Towards Robust Deep Neural Networks with BANG

Andras Rozsa, Manuel Günther, and Terrance E. Boult
Vision and Security Technology (VAST) Lab
University of Colorado, Colorado Springs, USA
{arozsa, mgunther, tboult}@vast.uccs.edu

Abstract

Machine learning models, including state-of-the-art deep neural networks, are vulnerable to small perturbations that cause unexpected classification errors. This unexpected lack of robustness raises fundamental questions about their generalization properties and poses a serious concern for practical deployments. As such perturbations can remain imperceptible – commonly called adversarial examples that demonstrate an inherent inconsistency between vulnerable machine learning models and human perception – some prior work casts this problem as a security issue as well. Despite the significance of the discovered instabilities and ensuing research, their cause is not well understood, and no effective method has been developed to address the problem highlighted by adversarial examples. In this paper, we present a novel theory to explain why this unpleasant phenomenon exists in deep neural networks. Based on that theory, we introduce a simple, efficient and effective training approach, Batch Adjusted Network Gradients (BANG), which significantly improves the robustness of machine learning models. While the BANG technique does not rely on any form of data augmentation or the application of adversarial images for training, the resultant classifiers are more resistant to adversarial perturbations while maintaining or even enhancing classification performance overall.

1. Introduction

Machine learning is broadly used in various real-world vision applications and recent advances in deep learning have made deep neural networks the most powerful learning models that can be successfully applied to different vision problems [23, 21, 6, 24, 17, 11, 15, 14, 25]. The recent performance gain is largely the result of improvements in two fields, namely, building more powerful learning models [21, 6] and designing better strategies to avoid overfitting [20], which are then leveraged by larger datasets and massive GPU-enhanced computing.

Although deep neural networks (DNNs) achieve state-of-the-art performance in a wide range of tasks, the generalization properties of these learning models were questioned by Szegedy et al. [22] when the existence of adversarial examples was revealed. DNNs are capable of learning high-level feature embeddings that enable them to be successfully adapted to different problems and were generally considered to generalize well and, hence, expected to be robust to moderate distortions to their inputs. Surprisingly, adversarial examples formed by applying impercep-
tible perturbations to otherwise correctly recognized inputs can lead machine learning models – including state-of-the-art DNNs – to misclassify those samples, often with high confidence. This highly unexpected and intriguing property of machine learning models highlights a fundamental problem that researchers have been trying to solve.

To explain why adversarial examples exist, several, sometimes controversial explanations were proposed. As hypothesized in [3][2], adversarial instability exists due to DNNs acting as high-dimensional linear classifiers that allow even imperceptibly small, well-aligned perturbations applied to inputs to spread among higher dimensions and radically change the outputs of DNNs. This belief was challenged in [16], where by analyzing and experimenting with DNNs trained to recognize objects in more unconstrained conditions, it was demonstrated that those classifiers are only locally linear to changes on the recognized object, otherwise DNNs act non-linearly. After performing various experiments, Gu et al. [5] concluded that adversarial instability is more related to “intrinsic deficiencies in the training procedure and objective function than to model topology.”

The problem addressed in this paper is not only about preventing attacks utilizing adversarial examples, it is a question of the overall robustness and generalizability of DNNs which is a more fundamental problem of deep learning. Considering state-of-the-art learning models applied to computer vision tasks, the classification of many incorrectly or highly uncertainly recognized inputs can be corrected and improved by a small perturbation [26][18], so this is a naturally occurring problem for learning-based vision systems.

In this paper, we introduce our novel theory on the instability of machine learning models and the existence of adversarial examples: evolutionary stalling. During training, network weights are adjusted using the gradient of loss, evolving to eventually classify examples correctly. Ideally, one wants broad flat regions around training samples, to achieve good generalization [9] and adversarial robustness [2]. Unfortunately, after a training sample is correctly classified, its contribution to the loss function, thus, its contribution on forming the weight updates is reduced. As the evolution of the local decision surface stalls, the correct training samples cannot further flatten and extend their surrounding regions to improve generalization. Therefore, as the contributions of those correctly classified training samples to boundary adjustments are highly decreased, they can end up being stuck close to the decision boundary and, hence, susceptible to small perturbations flipping their classifications. To counteract the problem of evolutionary stalling, we propose the novel Batch Adjusted Network Gradients (BANG) training algorithm. We experimentally evaluate robustness using a combination of gradient and non-gradient based adversarial perturbations, and random distortions. The paper further explores the impact of BANG parameters and architectural variations, such as Dropout [20], on instability and adversarial robustness. In conclusion, we validate our theory by experimentally demonstrating that BANG significantly improves the robustness of deep neural networks while the trained learning models still maintain or even improve their overall classification performance.

