Background Class Defense Against Adversarial Examples

Michael McCoyd and David Wagner
University of California, Berkeley

***Abstract*—Adversarial examples allow crafted attacks against deep neural network classification of images. We propose a defense of expanding the training set with a single, large, and diverse class of background images, striving to 'fill' around the borders of the classification boundary. We find it aids detection of simple attacks on EMNIST, but not advanced attacks. We discuss several limitations of our examination.**

I. Introduction

Deep neural networks have been very successful at many classification tasks. Yet they have been found to be vulnerable to misclassifying deliberately crafted images [SZS+14], [GSS15].

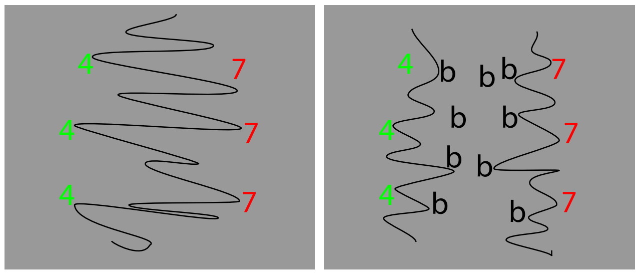
We examine whether adding a large and diverse background class to the training data can help detect and neutralize adversarial examples. We assume that there is a set of *key classes* that we care about distinguishing between. We propose to train the classifier by adding additional training images not from the key classes that we care about distinguishing between. We introduce a new class, that we call the *background class*, for these additional images. The background class effectively serves as a "none of the above". For example, if we want to recognize MNIST digits, we might use images of non-digit handwriting as examples of the background class. The background class is intended to create a default classification to provide better separation between the key classes within the model's feature space. We measure how this defense affects classification of normal images and detection of adversarial examples.

Figure 1 illustrates a simplified visualization of the decision boundary of an undefended classifier (left) and our proposed defense (right). We hope that the background class makes it harder to modify an image from one key class to be misclassified as another key class.

These background images are not noise, but images of objects distinct enough from our key classes to be distinguished from them by the model.

In use, our network is still evaluated on its ability to distinguish between images that humans perceive as key classes. We change the classification task from image $\rightarrow$ one of key classes, to image $\rightarrow$ one of key classes or background.

We evaluate this idea with the EMNIST dataset [CATv17], of hand written digits and letters, and the fast gradient sign [GSS15], fast gradient [LCLS17], and Carlini [CW17] attacks.



(a) The original classifier (b) With background class (our defense)

Fig. 1: The intuition behind our defense. In (b), we introduce an extra class, the background class, for images other than the classes we want to recognize. The hope is that this separates those classes enough that adversarial examples are no longer effective.

We find our defense aids detection of simple attacks but we have not found any increase in detection of state of the art attacks. There are weaknesses in our examination, in particular the letters of our background class are not crafted for the role of separating the digits in image or feature space.

II. Background

Neural networks have been very successful at learning a function, $F_\Theta(x)$, by tuning a set of weights, Θ, based on training data, x, with labels, y, that allow them to then accurately label new data. For m-class networks, y is a probability distribution over m labels. These networks are trained by defining a loss function, $R(\Theta)$, that defines how far off their predicted labels for the training data are from the training data's actual labels. The loss function is evaluated on the training data. Back propagation [RHW88] is used to allocate blame for the error in each of the network's outputs. This output error is propagated backward through the network based on the current weights on each neurons inputs. This allows the weights at each layer to be adjusted appropriately in the direction of reducing the cost, $\partial_\Theta R_\Theta(x)$.

Deep Neural networks have improved neural network performance, achieving near human level performance on several visual tasks, including visual recognition of objects

[RDS+15] and road signs [SSSI11], leading to their increased use.

A. Adversarial Examples

It has been found that an attacker with knowledge of the model can construct, from any normal image, x, an adversarial example [SZS+14], [GSS15], $x^\star$, that looks to humans like the normal image's classification but that the model classifies differently from the normal image. In an *untargeted* attack, the attacker might strive for these images to merely be classified by the model differently from how humans would classify them. In a *targeted* attack, they might strive for them to be classified as a specific class. A recent overview of adversarial examples can be found in [CW17].

