Bidirectional Learning for Robust Neural Networks

Sidney Pontes-Filho* and Marcus Liwicki†
*,†MindGarage, University of Kaiserslautern, Kaiserslautern, Germany
*Department of Computer Science, Oslo Metropolitan University, Oslo, Norway
*Department of Computer Science, Norwegian University of Science and Technology, Trondheim, Norway
†Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå, Sweden
Email: *sidneyp@oslomet.no, †marcus.liwicki@ltu.se

***Abstract*—A multilayer perceptron can behave as a generative classifier by applying bidirectional learning (BL). It consists of training an undirected neural network to map input to output and vice-versa; therefore it can produce a classifier in one direction, and a generator in the opposite direction for the same data. The learning process of BL tries to reproduce the neuroplasticity stated in Hebbian theory using only backward propagation of errors. In this paper, two novel learning techniques are introduced which use BL for improving robustness to white noise static and adversarial examples. The first method is *bidirectional propagation of errors*, which the error propagation occurs in backward and forward directions. Motivated by the fact that its generative model receives as input a constant vector per class, we introduce as a second method the *hybrid adversarial networks* (HAN). Its generative model receives a random vector as input and its training is based on generative adversarial networks (GAN). To assess the performance of BL, we perform experiments using several architectures with fully and convolutional layers, with and without bias. Experimental results show that both methods improve robustness to white noise static and adversarial examples, and even increase accuracy, but have different behavior depending on the architecture and task, being more beneficial to use the one or the other. Nevertheless, HAN using a convolutional architecture with batch normalization presents outstanding robustness, reaching state-of-the-art accuracy on adversarial examples of hand-written digits.**

***Index Terms*—adversarial example defense, noise defense, bidirectional learning, hybrid neural network, Hebbian theory**

I. Introduction

Deep neural networks present impressive performance in computer vision tasks, such as image classification and object detection [1], [2], but they are vulnerable to small designed perturbations and visually unrecognizable images giving high confidence predictions [3], [4]. This vulnerability can cause severe security issues. Just imagine a self-driving car controlled by these neural networks. Are they reliable? In computer vision literature, there are attacking methods for crafting small perturbations that are imperceptible by the human eye, but result in deep neural networks incorrectly identifying them with absolute certainty [4]. The images produced by those attacking methods are called *adversarial examples*. Other methods produce unrecognizable images for which deep neural networks give highly confident predictions [3].

The idea of bidirectional learning (BL) is to make the output layer of a discriminative neural network only active when real input data is given, like the behavior of a generative classifier. That is done by teaching the same model to learn how to "read" (discriminative) and "write" (generative). Because of that, an undirected neural network can be a classifier and a generator at the same time and thereby improve classifier's robustness to random noise and adversarial examples. Our goal is to make multilayer perceptrons behave as a generative classifier, such as radial basis function [5], [6], deep Bayes classifier [7], and many others. Generative classifiers were identified as robust to adversarial examples [7]. Weights and bias adaptation of multilayer perceptrons under BL is performed only by backward propagation of errors (backpropagation). Only real data is utilized for training the neural networks.

The main contribution of this paper is the introduction of two BL methods.[1] The first method, called bidirectional propagation of errors, trains a hybrid undirected neural network to map images to labels (classifier) and labels to images (generator) in the opposite direction. The second method replaces the training of its generator by using the framework of generative adversarial networks (GAN) introduced by Goodfellow et al. [8]. This leads to hybrid adversarial networks (HAN), where the generator that has as input a latent variable and is trained by an adversarial discriminator. The HAN classifier uses the transposed weights of the generator. Therefore it contains a hybrid model which merges the generator and classifier. To evaluate the performance of these two approaches, we perform experiments on many models for measuring accuracy on unmodified test data, test data with noise addition, and adversarial test data. We also assess the robustness of the models to white noise static by checking their rates of maximum output for noise data over real test data.

II. Related work

Bidirectional learning has similarities to deep belief networks (DBNs) [9] because they are also hybrid models. However, DBNs perform a pre-training phase with restricted Boltzmann machines (RBM) [9], [10] for an unsupervised input reconstruction layer-by-layer, from training data input layer to a final associative memory. Then an output layer for the discriminative model is added representing the ground truth, and backpropagation is executed for a fine-tuned classification training. Some autoencoder frameworks contain en-

Partially funded by Norwegian Research Council under SOCRATES project (grant number 270961).

[1]Complete project available at https://github.com/sidneyp/bidirectional.

coder and decoder sharing their weights for dimensionality reduction tasks. Such "mirrored" autoencoders are described in [11], [12]. There exist also deep hybrid models [13] where discriminative and generative models share the same latent variables. Another similar method, called Eigenboosting [14], which its authors present a generative classifier by its hybrid training with Harr-like features [15].