2. Related Work

Deep neural networks (DNNs) achieve high performance on various tasks as they are able to learn non-local generalization priors from training data. Counter-intuitively, Szegedy et al. [22] showed that machine learning models – including DNNs – can confidently misclassify samples that are formed by slightly perturbing correctly recognized inputs. These so-called adversarial examples are indistinguishable from their originating counterparts to human observers, and their unexpected existence itself presents a problem. The authors introduced the first technique that is capable of reliably finding adversarial perturbations, and claimed that some adversarial examples generalize across different learning models.

A simpler and computationally cheaper algorithm, the Fast Gradient Sign (FGS) adversarial example generation method was presented by Goodfellow et al. [3]. While this approach also uses the inner state of DNNs, FGS requires the gradient of loss to be calculated only once which makes it more efficient. Furthermore, the authors demonstrated that by using adversarial examples generated with FGS implicitly in an enhanced objective function, both accuracy and robustness of the trained classifiers can be improved.

Rozsa et al. [19] introduced the non-gradient based hot/cold approach, which is capable of efficiently producing multiple adversarial examples for each input. They demonstrated that using samples explicitly with higher magnitudes of adversarial perturbations than the sufficient minimal can outperform regular adversarial training. The authors also presented a new metric – the Perceptual Adversarial Similarity Score (PASS) – to better measure the distinguishability of original and adversarial image pairs in terms of human perception. As the commonly used L2 or L∞ norms are very sensitive to small geometric distortions that can remain unnoticeable to us, PASS is more applicable to quantify similarity and the quality of adversarial examples.

Although adversarial training, both implicit and explicit, was demonstrated to decrease the instability of learning models, forming those examples is still computationally expensive, which limits the application of such techniques. Furthermore, considering the various adversarial generation techniques, utilizing certain types of those samples might not lead to improved robustness to adversarial examples of other techniques. Alternatively, Zheng et al. [26] proposed
their stability training as a more lightweight and still effective method to stabilize DNNs against naturally occurring distortions in the visual input. The introduced training procedure uses an additional stability objective that makes DNNs learn weights that minimize the prediction difference of original and perturbed images. In order to obtain general robustness and not rely on any class of perturbations, the authors applied Gaussian noise to distort the training images.

Gu et al. [5] explored network topologies, pre-processing and training procedures to improve the robustness of DNNs. The authors proposed the Deep Contractive Network (DCN), which imposes a layer-wise contractive penalty in a feed-forward DNN. The formulated layer-wise penalty aims to minimize output variances with respect to perturbations in inputs, and eventually enable the network to explicitly learn flat, invariant regions around the training data. Based on some positive initial results, they concluded that adversarial instability is rather the result of the intrinsic deficiencies in the training procedure and objective function than of model topologies.

Luo et al. [16] proposed a foveation-based technique that selects and uses only a sub-region of the image during classification. As the authors demonstrated experimentally, the negative effect of foveated perturbations to the classification scores can be significantly reduced compared to entire perturbations. Recent research of Graese et al. [4] showed that transformations of the normal image acquisition process can also negate the effect of the carefully crafted perturbations of adversarial examples. While these pre-processing techniques can be efficiently used to alleviate the problem posed by adversarial images, they do not solve the inherent instability of DNNs. In other words, these methods treat the symptoms and not the disease.

In summary, a wide variety of more or less efficient approaches were proposed in the literature that all aim to improve the robustness of DNNs, but none of those proved to be effective enough.

3. Approach

In this section, we first briefly describe our intuition why the unexpected adversarial instability exists in machine learning models. Then we present our simple and straightforward modification in the training procedure that aims to optimize weights in a way that the resulting DNNs become more robust to distortions of their inputs.

