NIPS 2017: Defense Against Adversarial Attack Competition

4-th place solution by team “erko” https://www.kaggle.com/erkowa
Yerkebulan Berdibekov e.berdibekov@gmail.com

Organized by Google Brain kaggle

About competition

Most existing machine learning classifiers are highly vulnerable to adversarial examples. An **adversarial example** is a sample of input data which has been modified very slightly in a way that is intended to cause a machine learning classifier to misclassify it. In many cases, these modifications can be so subtle that a human observer does not even notice the modification at all, yet the classifier still makes a mistake.

Goals of this competition is to develop defense against attacks developed by another adversarial attack competition. In final round all defenders are tested on adversarial samples generated by attackers. Attackers generated adversarial samples on 5000 test set images (ImageNet format).

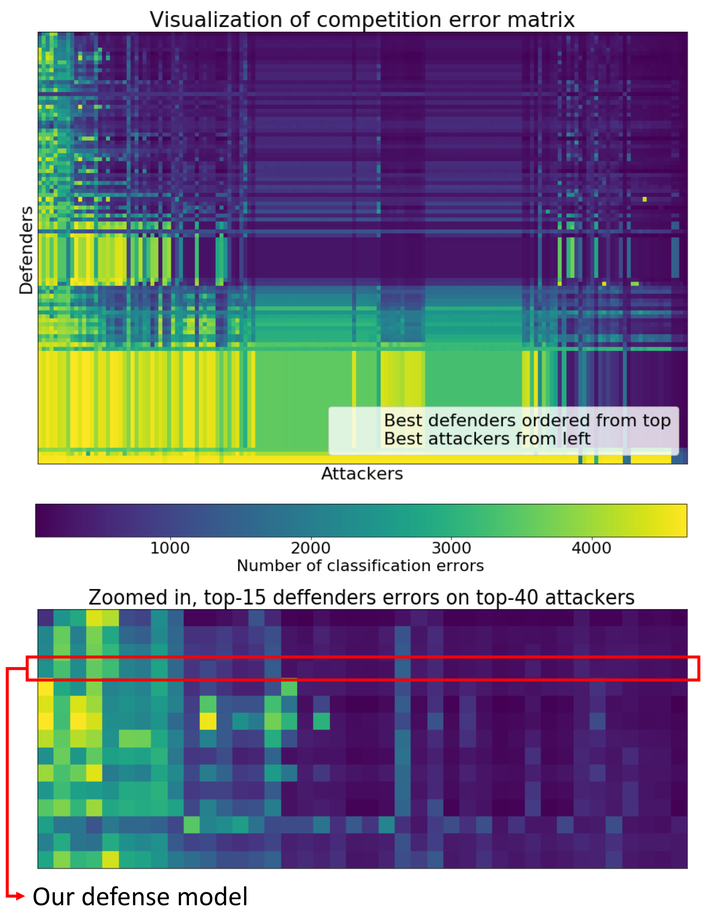
Competition constraints:

- evaluation time: max 500 seconds to batch of 100 images
- eps: 2, 4, 8, 16 (all values are tested)

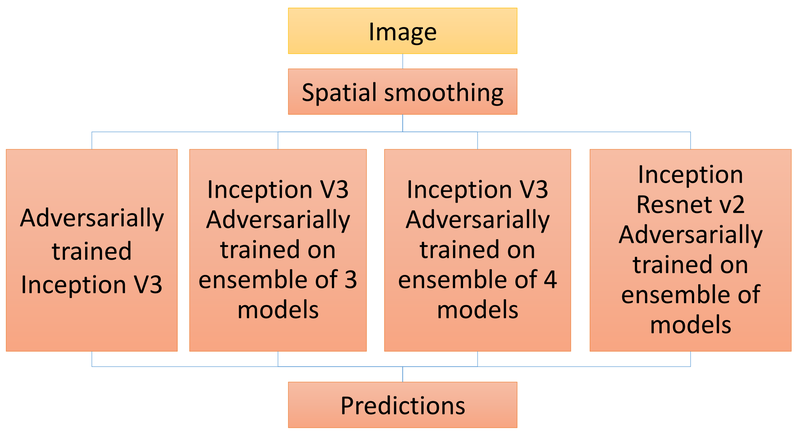
Timeline: Jul 4 – Oct 1, 2017

Scores are calculated by summing correct classifications counts.