Since the discovery that deep neural networks for image classification can be easily fooled by random noise; unrecognizable images; and adversarial examples [3], [4], several defensive and attacking strategies were described in literature [16]–[18]. One way to make neural networks more robust is adversarial re-training [17], [19], [20]. It consists of generating adversarial examples every epoch or iteration, and using them as training data. Another defensive strategy is adding an auxiliary classifier for adversarial examples detection [20]–[22]. Since the creation of adversarial examples is by adding noise into real data, a denoising method can be useful. Therefore [23] applies denoising autoencoder before feeding the data into a classifier. There are defensive methods that use generative adversarial networks. Adversarial perturbation elimination with GAN (APE-GAN) uses the generator of GAN as a denoising autoencoder [24], [25]. Another method using GAN is the Generative Adversarial Trainer [24], [26] which the generator of GAN produces adversarial perturbations into the training set. These previous defensive methods use adversarial examples during training, so those neural networks can be biased to the method which designs the adversarial examples.

The method of network distillation increases the robustness without the need for adversarial examples in training set [20], [27]. Its idea is to train a neural network to behave as another trained neural network. Instead of giving hard labels to a neural network, the *temperature*-controlled softmax output of the trained neural network is given as ground truth. However, [28] verified that network distillation is still vulnerable to adversarial examples. A generative classifier presented as a defensive method to adversarial examples is called Gaussian process hybrid deep neural networks [20], [29]. The last layer of that robust convolutional neural network architecture consists of radial basis function kernels. Therefore it behaves as a generative classifier. The authors state that their deep architecture knows when it doesn't know. A biologically inspired defense against adversarial examples for deep neural networks is presented by [24], [30]. Its principle is the creation of highly nonlinear neural networks which produces a saturated weight distribution found in the brain. All these three previous methods do not use adversarial examples during training and that is also our goal in this work.

III. Bidirectional learning

Bidirectional learning produces a classifier and a generator in undirected neural network using backward propagation of errors in both directions. So each direction of this network has its own biases and the weights are shared. The idea is that the same positive weights of the last layer of a generator for producing white pixels can be the first layer of a classifier for identifying white pixels. Negative weights are similar regarding black pixels. Formally and over-simplified, any perceptron without bias that contains a weight vector $\mathbf{w} \in \{-1, 1\}, \exists w = 1$ and an input $\mathbf{x} \in \{0, 1\}$ which its output $\mathbf{y} = f(\mathbf{w} \cdot \mathbf{x})$, where f is the threshold activation function defined by

$$f(a) = \begin{cases} 0 & \text{if } a \leq 0 \\ 1 & \text{if } a > 0 \end{cases} \tag{1}$$

The perfect activation input $\hat{\mathbf{x}} = \operatorname{argmax}_{\mathbf{x}} \mathbf{w} \cdot \mathbf{x}$ must have active inputs for positive weights and inactive inputs for negative weights, therefore $\hat{\mathbf{x}} = max(\mathbf{w}, 0)$. It shows $\mathbf{w}$ can also be adapted to be a contrast template of $\hat{\mathbf{x}}$, so the perceptron becomes a generative classifier with that fast adaptation procedure by "copying" its input when an activation occurs, as the Hebbian theory states [31]. In biology, a real neuron produces a back-propagating action potential where the activation that goes through the axon back-propagates to its dendrites for plasticity regulation [32]. When the activation output of $\hat{x}$ is back-propagated, the result is equal to itself and expressed by

$$\hat{\mathbf{x}} \equiv f(\mathbf{w}^T \cdot f(\mathbf{w} \cdot \hat{\mathbf{x}})). \tag{2}$$

When different activation functions and multilayers are used, 2 becomes an approximation. So bidirectional learning forces equivalence in these cases. We infer that adding a supporting backpropagation to a classifier in the opposite direction that it is normally used can make the classifier's outputs less active when non-real data are given as input and avoid the vulnerability to adversarial examples. Since biological neurons learn by inputs, bidirectional learning uses a common training algorithm of artificial neural networks for trying to mimic Hebbian learning to the excitatory synapses (positive weights), because "neurons wire together if they fire together" [33]; and for anti-Hebbian learning [34] to the inhibitory ones, because negative weights are strengthened when classifier's inputs remain inactive.

A. Bidirectional propagation of errors

Our supervised learning approach for both directions of a hybrid undirected neural network is the bidirectional propagation of errors. It consists of using backward propagation of errors (backpropagation) for mapping data to ground truth, and then ground truth to data. The mapping order can be reverted. The same batch of pairs of data and labels is utilized in normal and reversed backpropagation in a training iteration. Fig. 1 shows how it works.

B. Hybrid adversarial networks

The previous method explained in Section III-A has a limitation because the generator is trained with constant input per class. To avoid that, the hybrid adversarial networks (HAN) are introduced. There are three models in this framework based on GAN [8]: classifier C, generator G that shares the same weights of C, and discriminator D for being an adversary to G. The input of G is a random vector z of size equal to the