3.1. Intuition

During training some inputs in the batch are correctly classified and others are incorrectly classified by the learning model. In general, the gradient of loss is larger for the misclassified ones, and smaller for those that are correctly classified. Therefore, most of the weight updates go into learning those inputs that are badly predicted, while the correctly classified samples can only maintain their decision boundary shape very close to the one they initially obtained when they were first correctly classified. Due to this evolutionary stalling, samples with low gradients cannot form a flatter, more invariant region around themselves. Consequently, samples of those regions remain more susceptible to adversarial perturbations – even a small perturbation can push them back into the incorrect class. By increasing the contribution of the correctly classified examples in the batch on the weight updates, and forcing them to continue improving their decision boundaries, we can flatten the decision space around those training samples and train more robust DNNs.

3.2. Implementation

The core concept of our Batch Adjusted Network Gradients (BANG) is a variation of batch normalization [7]. However, rather than trying to make sure that the inputs of layers are balanced, we seek to ensure that the contributions on the weight updates are more balanced among batch elements by scaling their gradients.

Let us dive into the details. In short, we scale gradients that will be used to compute the weight updates. Let us consider a network \( f_w \) with weights \( w \) in a layered structure having layers \( y^{(l)} \) where \( l \in [1, L] \), with their respective weights \( w^{(l)} \):

\[
f_w(x_i) = y^{(L)} \left( y^{(L-1)} \left( \ldots \left( y^{(1)}(x_i) \right) \ldots \right) \right).
\]

For a given input \( x_i \), the partial derivatives of the loss \( E(f_w, x_i) \) with respect to the output of a layer are:

\[
\kappa_i^{(l)} = \kappa^{(l)}(x_i) = \frac{\partial E_i}{\partial y_i^{(l)}}.
\]

For simplicity, we leave out the structure of the weights \( w^{(l)} \) in each layer which are commonly in a two-dimensional structure, and of the layer output which are either one-dimensional (fully connected layers) or three-dimensional (convolutional layers).

The goal of BANG is to balance gradients in the batch by scaling up those that have low(er) magnitudes. In order to do so, we determine the highest gradient for the batch having \( N \) inputs \( x_i, i \in \{1, \ldots, N\} \) at given layer in terms of \( L_2 \) norm. We use that as the basis to compare gradients in the batch and then to balance their magnitudes. We calculate the weight update by scaling each derivative \( \kappa_i \) in the batch differently with the element-wise learning rate:

\[
\eta_i^{(l)} = \left( \frac{\max_{i' \in [1,N]} ||\kappa_i^{(l)}||}{||\kappa_i^{(l)}||} \right) \rho_i^{(l)}.
\]
where:

$$\rho_i^{(l)} = \epsilon \left( \frac{1 - \frac{\| \kappa_i^{(l)} \|}{\max_{r \in [1,N]} \| \kappa_r^{(l)} \|}}{\rho_p^{(l)}} \right).$$  \hspace{1cm} (4)

$\epsilon$ is one of the parameters of our approach which specifies the degree of balancing. While $\rho_i^{(l)}$ might seem a little complex and unnecessary, its purpose is to scale up smaller gradients more than others having larger $L_2$ norms.

Assuming that the regular backward pass combines the gradients of the batch elements by calculating:

$$\nabla f_w^{(i)} = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial E_i}{\partial w^{(i)}} = \frac{1}{N} \sum_{i=1}^{N} \kappa_i^{(l)} \frac{\partial y_i^{(l)}}{\partial w^{(i)}}$$  \hspace{1cm} (5)

which is normally scaled with the learning rate and then used to update weights (after combining with the previous weight update scaled with momentum), our approach produces:

$$\nabla f_w^{(i)} = \beta^{(l)} \frac{1}{N} \sum_{i=1}^{N} \eta_i^{(l)} \frac{\partial E_i}{\partial w^{(i)}},$$  \hspace{1cm} (6)

where $\beta^{(l)}$ is the second (set of) parameter(s) of our approach used for scaling. In general, $\beta^{(l)}$ acts as a local learning rate that can play a more important role in future work, but here we keep it fixed for all layers; $\beta^{(l)} = \beta$ (which will actually just modify the original learning rate $\eta$).

Note that although we change the actual calculation of weight updates, there is no impact on the backpropagation of the original gradient down the network. Finally, BANG can be implemented by applying small modifications to the regular training procedure with negligible computational overhead.