Two important aspects of adversarial examples are that 1) they are classified by humans and the model differently, and 2) we care about that misclassification for some task. Our defense will exploit the second of these.

Three quantitative metrics are commonly used to measure how much an attack image differs from the original image, the L_0, L_2, and L_∞ distance metrics. L_0 is the number of pixels changed. L_2 is the square root of the sum of the squares of the changes to each pixel. L_∞ is the maximum change made to any pixel.

B. Attacks

The basic approach of finding adversarial examples can be seen as a variation on the normal back propagation training of the model weights. Instead of propagating the error back through the network to ascribe proportional blame to the weights, we ascribe proportional blame to each input pixel. For each pixel we obtain a contributory sign and magnitude for the errors in the output neurons. Attack methods vary in how they use this information. All attack images must be clipped as needed to remain within the range of valid pixel values. There are several attacks; we highlight only a few.

a) Fast Gradient Sign: One of the simplest attacks to understand is the fast gradient sign attack [GSS15]. Using back propagation, it finds the gradient of the loss function with respect to the inputs, simplifies that to just the sign of the gradient[1], and then moves by a factor τ in that direction:

$$x^\star \leftarrow x + \tau\,\mathbf{sgn}(\nabla_x R(F(x))).$$

b) Fast Gradient: The fast gradient attack [LCLS17] normalizes the gradient to have an L_2 norm of one and then moves by a factor τ in that direction:

$$x^\star \leftarrow x + \tau\,\frac{\nabla_x R(F(x))}{\|\nabla_x R(F(x))\|_2}.$$

c) Carlini: The Carlini attack [CW17] is a strong iterative attack that reduces the set of pixels it uses by repeatedly eliminating the least effective pixels from the set it uses. It produces attack images with very small changes, based on either L_0, L_2, or L_∞ distances.

[1]Normalizing to an L_∞ norm of one.

(a) Original image, classified as a 9. (b) Attack image, classified as a 7.

Fig. 2: Fast gradient untargeted L_2 attack with $\tau = 6.79$.

(a) Original image, classified as a 2. (b) Attack image, classified as a 3.

Fig. 3: Carlini targeted L_2 attack, adapted from [CW17].

C. Defenses

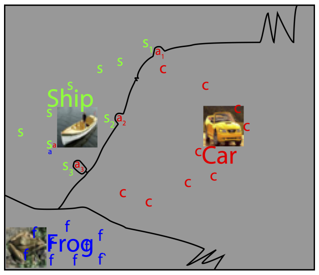
Several defenses have been proposed but none so far are effective. One of the most natural defenses is adversarial retraining. [GSS15] found that an MNIST network trained with fast gradient sign adversarial examples reduced the success of that attack from 89.4% to 17.9%. [SYN15] retrain based on a gradient step of L_1, L_2, or L_∞, and see increased robustness to fast gradient sign attacks, while achieving increased accuracy on normal images due to regularization, on MNIST and CIFAR-10 datasets. Yet unpublished work by others find that such defenses are vulnerable to attack. They find that for MNIST, retraining with images generated with the fast gradient sign attack only increases the mean minimal distance of adversarial examples by a factor of two measured using the Carlini L_2 attack.

III. Background Class Defense

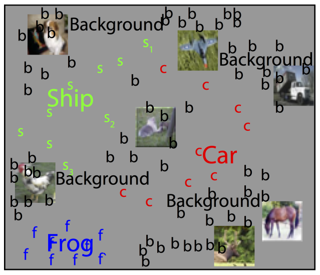
Without a diverse background class, there are large areas of the feature space that are not represented in the training data. In these unrepresented parts of the feature space, there is little to drive the model toward a smooth decision boundary. Our rationale is that having no classifications besides the ones we care about may also contribute to not having the smoothness we would prefer in the areas around our key classes.