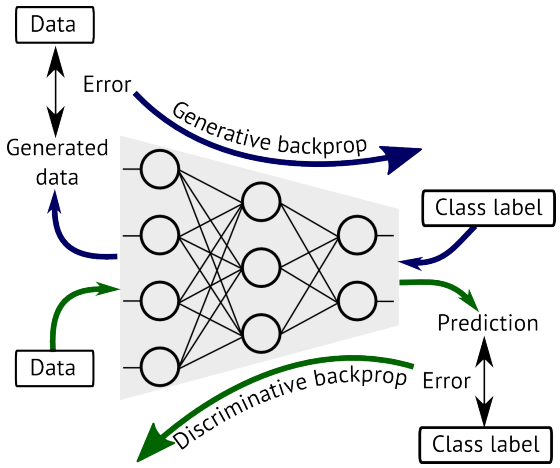


Fig. 1. Illustration of one training iteration in bidirectional propagation of errors. Dark green arrows represent the training with backpropagation (backprop) of discriminative model. Dark blue arrows represent the training of generative model. Same data and class labels are used for both in an iteration.

number of classes for C. While C is trained normally, D and G compete in a minimax game where G tries to reproduce the real data to increase the error of D, and D learns how to distinguish real data and data from G.

Our hybrid model merges G and C, so the generator of GAN can be trained simultaneously as a transposed classifier for more robustness because we infer it can produce neurons that become active when images look "realistic". Fig. 2 presents this framework. The training order in an iteration is C, then D, finishing with G.

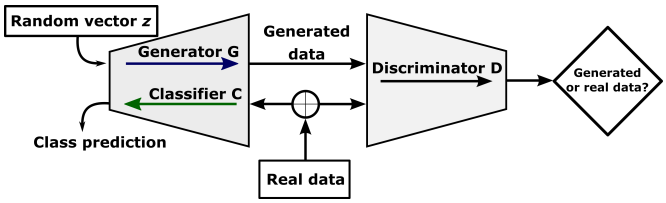


Fig. 2. Illustration of hybrid adversarial networks. Same color scheme is used in the hybrid model. Dark green arrow represents the discriminative model (classifier). Dark blue arrow represents the generative model.

IV. Experiments

These two methods were evaluated using the architectures with and without bias described in Table I. The architectures without bias are introduced to force all neurons in the network to have the same likelihood of activation and thereby increasing robustness. Architectures have been trained by mini-batch gradient descent with Adam optimizer [36] and mini-batch size of 100 data samples with ground truth (one mini-batch means one iteration). Bidirectional propagation of errors was trained with 50,000 iterations and HAN with 500,000 iterations because the adversarial training in HAN takes more time to converge. The implementation is based on TensorFlow 1.7 [37]. The datasets used in the experiments are MNIST [38] and CIFAR-10 [39]. Training set consists of 60,000 samples and test set of 10,000 samples. The adversarial attacking method to test the robustness of bidirectional learning is the fast gradient sign method (FGSM) [17], [18]. It disturbs real images to fool the classifier to make predictions for wrong classes. The equation for disturbing an image x is

$$x_{adv} = x + \epsilon * sign(\nabla_x J_\theta(x, y)), \tag{3}$$

where the adversarial image x_{adv} is produced by adding to the normal image x with the sign method result of the gradient ascent ∇ for x of the loss function J for model θ when image x and label y are given. This addition is limited by ϵ which is the maximum change in the pixels of x. The method which we use is from CleverHans v2.0.0 [18]. The testing images were modified by FGSM with a max-norm epsilon (ϵ) of 0.3 for MNIST, and 0.03 for CIFAR-10. Minimum and maximum pixel values of disturbed images are 0 and 1, respectively. We tested the robustness to white noise static by adding 10 % of it into the test set for accuracy verification, and giving 100% of that noise as classifier's input for measuring the sigmoid [40] and softmax [41] output layer. The maximum output for random noise x_{noise} is divided by the maximum output for real test data x_{test}. Both x_{noise} and x_{test} have the same shape. That gives a rate of outputs to white noise static over real data. The sigmoid rate is for measuring output layer activity and expressed by

$$r_{sigmoid} = \frac{\max(C_{sigmoid}(x_{noise}))}{\max(C_{sigmoid}(x_{test}))}. \tag{4}$$

The softmax rate for classification probability and formally denoted as

$$r_{softmax} = \frac{\max(C_{softmax}(x_{noise}))}{\max(C_{softmax}(x_{test}))}. \tag{5}$$

All architectures with and without biases were trained by:

1) Backpropagation (BP)
2) Bidirectional learning on first half of iterations, then backpropagation (BL then BP)
3) Bidirectional learning (BL)

V. Results

This section presents the results of our two methods on MNIST and CIFAR-10 dataset. It contains the accuracy for real test data, for test data with noise addition, and for test data modified by FGSM. The desired accuracy is 1.0 or 100 %. We measure robustness to white noise static with sigmoid (activity) and softmax (class probability) rate of noise over real test data. The desired sigmoid output for noise data is 0.0 (fully inactive) and for test data is 1.0 (fully active). Therefore, the desired sigmoid rate is 0.0. Since we use datasets with ten classes, the desired softmax output for noise data is 0.1 or 10 % confidence, and for test data is 1.0 or 100 %. It means a softmax rate of 0.1.

