Deep Networks with RBF Layers to Prevent Adversarial Examples

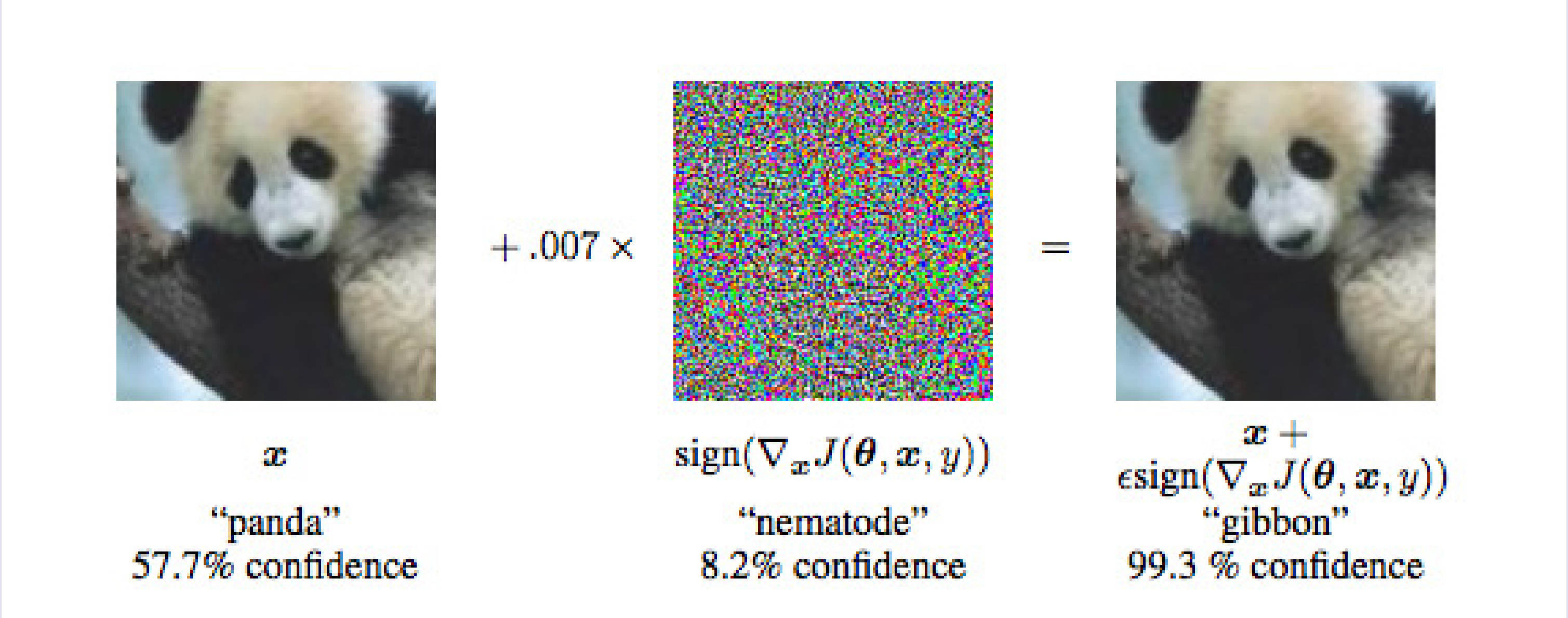
Petra Vidnerová and Roman Neruda
Institute of Computer Science, Czech Academy of Sciences, Prague

Introduction

We propose a simple way to increase the robustness of deep neural network models to adversarial examples. The new architecture obtained by stacking deep neural network and RBF network is proposed. It is shown on experiments that such architecture is much more robust to adversarial examples than the original one while its accuracy on legitimate data stays more or less the same.

Adversarial examples

Adversarial examples differ only slightly from correctly classified examples drawn from the data distribution, but they are classified incorrectly by the classifier learned on the data.