For our work, there are two key aspects to adversarial examples: 1) for humans they are perceptually close to other images that the model classifies differently, 2) we care about that misclassification. We hope to manipulate the model so that when perceptually close images are classified differently, we no longer care as much, as the classification difference is between a key class and the background class instead of between two key classes. Such misclassification as the background class alerts us that we are under attack.

a) Illustrating Example: As an example, we might want to distinguish between ships, cars, and frogs, as in figure 4. With normal training, the model learns to separate the key classes, yet the decision boundary may have a very erratic shape in areas with no training data, and the boundary includes adversarial examples around all the training data points.



(a) Labels: Ship, Car, Frog



(b) Labels: Ship, Car, Frog, Background

Fig. 4: Example classification task. (a) with a potentially erratic decision boundary in areas with no data and adversarial examples including a_1, a_2, a_3, (b) with addition of background class training data.

Our training adds one extra label 'background' for a diverse set of background images added to the training data, e.g.: birds, trucks, planes, cats, dogs, horses, and deer, as in figure 4b. We hope that the model will find it cheapest to learn a decision boundary that connects all these background images as one area, restricting the key class areas to be around their training data. The goal is that around images perceived as key classes by humans, such as S_2, the decision boundary might still not be smooth, but there would no longer be adversarial examples misclassified as other key classes, but only images misclassified as background, a misclassification we are less concerned about. Such background classifications might serve as a flag of some failure in our input or of an attack.

IV. Data

We base our evaluation on the EMNIST dataset, chosen for ease of training. EMNIST [CATv17] expands MNIST to include letters. It contains 28×28 grayscale images of handwritten digits and upper and lower case letters, in 62 classes, with 697,932 training images and 116,323 test images. In our EMNIST tests, the digits serve as our key classes and the letters collectively as the background class. We adjust pixels to the range [-0.5, 0.5].

EMINST has different versions depending on whether you want digits and/or letters and whether you want the classes to be balanced. The Balanced set has digits and letters with balanced class sizes. The By Merge set is roughly balanced within the digits but very unbalanced within the letters; it is however larger. Both of these datasets have 47 classes as they merge several letters whose lower/upper appearances are similar, the 15 letters 'cijklmopsuvwxyz'.

A. *Confusing Letters*

As a source of background images, letters have a problem: some look to humans like digits. We are trying to use the background class to fill in the part of the input/feature space that is away from the key classes and thus normally unrepresented. Yet pairs such as 0/O and 1/l look alike. Using such letters in our background class brings the edges of the background class closer to our key classes than we may need for our defense.

As a data preparation step, we pick letters for our background class that are very distinct from digits. We measure digit/letter confusion by creating a confusion matrix using the original digit and letter labels. For each letter we measure how often it is classified as any letter or 'misclassified' as any digit. We keep for our background class the 22 letters, 'TtUdEPfFXHCnhAKMVWeRrN', that are classified as letters $98-100\%$ of the time, an arbitrary cutoff. We remove from the dataset the remaining 15 letters, 'OLqIgZSYGbDBJaQ', that are classified as letters only $69-97\%$ of the time. To measure this we use 10-fold cross validation repeated four times on the Balanced set trained for 20 epochs. We used the Balanced set for these tests because the larger By Merge set caused memory issues.

B. *Digits + Background Dataset*

We train our defense with the By Merge dataset, the largest EMNIST dataset, stripped of the confusing letters found above. All letters are merged into a single background class, so the classifier classifies every image into one of 11 classes (0-9, or background). This gives us a dataset for the train/validation splits with class counts of between 31,280 and 38,304 for 0-9 and a class count of 212,657 for the background class, with a count of 2,535 for the least frequent letter and 24,657 for the most, summarized in table I.

TABLE I: Data sizes

Baseline classifier (no background class):	
Train:	310,912 digits
Test:	34,432 digits
With background class defense:	
Train:	310,912 digits + 191,360 letters
Test:	34,432 digits

We found that using more of the confusing letters during training increases detection of attack, yet decreases accuracy when not under attack. In the extreme it must learn that zero and the letter 'O' belong to different classes. We do not have enough results on that tradeoff to report more fully.