TABLE I

DESCRIPTION OF ARCHITECTURES USED ON TWO METHODS WITH AND WITHOUT BIAS. NN STANDS FOR FULLY CONNECTED NEURAL NETWORK AND CNN FOR CONVOLUTIONAL NEURAL NETWORK. CONVOLUTIONAL LAYERS ARE DESCRIBED BY THE NUMBER OF KERNELS, INSIDE THE PARENTHESIS IS THE KERNEL SIZE AND ITS STRIDE (STR).

Method	Architecture	Units in discriminative hidden layer
Bidirectional propagation of errors	NN no hidden layer	-
	NN one hidden layer	16
	NN two hidden layers	16,16
	NN four hidden layers	200,100,60,30
	CNN three conv. layers	4 (5x5str1), 8(5x5str2),12(4x4str2),200
Hybrid generative nets	NN one hidden layer	128
	CNN two conv. layers	infoGAN architecture for MNIST [35]

A. Results of bidirectional propagation of errors

Table II shows in the first row the architecture with most relative improvement in accuracy on adversarial examples. It is the architecture without hidden layer and bias, then it is a linear classifier. Backpropagation presents accuracy on adversarial examples of 4.17 %, while bidirectional learning shows 60.14 %. Since this is a simple architecture, the learned weights are easy to understand and to verify the causes of difference in robustness. Fig. 3a shows that for MNIST dataset including adversarial examples and generated images for each class. The second row of Table II shows the best result regarding robustness to white noise static measured by the sigmoid and softmax rate of maximum output for noise over test data. The learning method that reached that was BL. The value of sigmoid rate is 0.5 which is a low value for sigmoid, meaning that the input for this activation function was zero. The value of softmax rate was the best value possible, 0.1 or 10 %.

Table III shows the results of bidirectional propagation of errors for CIFAR-10 dataset. The first row contains the best relative accuracy improvement on adversarial examples. It is reached by the architecture trained by BL. Its weights, adversarial examples and generated images of all three learning methods are in Fig. 3b. The order of CIFAR-10 classes is: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The weights of BP are noisy representations of those classes, but when we check the weights learned by BL, they are smooth and recognizable. We can see, for example, the weights of the blue color channel with high values for representing the sky and sea in airplane and ship classes. The second architecture in Table III presents an increase of 2.25 % accuracy on test data when trained partially by BL compared with only BP. The architecture with two hidden layers and no bias is not shown here, but the training with BL then BP increases the accuracy on normal test data as well when compared to the results with BP.

B. Results of hybrid adversarial networks

Table IV shows the best robustness of all experiments performed in this work. Hybrid adversarial networks on the architecture of infoGAN for MNIST [35] achieved that. The model trained by BL then BP and without biases reached the 95.92 % on adversarial examples of MNIST test set, while BP presented 5.08 %. Even though there was a small reduction of accuracy on real test set compared with BP, from 99.21 % to 98.49 %. Fig. 4a shows the adversarial examples for this architecture and the images generated by a random vector. That presents that HAN can also be a generative method when trained with BL.

Table V is for the results of HAN on CIFAR-10 dataset. These results are not as good as the ones of HAN on MNIST dataset. The reason is that accuracy on real test set reduced drastically while presenting small improvement in robustness. However, Fig. 4 presents that generator of HAN trained by BL has recovered the data distribution of CIFAR-10. Even though a generative model was not our goal.

VI. ANALYSIS AND DISCUSSION

We can see in Fig. 3a that backpropagation in the architecture without layer and no bias presents a noisy representation for each digit of MNIST dataset. Disturbances can be manually designed to this neural network; for example, the weights for number two have high positive values in the right. These positive weights in this part do not represent the most important white pixels in images with number two, but backpropagation identifies that part as the most relevant white pixels to recognize an image as number two. When increasing the pixel values in that part of an image with a different class, the resulting disturbed image can be recognized as number two. The adversarial examples generated by FGSM show that as well, and how noisy adversarial examples can be. Partial bidirectional learning (BL then BP) is similar to backpropagation, the only difference is that it knows that the pixels of the border are black because of high negative weights in that region. On the other hand, bidirectional learning performed in all iterations increased the robustness of this neural network. We can easily see the reasons in the learned weights, the adversarial examples, and the images generated by the label of each class. The creation of adversarial examples for BL became harder. FGSM tries in some of the test images to draw another number for fooling the neural network trained by BL.

By analysis of these characteristics, we infer the biological function of neural networks is not of a fine-tuned universal function approximator [42], but it is of a multilayer contrast matching algorithm like a multilayer of Harr-like features from Viola-Jones object detection framework [15]. The reason is that more robust weights present the characteristics of Harr-like features and that explains how real neurons learn so fast

TABLE II
MOST SIGNIFICANT RESULT OF BIDIRECTIONAL PROPAGATION OF ERRORS ON MNIST. SELECTED ITERATION WITH BEST ACCURACY TEST. BOLD NUMBERS ARE THE BEST RESULTS FOR EACH MODEL.