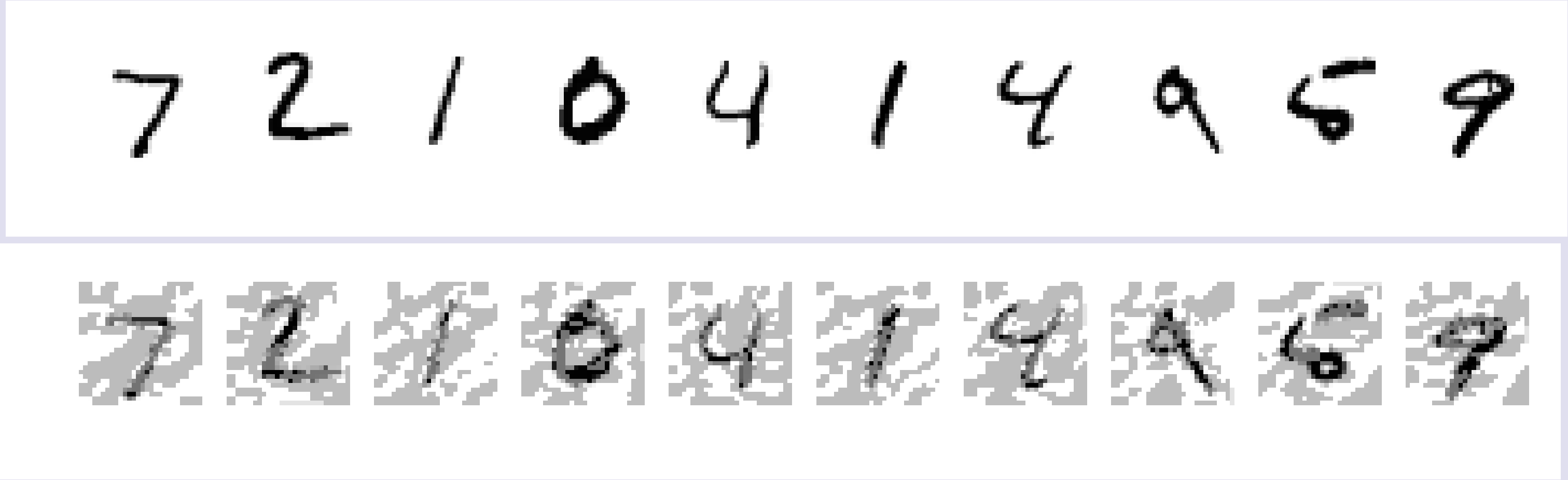

+ .007 ×
=
$\boldsymbol{x}$
"panda"
57.7% confidence
$\mathrm{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon\mathrm{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

(image taken from [3]).

Let θ be the parameters of a model, x an input, y the targets for x, and $J(\theta, x, y)$ the cost function. If we linearize the cost function around the θ, we obtain an optimal perturbation:

$$\eta = \varepsilon \ \mathrm{sgn}(\bigtriangledown_x J(\theta, x, y)).$$

This represents an efficient way of generating adversarial examples and it is referred to as the *fast gradient sign method* (FGSM) [3].

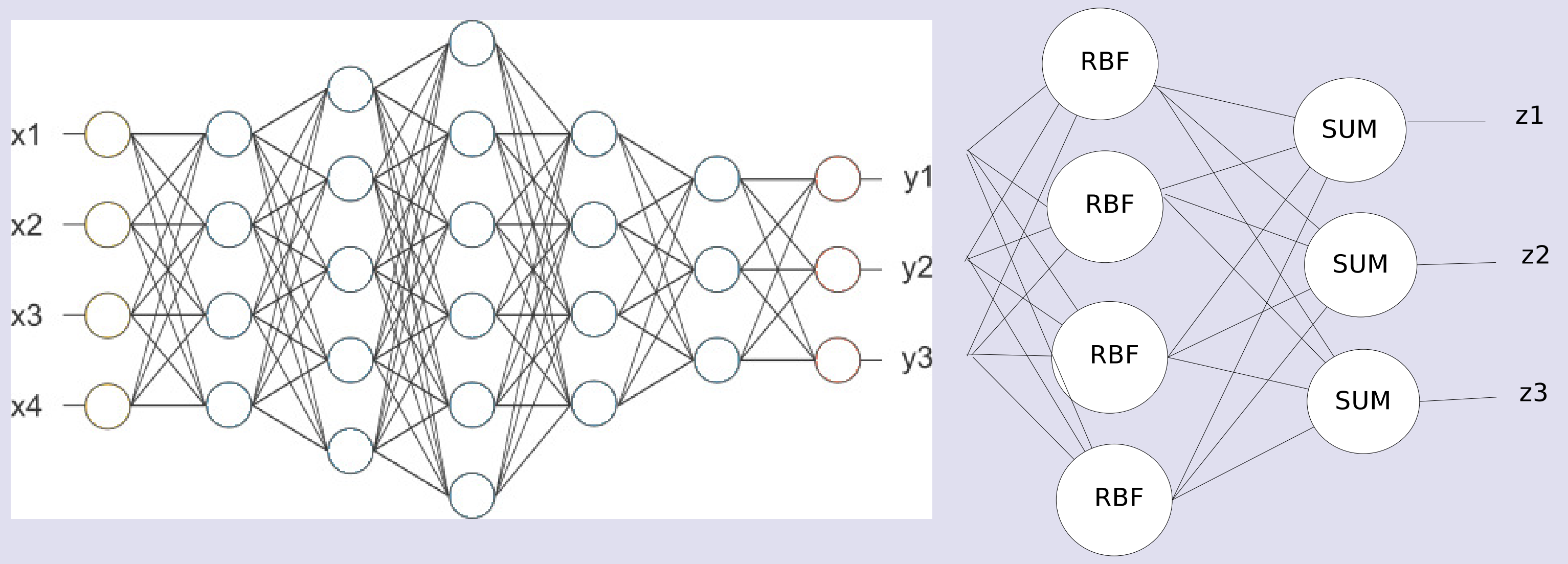


Original test examples and corresponding adversarial examples crafted by FGSM with ϵ 0.3.