4. Experiments

To evaluate our approach, we conducted experiments on the slightly modified versions of LeNet 13 and CIFAR-10 models distributed with Caffe 8. Namely, after running preliminary experiments with BANG, we added a Dropout layer 20 to both model architectures that serves multiple purposes. We observed that BANG tends to cause overfitting on the trained LeNet networks, and the resultant models made very confident recognitions – even when they misclassified the test samples. While the additional Dropout layer alleviates both problems, the adjusted network architectures also result in improved recognition performances with both regular and BANG training.

After obtaining learning models with regular and BANG training, we assess and compare the robustness of those classifiers in two ways. It is important to note that we do not select the best training models based on their performance on the validation set for these evaluations, but we simply use the models obtained at the last training iteration.

As our primary goal is to measure the evolving robustness, we believe that this decision leads to a fairer comparison, however, the recognition performance of the selected models are not optimal. Finally, we would like to mention that we conducted experiments to discover the effectiveness of BANG used for fine-tuning regularly trained models, and found that the robustness of the resultant networks are not even comparable to those that we trained from scratch.

First, we evaluate the adversarial vulnerability against two adversarial example generation methods: the gradient based fast gradient sign (FGS) method 3 and the non-gradient based hot/cold approach 19. Although the latter is capable of forming multiple adversarial perturbations for each input, we only target the most similar class with the hot/cold approach, hence, referred as HC1. We aim to form adversarial perturbations for every single correctly recognized image of the MNIST 12 or CIFAR-10 10 test set, respectively. We consider an adversarial generation attempt successful for a given correctly recognized image, if the direction specified by either FGS or HC1 leads to a misclassification, where the only constraint is that the discrete pixel values are in $[0,255]$ range. Of course, this limitation also means that the produced perturbations may or may not be adversarial in nature, as they can be highly perceptible to human observers. We compare the adversarial robustness of classifiers by collecting measures to quantify the quality of the produced adversarial examples. For this purpose, we calculate the Perceptual Adversarial Similarity Score (PASS) 19 of original and adversarial image pairs, and we also determine the $L_\infty$ norms of adversarial perturbations. Although the $L_\infty$ norm is not a good metric to quantify adversarial in terms of human perception, it is capable of demonstrating how far the actual perturbed image is from the original sample.

Second, we quantify how the robustness of the learning models evolve during training by applying a more general approach. For a given pair of classifiers, where one was regularly trained while the other was obtained by BANG training, we add a certain level of random noise to 100 test images from each class that are correctly classified by both classifiers at all tested stages, and compute the proportion of perturbed images that are classified differently than the originating one. While our previously described test assessing the adversarial vulnerability discovers only two directions – specified by the FGS method and the HC1 approach – applying 1000 random distortions to each inspected image for every noise level gives us a more general evaluation. Note that although this experiment is more universal as it does not rely on any specific adversarial generation technique, small random perturbations that cause recognition errors are hard to find 19, hence, the collected results are qualitatively not as good as explicitly forming adversarial perturbations. Furthermore, in order to evaluate the stability of the trained
Table 1: LE.NET TRAINING. This table highlights the difference between LeNet models obtained by using regular (R0-R1) and BANG training (B0-B5). Accuracy on the MNIST test set, the achieved success rates of FGS and HC1 adversarial example generation methods with PASS scores and $L_\infty$ norms of the produced on the MNIST test set are listed.