Model	Learning	Accuracy test	Accuracy noisy	Accuracy adversarial	Sigmoid rate	Softmax rate
Fully connected	BP	**0.9273**	**0.7138**	0.0417	3.34E-12	1
no hidden	BL then BP	0.9265	0.3216	0.045	**0**	1
layer & no bias	BL	0.8781	0.6419	**0.6014**	**0**	1
Fully connected	BP	**0.9456**	**0.6502**	0.0318	0.9983	0.984
one hidden	BL then BP	0.9338	0.3807	0.06	0.9923	0.6429
layer & no bias	BL	0.905	0.5148	**0.0814**	**0.5**	**0.1**

TABLE III
MOST SIGNIFICANT RESULT OF BIDIRECTIONAL PROPAGATION OF ERRORS ON CIFAR-10. SELECTED ITERATION WITH BEST ACCURACY TEST. BOLD NUMBERS ARE THE BEST RESULTS FOR EACH MODEL.

Model	Learning	Accuracy test	Accuracy noisy	Accuracy adversarial	Sigmoid rate	Softmax rate
Fully connected	BP	**0.3769**	**0.373**	0.1853	0.9999	0.996
no hidden	BL then BP	0.374	0.3678	0.1882	**0**	**0.9725**
layer & no bias	BL	0.3211	0.3203	**0.2711**	**0**	0.9999
Fully connected	BP	0.4208	0.4137	0.351	**0.9791**	0.8627
four hidden	BL then BP	**0.4433**	**0.4334**	**0.3658**	0.9911	0.8359
layers & no bias	BL	0.4314	0.4283	0.3596	0.9807	**0.8289**

TABLE IV
MOST SIGNIFICANT RESULT OF HYBRID ADVERSARIAL NETWORKS ON MNIST. SELECTED ITERATION WITH BEST ACCURACY TEST. BOLD NUMBERS ARE THE BEST RESULTS FOR EACH MODEL.

Model	Learning	Accuracy test	Accuracy noisy	Accuracy adversarial	Sigmoid rate	Softmax rate
CNN	BP	**0.9925**	**0.9913**	0.0477	1	1
two conv.	BL then BP	0.9854	0.9783	**0.9375**	1	1
layers	BL	0.9823	0.9696	0.9084	1	1
CNN	BP	**0.9921**	**0.9906**	0.0508	1	1
two conv.	BL then BP	0.9849	0.9768	**0.9592**	1	1
layers & no bias	BL	0.9829	0.9491	0.9566	1	1

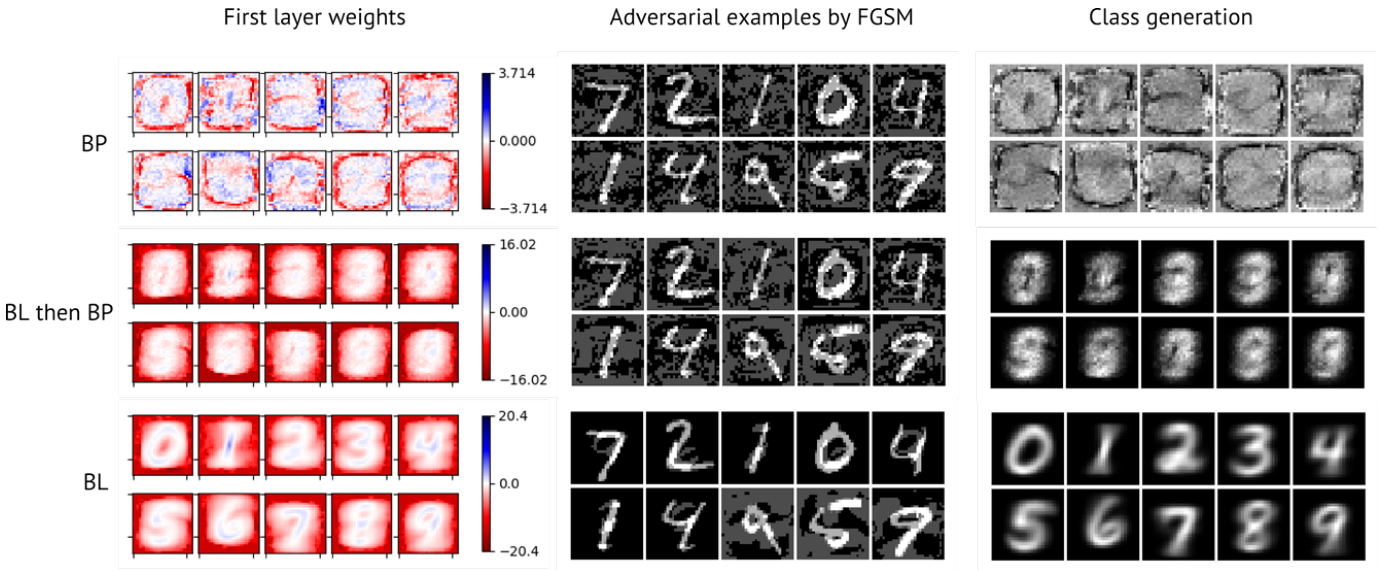
TABLE V
MOST SIGNIFICANT RESULT OF HYBRID ADVERSARIAL NETWORKS ON CIFAR-10. SELECTED ITERATION WITH BEST ACCURACY TEST. BOLD NUMBERS ARE THE BEST RESULTS FOR EACH MODEL.