Competition results



Final architecture of defense model



Spatial smoothing: median filtering.

Median filtering – often used in image/photo pre-processing to reduce noise while preserving edges and other features. Robust against salt-pepper like noise, random high-magnitude perturbations.

Experiments:

We have experimentally observed, using median filtering alone is not giving any defense against strong attacks like described by Carlini-Wagner (from now CW – is currently best algorithm generating strong attacks with about 100% ratio fooling samples on white-box attacks). However, simple model architecture of **using filtering with only adversarially trained models** shows very good defense to strong adversarial attacks. See Table 1 and Table 2.

In our experiments we used dataset given by competition and used modified CW L2 attack to generate adversarial samples. This samples later used to check adversarial samples misclassification ratio and to rank defenses. To generate adversarial samples used either single model or ensemble of models (list of multiple models in cell).

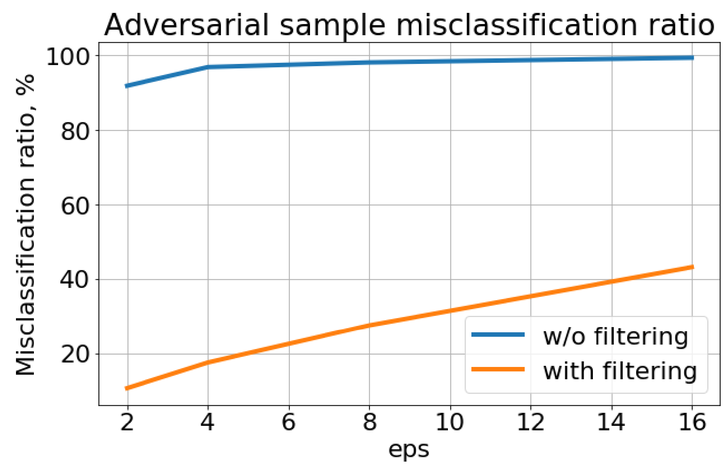
In any of our experiments “hold-out” InceptionV4 model not used to generate adversarial samples (orange cell in tables 1 & 2). This allowed us to test transferability of attacks and testing spatial smoothing effects.

Effects of median filtering.

On our hold-out inception_v4 model, with median filtering performs nearly same as without median filtering. Same results on other non-adversarially trained models. With median filtering or without, misclassification ratio differences are small.

Adversarially trained models with median filtering shows good defense against attacks. Ensemble of this adversarially trained models with median filtered inputs are robust against attacks, also to attacks generated by ensemble containing same models (green cell in tables 1 & 2).

All this attacks generated using eps=16 max pixel perturbations. In case of best ensemble defense against best ensemble attacker, we tested other values of epsilon and plotted next graph, showing that in case of lower eps values this defense approach is more robust against attacks:



Conclusion:

Following competition results we investigated that adversarially trained models with filtering are indeed robust to most types of attacks. We suggesting more research on this effect of adversarially trained models in future.

During competition, new types of attacks developed with smooth adversarial samples that can fool spatially smoothed defenses with as high as 50-60% ratio and with high transferability. Additional research needed on defending against this and new types of attacks.