Deep RBF Networks

We introduce new deep architecture that is defined as a concatenation of a feedforward deep neural network [1] and an RBF network [2].

The DNN is trained by any appropriate learning algorithm. The centers of RBF are set randomly, drawn from uniform distribution on $(0, 1.0)$. The widths β are set to the constant value. The output weights are initialized to random small values. The whole network is retrained by back propagation.



Experimental Results

For our experiments we use the FGSM implemented in Cleverhans library [4] and the MNIST data set. The scripts used for experiments can be found at [5].

We have two target architectures — MLP (two dense hidden layers with 512 ReLU units each, dense output layer of 10 softmax units) and CNN (two convolutional layers with 32 3x3 filters, ReLU activation, 2x2 max pooling layer, dense layer with 128 ReLU units, dense output layer of 10 softmax units).

These two architectures were trained 30 times by RMSProp for 20 and 12 epochs for MLP and CNN respectively.To each of the 30 trained networks we added the RBF network and retrained the whole new networks for 3 epochs.

We found that the results depend on the parameters β of the Gaussians, therefore we tried several initial setups. The best results were obtained with initial β 2.0, and on adversarial data they were 89.21 % for MLPRBF and 74.57 % for CNNRBF.

	Legitimate samples				Adversarial samples			
model	mean	std	min	max	mean	std	min	max
MLP	**98.35**	**0.12**	**98.04**	**98.59**	**1.95**	**0.41**	**1.30**	**2.86**
MLPRBF(0.01)	97.62	2.43	88.44	98.65	2.56	2.09	1.16	10.71
MLPRBF(0.1)	88.61	8.56	69.91	98.36	10.04	6.45	1.71	23.10
MLPRBF(1.0)	98.23	0.10	98.08	98.48	81.77	7.84	64.18	94.06
MLPRBF(2.0)	**98.19**	**0.14**	**97.91**	**98.38**	**89.21**	**5.03**	**66.28**	**94.83**
MLPRBF(3.0)	98.18	0.14	97.88	98.45	81.66	4.38	70.13	87.23
MLPRBF(5.0)	97.64	2.09	89.34	98.36	69.47	13.26	13.01	81.95
MLPRBF(10.0)	80.94	11.82	58.57	98.33	21.49	16.32	2.48	65.11
CNN	**98.97**	**0.07**	**98.84**	**99.13**	**8.49**	**3.52**	**3.11**	**16.43**
CNNRBF(0.01)	98.36	1.73	89.12	99.01	15.60	4.28	10.26	28.44
CNNRBF(0.1)	94.19	8.21	58.88	98.92	18.58	6.42	6.01	31.29
CNNRBF(1.0)	98.83	0.13	98.46	99.04	57.09	9.23	33.39	78.99
CNNRBF(2.0)	**98.85**	**0.13**	**98.38**	**99.09**	**74.57**	**7.69**	**53.07**	**84.67**
CNNRBF(3.0)	98.82	0.14	98.55	99.10	68.65	7.77	44.36	80.13
CNNRBF(5.0)	98.74	0.11	98.49	98.94	62.35	7.04	48.03	77.04
CNNRBF(10.0)	97.86	2.24	89.33	98.84	64.71	8.32	46.61	79.89

References

[1] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)

[2] Neruda, R., Kudová, P.: Learning methods for radial basis functions networks. Future Generation Computer Systems **21** (2005) 1131–1142

[3] Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples (2014) arXiv:1412.6572.

[4] Nicolas Papernot, e.a.: cleverhans v2.0.0: an adversarial machine learning library. arXiv preprint arXiv:1610.00768 (2017)

[5] Vidnerová, P.: Experiments with deep rbf networks. `github.com/PetraVidnerova/rbf_tests` (2017)

Acknowledgment

This work was partially supported by the Czech Grant Agency grant GA18-23827S and institutional support of the Institute of Computer Science RVO 67985807.
Access to computing and storage facilities owned by parties and projects contributing to the NGI MetaCentrum provided under the programme CESNET LM2015042, is greatly appreciated.