Model	Learning	Accuracy test	Accuracy noisy	Accuracy adversarial	Sigmoid rate	Softmax rate
CNN	BP	**0.7101**	**0.6973**	0.161	1	1
two conv.	BL then BP	0.574	0.5645	**0.258**	1	1
layers	BL	0.565	0.5429	0.2445	1	1
CNN	BP	**0.7134**	**0.7067**	0.1733	1	1
two conv.	BL then BP	0.5419	0.531	**0.3366**	1	1
layers & no bias	BL	0.4264	0.4114	0.1981	**3.61E-07**	**0.9014**

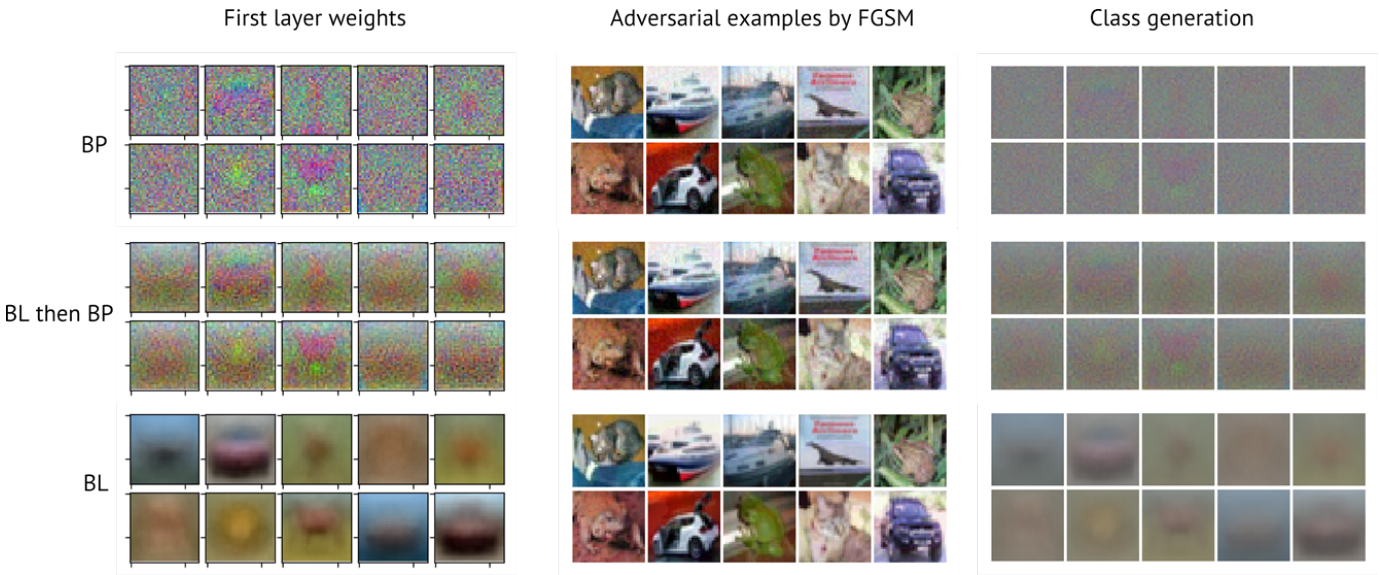
new complex patterns, just by "copying" the contrast of input that produces activation. The neurons of the primary visual cortex, from the retina through the lateral geniculate nucleus of the thalamus to V1 visual cortex [43], have the attributes for contrast detection of the weights trained by bidirectional learning or of the Harr-like features. The activation of a neuron depends on inputs with negative weights remaining inactive and inputs with positive weights being active. That also gives some light to the functionality of Hebbian and anti-Hebbian learning. The exclusion of bias makes neural networks more robust since it reduces the difference of neurons for activation likelihood. Therefore it tries to maintain neurons with equal importance in the network. Results on CIFAR-10 dataset show there are some architectures that when trained with full or partial BL can increase the accuracy on normal test data. Batch normalization also works to balance the neurons by keeping their inputs for activation function closer to zero. HAN results of infoGAN architecture without bias on MNIST dataset supports our analysis for equality in neuron importance and that a hybrid undirected neural network can be robust to adversarial examples.

VII. CONCLUSION AND FUTURE WORK

Bidirectional learning produces a classifier and a generator in an undirected neural network, giving benefits to the classification task which is our main goal; moreover, it can also support generation of images too. Producing supporting