| ID | $\beta$ | $\epsilon$ | Accuracy | FGS-Rate | FGS-PASS | FGS-$L_\infty$ | HC1-Rate | HC1-PASS | HC1-$L_\infty$ |
|----|---------|------------|----------|----------|----------|-------------|----------|----------|---------------|
| R0 | -       | -          | 99.16%   | 90.33%   | 0.4072 ± 0.1081 | 40.51 ± 15.72 | 99.53%   | 0.7535 ± 0.1143 | 122.60 ± 49.46 |
| R1 | -       | -          | 99.15%   | 91.41%   | 0.4072 ± 0.1065 | 40.70 ± 15.88 | 99.77%   | 0.7517 ± 0.1160 | 122.16 ± 49.07 |
| B0 | 1.00    | 0.785      | 99.16%   | 3.51%    | 0.6806 ± 0.1457 | 8.34 ± 04.32 | 95.13%   | 0.5359 ± 0.2023 | 187.86 ± 63.66 |
| B1 | 1.00    | 0.815      | 99.22%   | 1.68%    | 0.7638 ± 0.1367 | 5.52 ± 02.86 | 94.19%   | 0.4880 ± 0.2110 | 201.84 ± 62.51 |
| B2 | 1.20    | 0.810      | 99.31%   | 2.13%    | 0.7579 ± 0.1452 | 5.57 ± 03.05 | 94.56%   | 0.5129 ± 0.2015 | 186.10 ± 63.77 |
| B3 | 1.35    | 0.780      | 99.25%   | 3.86%    | 0.6763 ± 0.1471 | 8.28 ± 04.26 | 94.73%   | 0.5709 ± 0.2127 | 178.11 ± 65.76 |
| B4 | 1.50    | 0.840      | 99.11%   | 1.52%    | 0.8220 ± 0.1310 | 4.19 ± 03.01 | 97.68%   | 0.4669 ± 0.1881 | 203.50 ± 58.97 |
| B5 | 1.60    | 0.780      | 99.32%   | 4.45%    | 0.6771 ± 0.1487 | 8.20 ± 04.40 | 98.95%   | 0.6376 ± 0.1829 | 146.96 ± 61.97 |

![Accuracy](image1)

(a) Accuracy

![FGS Success Rate](image2)

(b) FGS Success Rate

![HC1 PASS](image3)

(c) HC1 PASS

Figure 2: LE.NET MODELS TRAINED WITH BANG. These plots summarize our results on LeNet models trained with BANG using combinations of $\beta$ and $\epsilon$. We tested a grid of those two parameters where $\beta \in [1.0, 1.6]$ with step size 0.05, and $\epsilon \in [0.78, 0.84]$ with step size 0.005. We trained a single model with each combination and show (a) the obtained accuracy on the MNIST test set, (b) the achieved success rates by using FGS and (c) the mean PASS score of HC1 adversarial examples on the MNIST test images. Each solid green line represents the level of regularly trained learning models. For better visual representation we applied interpolation.

4.1. LeNet on MNIST

We commence our experiments by evaluating BANG on the LeNet model optimized on the MNIST dataset. MNIST contains 70k images overall, 50k used for training, 10k forms the validation dataset, and the remaining 10k images are for testing. The tested network originally has four layers (two convolutional and two fully connected) – extended with one additional Dropout layer – that we optimize without changing the hyperparameters distributed with Caffe. The learning model is trained with a batch size of 64 for 10k iterations using the inverse decay learning rate policy with an initial learning rate of 0.01.

As our novel training procedure has two parameters, $\beta$ defined in Equation (6) and $\epsilon$ introduced in Equation (4), we trained LeNet models with parameter combinations from a grid, and evaluated the accuracy and adversarial vulnerability of the trained classifiers. The results of the conducted experiments are visualized in Figure 2, we also show accuracies and metrics indicating adversarial robustness in Table 1 for some models obtained with regular training (RO-R1) and optimized with BANG training (BO-B5).

As we can see in Table 1, the level of FGS success rates achieved by regular training can be dramatically decreased by BANG: the rate drops from above 90% even below 2%. Almost every single failed adversarial example generation attempt is due to blank gradients – the gradient of loss with respect to the original image contains nothing but zeros – which means that every gradient based method would fail to form adversarial perturbations. As we increase $\epsilon$, in other words, as we balance the contributions of batch elements more by scaling up low(er) gradients, the resultant classifier becomes more resistant to gradient based adversarial generation methods. Although the success rates achieved by the HC1 method remains relatively high, we can observe that the quality of HC1 adversarial examples degrade significantly on LeNet models trained with BANG compared...
to regular training as displayed in Figure 2(c). This degradation is highlighted by both decreasing PASS scores and by the significantly increased $L_\infty$ norms of adversarial perturbations listed in Table 1.

With respect to the achieved classification performances, we find that there can be a level of degradation depending on the selected values for $\beta$ along with $\epsilon$. This phenomenon can be seen in Figure 2(a); it is partially due to random initializations and, in some cases, it is simply the result of overfitting and our decision to evaluate the trained models at 10k iterations. Still, we can observe that BANG training can yield improved classification performance over regular training paired with improved robustness as listed in Table 1.