Table 1. Misclassification ratio without filtering, %

Defenders \ Attackers	inception_v3	*	adv_inception_v3	ens3_adv_inception_v3	adv_inception_v3 ens3_adv_inception_v3 ens4_adv_inception_v3	adv_inception_v3 ens3_adv_inception_v3 ens_adv_inception_resnet_v2	**
inception_v3	100.00	100.00	13.75	21.25	22.50	26.25	99.38
inception_v4	42.50	80.63	11.25	16.25	19.38	21.88	62.50
adv_inception_v3	20.62	41.25	100.00	20.63	100.00	100.00	100.00
ens3_adv_inception_v3	15.62	38.13	16.88	100.00	99.38	100.00	99.38
ens_adv_inception_resnet_v2	10.62	23.75	8.75	8.75	10.63	94.38	95.00
inception_v3 inception_v4	99.38	100.00	9.38	13.75	16.25	18.13	98.13
inception_v3 inception_v4 adv_inception_v3	46.25	80.00	21.88	10.63	32.50	39.38	98.13
inception_v3 inception_v4 adv_inception_v3 ens3_adv_inception_v3	34.38	64.38	16.88	21.25	60.00	61.25	98.13
inception_v3 inception_v4 adv_inception_v3 ens3_adv_inception_v3 ens_adv_inception_resnet_v2	23.75	48.75	8.75	16.25	29.38	74.38	96.25
adv_inception_v3 ens3_adv_inception_v3	15.00	36.25	96.25	96.25	100.00	100.00	100.00
adv_inception_v3 ens3_adv_inception_v3 ens4_adv_inception_v3	16.25	33.13	35.00	29.38	100.00	100.00	99.38
adv_inception_v3 ens3_adv_inception_v3 ens_adv_inception_resnet_v2 ens4_adv_inception_v3	**12.50**	**28.75**	**17.50**	**20.63**	**99.38**	**100.00**	**99.38**

Ensembles of only adversarially trained models

* - inception_v3, inception_resnet_v2, resnet_v1_101, resnet_v1_50, resnet_v2_101, resnet_v2_50, vgg_16

** - inception_v3, adv_inception_v3, ens3_adv_inception_v3, ens_adv_inception_resnet_v2, ens4_adv_inception_v3, inception_resnet_v2, resnet_v1_101, resnet_v1_50, resnet_v2_101

Adversarially trained models are not robust against many kinds of attacks

Table 2. Misclassification ratio with filtering, %

Defenders \ Attackers	inception_v3	*	adv_inception_v3	ens3_adv_inception_v3	adv_inception_v3 ens3_adv_inception_v3 ens4_adv_inception_v3	adv_inception_v3 ens3_adv_inception_v3 ens_adv_inception_resnet_v2 ens4_adv_inception_v3	**
inception_v3	100.00	97.50	16.25	24.38	24.38	27.50	95.63
inception_v4	40.00	75.63	12.50	18.13	21.25	22.50	57.50
adv_inception_v3	21.88	43.13	32.50	17.50	31.88	33.13	40.00
ens3_adv_inception_v3	21.88	43.75	14.38	61.88	55.63	57.50	58.13
ens_adv_inception_resnet_v2	13.13	30.63	11.88	16.88	16.25	30.63	39.38
inception_v3 inception_v4	96.25	96.88	11.88	16.25	20.63	21.25	88.75
inception_v3 inception_v4 adv_inception_v3	43.75	76.88	14.38	15.00	21.25	24.38	62.50
inception_v3 inception_v4 adv_inception_v3 ens3_adv_inception_v3	33.13	62.50	13.13	21.25	27.50	30.63	59.38
inception_v3 inception_v4 adv_inception_v3 ens3_adv_inception_v3 ens_adv_inception_resnet_v2	25.63	52.50	10.63	16.88	21.88	29.38	53.75
adv_inception_v3 ens3_adv_inception_v3	17.50	40.00	20.63	38.13	40.63	43.75	47.50
adv_inception_v3 ens3_adv_inception_v3 ens4_adv_inception_v3	17.50	38.75	16.88	26.25	41.25	43.75	48.75
adv_inception_v3 ens3_adv_inception_v3 ens_adv_inception_resnet_v2 ens4_adv_inception_v3	**14.38**	**35.00**	**14.38**	**20.63**	**33.13**	**39.38**	**43.13**

Ensembles of only adversarially trained models

Median filtering is not “cleaning” or mitigating adversarial samples. On hold out inception_v4 model error ratio is same as in Table 1

Adversarially trained models with median filtering shows better robustness against many kinds of attacks

Table 3. Misclassification ratio on epsilon values, %

	ε=16	ε=8	ε=4	ε=2
ensemble of adversarial models non-filtered input	99.375	98.125	96.875	91.875
ensemble of adversarial models with filtered input	**43.125**	**27.5**	**17.5**	**10.625**

Source code published here: https://github.com/erko/nips17-defense

To see overview video scan QR-code