(a) MNIST



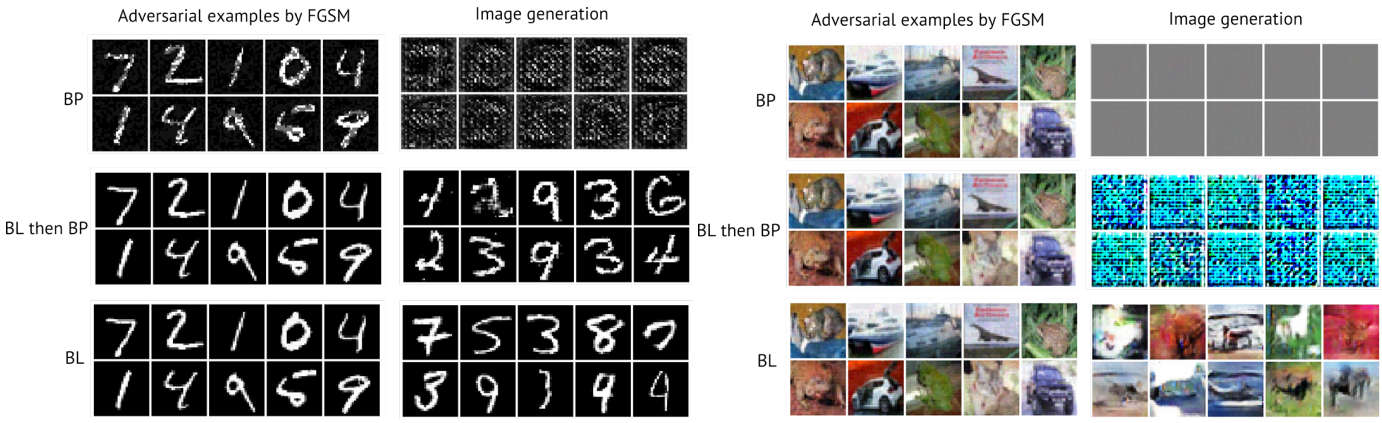
(b) CIFAR-10

Fig. 3. Weights of the first layer, generated adversarial examples and images generated by a class label in bidirectional propagation of errors with a fully connected architecture without hidden layer and bias in all three learning methods on each row.

methods and alternatives to backpropagation algorithm regarding robustness is essential for a reliable neural network. The defensive and learning method proposed in this paper was created by only adding a generative backpropagation in a discriminative multilayer perceptron. However, the difference of results on MNIST and CIFAR-10 dataset and on different architectures should be investigated.

For future work, we list the following advances possible after the proposal of bidirectional learning:

- application of BL on different datasets and architectures;
- the generator of bidirectional propagation of errors receiving as an input the label and the image together, then giving some variation to the generator's input and because of that it becomes an autoencoder;
- hybrid adversarial networks framework can be improved like its first version for generation [8] because several improvements to GAN appeared since its introduction and they can be applied to extend HAN as well, but for classification purposes;
- the decoder (generator) of an autoencoder as a transposed classifier;
- weight decay can improve accuracy for data with white noise static and mimic non-Hebbian learning for positive weights and Hebbian learning for negative weights, because they can reduce weight of connections with, respectively, constantly active and inactive inputs, since



(a) MNIST (b) CIFAR-10

Fig. 4. Generated adversarial examples and images generated by latent variable of hybrid adversarial networks in a CNN with two convolutional layers.

constant inputs are meaningless for neurons like random inputs;

- HAN can be verified as a generative method;
- other tasks can be performed by coding input or desired output as images or binary strings;
- alternatives to backward propagation of errors can be verified by analysis of BL behavior.

ACKNOWLEDGMENTS

We are grateful for the support of Stefano Nichele in the review process and for the suggestions of all reviewers. We thank Benjamin Grewe for helpful discussions. We also thank Grant Sanderson of 3Blue1Brown channel on Youtube. His videos helped the main author of this work to come with bidirectional learning idea. Finally, we would like to thank Insiders Technologies GmbH for providing us with the necessary computational resources.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 1097–1105. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999134.2999257

[2] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2553–2561. [Online]. Available: http://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detection.pdf

[3] A. M. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," *CoRR*, vol. abs/1412.1897, 2014. [Online]. Available: http://arxiv.org/abs/1412.1897

[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *CoRR*, vol. abs/1312.6199, 2013. [Online]. Available: http://arxiv.org/abs/1312.6199

[5] D. S. Broomhead and D. Lowe, "Radial basis functions, multi-variable functional interpolation and adaptive networks," Royal Signals and Radar Establishment Malvern (United Kingdom), Tech. Rep., 1988.

[6] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, no. 2, pp. 246–257, 1991. [Online]. Available: https://doi.org/10.1162/neco.1991.3.2.246

[7] Y. Li, "Are generative classifiers more robust to adversarial attacks?" *CoRR*, vol. abs/1802.06552, 2018. [Online]. Available: http://arxiv.org/abs/1802.06552

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[9] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1162/neco.2006.18.7.1527

[10] P. Smolensky, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds. Cambridge, MA, USA: MIT Press, 1986, ch. Information Processing in Dynamical Systems: Foundations of Harmony Theory, pp. 194–281. [Online]. Available: http://dl.acm.org/citation.cfm?id=104279.104290

[11] L. Xu, "Least mean square error reconstruction principle for self-organizing neural-nets," *Neural networks*, vol. 6, no. 5, pp. 627–648, 1993.

[12] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[13] V. Kuleshov and S. Ermon, "Deep hybrid models: Bridging discriminative and generative approaches." UAI, 2017.