Additionally, we conducted experiments to quantify and compare how the robustness of classifiers vary during training to random perturbations. For this general approach, we selected to test two classifiers from Table 1: R0 optimized with regular training, and B1 trained with BANG. We can see in Figure 3(a) that the regularly trained model is initially highly susceptible to larger distortions, but as the training evolves it becomes more stable, and settles at approximately 20% with respect to the strongest class of Gaussian noise that we formed by using standard deviation of 100 pixels. Contrarily, the classifier trained with BANG maintains significantly lower rates throughout the whole training as shown in Figure 3(b) and after 10k iterations only 3% of the strongest distortions change the original classification. The absolute improvements are displayed in Figure 3(c).

4.2. CIFAR-10

We also evaluated our novel training procedure on the so-called “CIFAR-10 quick” model of Caffe trained on the CIFAR-10 dataset. CIFAR-10 consists of 60k images, 50k training images, and 10k images used for both validation and testing purposes. The network architecture originally has five layers (three convolutional and two fully connected) that we extended with one Dropout layer, and the learning model is trained with a batch size of 100 for 20k iterations (40 epochs). We use a fixed learning rate of 0.001 that we decrease by a factor of 10 after 36 epochs, and reduce the same way after another 2 epochs.

Due to the different nature of CIFAR-10 training, we slightly adjusted the BANG parameters. Specifically, as the network performance is significantly worse than LeNet on the MNIST dataset yielding proportionately more incorrectly classified samples in each mini-batch, we applied lower local learning rates ($\beta$) and higher values for scaling ($\epsilon$) in BANG training. Furthermore, we found that scaling incorrectly recognized inputs less than correct ones has beneficial effects on robustness, hence, we applied 50% of the specified $\epsilon$ values on incorrect samples of the batch. Similarly to our conducted experiments on LeNet, we trained classifiers on CIFAR-10 with all possible combinations of $\beta$ and $\epsilon$ parameters of a grid, and then determined the accuracy and adversarial vulnerability of the obtained learning models. The results are visualized in Figure 4 and for some models obtained with regular training (R0-R1) and optimized with BANG training (B0-B5) we show accuracies and metrics indicating adversarial robustness in Table 2.

As we can see in Table 2, the level of FGS success rates achieved by regular training is significantly decreased by BANG: the rate drops from approximately 96% to 34%, where again the majority of the failed adversarial example generation attempts are due to blank gradients. Figure 4(b) shows that as we increase $\epsilon$, the classifiers become more resistant to gradient based adversarial generation methods. The higher levels of success rates compared to LeNet might...
be simply due to the fact that the trained classifiers on CIFAR-10 are less accurate, therefore, learning the incorrect samples of the batch still has a large contribution on weight updates. While the success rates achieved by HC1 remains high, we can see that the quality of HC1 adversarial examples degrade significantly compared to regular training. This degradation is highlighted by both decreasing PASS scores shown in Figure 4(c) and by the significantly increased $L_{\infty}$ norms of adversarial perturbations listed in Table 2. Finally, as shown in Table 2, we can train classifiers with BANG that slightly outperform models trained with regular training in terms of classification accuracy. Of course, the achieved overall performance depends on the chosen parameters, as depicted in Figure 4(a).

Finally, we ran experiments to better quantify and compare how the general robustness to random perturbations of trained classifiers evolve during training. Similarly to our described experiments on LeNet, we selected two classifiers from Table 2 for testing: R0 trained regularly, and B0 optimized with BANG. We can see in Figure 5(a) that the regularly trained R0 model is highly susceptible to larger distortions, its robustness does not improve during training, and finally achieves 46.0% with respect to the strongest class of Gaussian noise that we formed by using standard deviation of 40 pixels. Contrarily, the B0 model trained with BANG remains more robust throughout training epochs as shown in Figure 5(b) and at the end 39.1% of the strongest distortions change the original classification. The absolute improvements are visualized in Figure 5(c). We can conclude that although BANG enhanced robustness to random perturbations, the results are less impressive in comparison to LeNet – at least, with respect to the strongest distortions.

