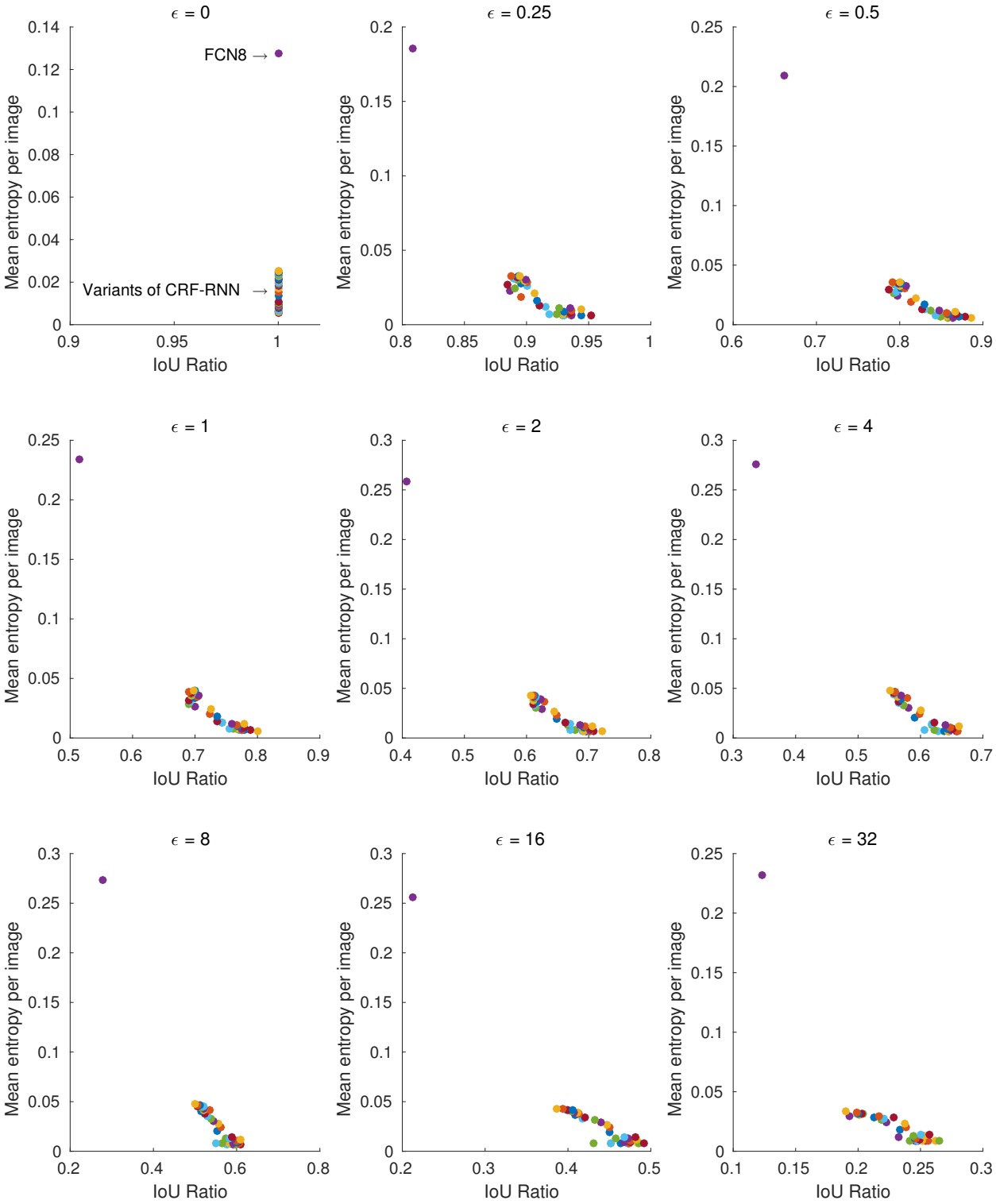


Figure A14: The mean entropy of the marginal distribution over all labels at each pixel, averaged over all images in the Pascal VOC validation set. A lower entropy corresponds to a more confident prediction. This is performed for the FGSM attack for multiple ϵ values. $\epsilon = 0$ corresponds to clean inputs (no adversarial attack). Note how FCN8s (the purple dot) consistently has the highest mean entropy (least confidence). This entropy is significantly higher than other variants of CRF-RNN (with varying θ_α, θ_β, θ_γ), shown by the other coloured dots. Moreover, note the correlation between the confidence in the prediction, and adversarial robustness to the FGSM attack.