[14] H. Grabner, P. M. Roth, and H. Bischof, "Eigenboosting: Combining discriminative and generative information," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[15] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[16] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks," *CoRR*, vol. abs/1608.04644, 2016. [Online]. Available: http://arxiv.org/abs/1608.04644

[17] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[18] N. Papernot, N. Carlini, I. Goodfellow, R. Feinman, F. Faghri, A. Matyasko, K. Hambardzumyan, Y.-L. Juang, A. Kurakin, R. Sheatsley *et al.*, "cleverhans v2. 0.0: an adversarial machine learning library," *arXiv preprint arXiv:1610.00768*, 2016.

[19] R. Huang, B. Xu, D. Schuurmans, and C. Szepesvári, "Learning with a strong adversary," *CoRR*, vol. abs/1511.03034, 2015. [Online]. Available: http://arxiv.org/abs/1511.03034

[20] X. Yuan, P. He, Q. Zhu, R. R. Bhat, and X. Li, "Adversarial examples:

Attacks and defenses for deep learning," *CoRR*, vol. abs/1712.07107, 2017. [Online]. Available: http://arxiv.org/abs/1712.07107

[21] J. Lu, T. Issaranon, and D. A. Forsyth, "Safetynet: Detecting and rejecting adversarial examples robustly," *CoRR*, vol. abs/1704.00103, 2017. [Online]. Available: http://arxiv.org/abs/1704.00103

[22] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," in *Proceedings of 5th International Conference on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1702.04267

[23] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *CoRR*, vol. abs/1412.5068, 2014. [Online]. Available: http://arxiv.org/abs/1412.5068

[24] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: A survey," *CoRR*, vol. abs/1801.00553, 2018. [Online]. Available: http://arxiv.org/abs/1801.00553

[25] S. Shen, G. Jin, K. Gao, and Y. Zhang, "AE-GAN: adversarial eliminating with GAN," *CoRR*, vol. abs/1707.05474, 2017. [Online]. Available: http://arxiv.org/abs/1707.05474

[26] H. Lee, S. Han, and J. Lee, "Generative adversarial trainer: Defense to adversarial perturbations with GAN," *CoRR*, vol. abs/1705.03387, 2017. [Online]. Available: http://arxiv.org/abs/1705.03387

[27] N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," *CoRR*, vol. abs/1511.04508, 2015. [Online]. Available: http://arxiv.org/abs/1511.04508

[28] N. Carlini and D. A. Wagner, "Defensive distillation is not robust to adversarial examples," *CoRR*, vol. abs/1607.04311, 2016. [Online]. Available: http://arxiv.org/abs/1607.04311

[29] J. Bradshaw, A. G. d. G. Matthews, and Z. Ghahramani, "Adversarial Examples, Uncertainty, and Transfer Testing Robustness in Gaussian Process Hybrid Deep Networks," *ArXiv e-prints*, Jul. 2017.

[30] A. Nayebi and S. Ganguli, "Biologically inspired protection of deep networks from adversarial attacks," *ArXiv e-prints*, Mar. 2017.

[31] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. New York: Wiley, Jun. 1949.

[32] B. Grewe, A. Bonnan, and A. Frick, "Back-propagation of physiological action potential output in dendrites of slender-tufted l5a pyramidal neurons," *Frontiers in Cellular Neuroscience*, vol. 4, p. 13, 2010. [Online]. Available: https://www.frontiersin.org/article/10.3389/fncel.2010.00013

[33] S. Lowel and W. Singer, "Selection of intrinsic horizontal connections in the visual cortex by correlated neuronal activity," *Science*, vol. 255, no. 5041, pp. 209–212, 1992. [Online]. Available: http://science.sciencemag.org/content/255/5041/209

[34] T. P. Vogels, R. C. Froemke, N. Doyon, M. Gilson, J. S. Haas, R. Liu, A. Maffei, P. Miller, C. J. Wierenga, M. A. Woodin, F. Zenke, and H. Sprekeler, "Inhibitory synaptic plasticity: spike timing-dependence and putative network function," *Front Neural Circuits*, vol. 7, p. 119, Jul 2013, 23882186[pmid]. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3714539/

[35] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," *CoRR*, vol. abs/1606.03657, 2016. [Online]. Available: http://arxiv.org/abs/1606.03657

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[37] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, vol. abs/1603.04467, 2016. [Online]. Available: http://arxiv.org/abs/1603.04467

[38] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

[39] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[40] J. Han and C. Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *Proceedings of the International Workshop on Artificial Neural Networks: From Natural to Artificial Neural Computation*, ser. IWANN '96. London, UK, UK: Springer-Verlag, 1995, pp. 195–201. [Online]. Available: http://dl.acm.org/citation.cfm?id=646366.689307

[41] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998. [Online]. Available: http://www.cs.ualberta.ca/~sutton/book/the-book.html

[42] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359 – 366, 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0893608089900208

[43] R. C. O'Reilly, Y. Munakata, M. J. Frank, T. E. Hazy, and Contributors, *Computational Cognitive Neuroscience*. Wiki Book, 1st Edition, URL: http://ccnbook.colorado.edu, 2012. [Online]. Available: http://ccnbook.colorado.edu