5. Conclusion

In this paper, we introduced our theory to explain the revealed intriguing property of machine learning models. Namely, the regular training procedure can prevent training samples from forming flatter and broader regions around themselves, and this evolutionary stalling yields samples re-

---

Table 2: CIFAR-10 TRAINING. This table shows the difference between classifiers obtained using regular (R0-R1) and BANG training (B0-B5). The accuracy on the CIFAR-10 test set, the achieved success rates of FGS and HC1 adversarial example generation methods with PASS scores and $L_{\infty}$ norms of the formed CIFAR-10 test images are listed.

| ID | $\beta$ | $\epsilon$ | Accuracy | FGS-Rate | FGS-PASS | FGS-$L_{\infty}$ | HC1-Rate | HC1-PASS | HC1-$L_{\infty}$ |
|----|---------|-----------|----------|----------|---------|----------------|----------|----------|----------------|
| R0 | -       | -         | 79.59%   | 96.52%   | 0.9553 ± 0.0969 | 4.08 ± 0.6440 | 98.97%   | 0.9669 ± 0.1005 | 18.15 ± 29.80 |
| R1 | -       | -         | 79.55%   | 96.71%   | 0.9513 ± 0.1057 | 4.43 ± 0.7055 | 98.91%   | 0.9557 ± 0.1332 | 22.16 ± 39.77 |
| B0 | 0.40    | 0.855     | 79.26%   | 34.27%   | 0.9511 ± 0.1302 | 4.11 ± 0.1031 | 95.94%   | 0.8712 ± 0.1649 | 55.52 ± 49.98 |
| B1 | 0.45    | 0.805     | 80.43%   | 45.94%   | 0.9818 ± 0.0548 | 2.04 ± 0.0249 | 96.20%   | 0.7966 ± 0.2438 | 77.34 ± 71.20 |
| B2 | 0.75    | 0.800     | 79.74%   | 41.71%   | 0.9828 ± 0.0586 | 1.94 ± 0.0303 | 98.34%   | 0.8362 ± 0.2195 | 64.26 ± 63.57 |
| B3 | 0.75    | 0.845     | 79.41%   | 35.00%   | 0.9526 ± 0.1266 | 3.94 ± 0.0871 | 96.54%   | 0.8603 ± 0.1981 | 59.83 ± 58.28 |
| B4 | 0.95    | 0.840     | 79.30%   | 34.88%   | 0.9575 ± 0.1236 | 3.61 ± 0.0960 | 96.87%   | 0.8994 ± 0.1487 | 48.44 ± 47.35 |
| B5 | 1.00    | 0.800     | 79.22%   | 41.34%   | 0.9803 ± 0.0722 | 2.03 ± 0.0364 | 98.17%   | 0.8586 ± 0.1948 | 61.14 ± 61.23 |

Figure 4: BANG CIFAR-10 MODELS. These plots summarize our results on CIFAR-10 models trained with BANG using combinations of $\beta$ and $\epsilon$. We tested a grid of those two parameters where $\beta \in [0.4, 1.0]$ with step size 0.05, and $\epsilon \in [0.80, 0.86]$ with step size 0.005. We trained a single model with each combination and show (a) the obtained accuracy on the CIFAR-10 test set, (b) the achieved success rates by FGS, and the (c) mean PASS score of HC1 adversarial examples on the CIFAR-10 test images. Each solid green line represents the level of regularly trained learning models. For better visual representation we applied interpolation.
Figure 5: CIFAR-10: Robustness to Random Distortions. These plots show the evolving robustness of CIFAR-10 models: (a) obtained with regular training (R0 from Table 2), (b) trained with BANG (B0 from Table 2), and (c) displays the improvement. After identifying 100 test images per class that are correctly classified by both networks at every second epoch, we perturb each 1000 times with the level of Gaussian noise specified by the standard deviation, and test the networks at different stages of training. The plots show the percentage of distortions yielding misclassifications. For better visual representation we applied interpolation.

remaining close to decision boundaries and, hence, being susceptible to imperceptibly small perturbations causing misclassifications. To address this problem, we proposed a novel approach to improve the robustness of Deep Neural Networks (DNNs) by slightly modifying the regular training procedure. Our approach does not require additional training data – neither adversarial examples nor any sort of data augmentation – to achieve improved robustness, while the overall performance of the trained network is maintained or even enhanced.

We experimentally demonstrated that optimizing DNNs with our Batch Adjusted Network Gradient (BANG) technique leads to significantly enhanced stability in general. By balancing the contributions of elements in a batch on forming the weight updates, BANG allows training samples to form flatter, more invariant regions around themselves. The trained classifiers become more robust to random distortions, and as we demonstrated with the gradient based Fast Gradient Sign (FGS) method and the non-gradient based hot/cold approach where we targeted the closest scoring class (HC1), they are also less vulnerable to adversarial example generation methods that aim to form adversarial perturbations explicitly. While BANG is proven to be efficient in mitigating adversarial instability, learning models can also maintain or even improve their overall classification performances due to their enhanced generalization properties. Our proposed approach achieves these results with no data augmentation involved, and with negligible computational overhead over the regular training procedure.

Although with BANG we managed to achieve good results on two DNNs trained on different datasets, we found that parameters needed to be adjusted to these optimization problems which can be cumbersome. In order to obtain better results, exploring the effect of various BANG parameters applied to different layers, and changing the contributions of correctly and incorrectly classified inputs in the batch by having different \( \epsilon \) ratios might be considered. Future work will focus on having a better understanding of BANG, enhance the algorithm to be more self-adaptive, and explore how well it can be applied for training deeper neural networks on real-world datasets. While some might argue that similar balancing effect can be achieved by distillation, Carlini et al. [1] demonstrated that defensive distillation is not effective to improve adversarial robustness.

In summary, we can conclude that the discovered adversarial instability of DNNs is closely related to the applied training procedures – as it was claimed by Gu et al. [5] – and there is a huge potential in this research area to further advance the generalization properties of machine learning models and their overall performances as well.

Acknowledgments

This research is based upon work funded in part by NSF IIS-1320956 and in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.
References

[1] N. Carlini and D. Wagner. Defensive distillation is not robust to adversarial examples. arXiv preprint arxiv:1607.04311, 2016.

[2] A. Fawzi, O. Fawzi, and P. Frossard. Fundamental limits on adversarial robustness. In International Conference on Machine Learning (ICML), Workshop on Deep Learning, 2015.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In International Conference on Learning Representation (ICLR), 2015.

[4] A. Grause, A. Rozsa, and T. E. Boult. Assessing threat of adversarial examples on deep neural networks. In IEEE International Conference on Machine Learning and Applications (ICMLA), 2016.

[5] S. Gu and L. Rigazio. Towards deep neural network architectures robust to adversarial examples. In International Conference on Learning Representation (ICLR) Workshops, 2015.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[7] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International Conference on Machine Learning (ICML), 2015.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In International Conference on Multimedia. ACM, 2014.

[9] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint arXiv:1609.04836, 2016.

[10] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

[11] H. Lai, Y. Pan, Y. Liu, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[12] Y. LeCun, C. Cortes, and C. J. Burges. The MNIST database of handwritten digits, 1998.

[13] Y. LeCun, L. Jackel, L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, P. Simard, et al. Learning algorithms for classification: A comparison on handwritten digit recognition. Neural networks: the statistical mechanics perspective, 261:276, 1995.

[14] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen. Deep learning of binary hash codes for fast image retrieval. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2015.

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[16] Y. Luo, X. Boix, G. Roig, T. Poggio, and Q. Zhao. Povetion-based mechanisms alleviate adversarial examples. arXiv preprint arXiv:1511.06292, 2015.

[17] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, C.-C. Loy, et al. DeepID-Net: Deformable deep convolutional neural networks for object detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[18] A. Rozsa, M. Günther, E. M. Rudd, and T. E. Boult. Are facial attributes adversarially robust? In International Conference on Pattern Recognition (ICPR), 2016.

[19] A. Rozsa, E. M. Rudd, and T. E. Boult. Adversarial diversity and hard positive generation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2016.

[20] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research (JMLR), 15(1):1929–1958, 2014.

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[22] C. J. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In International Conference on Learning Representation (ICLR), 2014.

[23] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[24] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Convolutional channel features. In IEEE International Conference on Computer Vision (ICCV), 2015.

[25] Z. Zhang, Y. Chen, and V. Saligrama. Efficient training of very deep neural networks for supervised hashing. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[26] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.