V. Standard Model and Training

Our EMNIST model is very simple. It is the example MNIST model of the Keras framework with dropout replaced by batch normalization, described in table II. A ReLU activation function was used for the convolutional and first fully connected layers. Logits are produced by the final layer. We did not do any tuning of model parameters. We simply picked common values for kernel, stride, and layer outputs.

TABLE II: EMNIST model's architecture

Layer	Kernel	Outputs	Activation
Conv	3×3	32	BatchNorm + ReLU
Conv	3×3	64	BatchNorm + ReLU
MaxOut	2×2	64	
Dense		128	BatchNorm + ReLU
Dense		11	BatchNorm

Training was done with the Ada Delta optimizer[Zei12] with a cross entropy loss function and batch size of 128.

a) Results Before Attack: To measure the cost of the defense, we examine the model's classification rates for normal images when trained on the data sets, before examining changes to the robustness to adversarial examples. Training and testing were done with 10 random 90:10 splits of the train dataset for 10 epochs each.

The accuracy is shown in table III. We obtain 99.6% accuracy when trained on digits, and 99.1% when trained on digits and letters. These results are not state of the art for MNIST, but those trained with the background class are still near the range of performance for which we turn to deep models for.

TABLE III: Classification accuracy for normal images (not under attack)

Dataset	Mean Accuracy (%)		
	Correct	Other	Background
Digits	99.62	0.38	
Digits+Background	99.11	0.41	0.48
Change	-0.51	+0.03	+0.48

VI. Attack Results

We use three attacks from the literature in our evaluation, two weak attacks and one strong. Fast gradient sign has been used in the past to evaluate defenses, though it is now considered a weak attack. Fast gradient is an L_2 version of fast gradient sign. The Carlini attack is a strong L_2 attack that has defeated several proposed defenses in the literature.

Table IV summarizes the difference that addition of our background class has on these attacks. In each case the attack has full access to the trained model under attack, and uses images from the EMNIST dataset. We do not see an increase in security against the Carlini attack. We discuss these attacks below.

TABLE IV: Attack Success Rates

	Attack		
Dataset	FGS	FG	Carlini
No background	92%	94%	100%
Background	68%	56%	100%
Change:	-24%	-38%	0%

A. Weak Attacks

As an initial test of our defense, we use fast gradient sign and fast gradient, with epsilons of 0.25 and 6.79 respectively. Fast gradient with $\epsilon = 6.79$ produces an L_2 difference similar to fast gradient sign with $\epsilon = 0.25$. As before, training and attack were done with 10 random 90:10 splits of the train dataset for 10 epochs each.

Table V reports details of whether each attack image is classified with its correct original classification, some other key classification, or the background classification. Fast gradient sign shows a reduction in attack success from 92% to 68% when the background class is used; the attack is detected 29% of the time when we use the background class. Fast gradient shows a reduction in attack success from 94% to 56% when the background class is used; the attack is detected 43% of the time when we use the background class. These are significant reductions in attack success.

B. Strong Attack

One of the strongest current attacks is the targeted Carlini attack. We evaluate our defense against it, training and testing on a smaller version of the EMNIST dataset, the balanced dataset instead of the by merge dataset.

Our results show no increase in detection of the Carlini attack. The Carlini L_2 attack succeeds 100% of the time for both a dataset of digits and a dataset of digits plus background.

Because we have not found an increase in detection of strong attacks, we have not used the testing portion of the EMNIST dataset to produce our results.

VII. Limitations

We have not found the approach to work, but have not pushed hard at it. These are some possible changes.

A. Strain the model's capacity

In adding a large background class, when does the model architecture have to change to preserve accuracy on key classes? We did not change the model architecture for our work, other than switching it to use batch normalization in an effort to avoid having to tune hyper parameters to maintain normal image accuracy. Yet a key hypothesis in our work is that adding the background class will force the model to learn a default classification of background for the area between the key classes. We hoped this would be

TABLE V: Classification rates for adversarial images (weak attacks)

Attack and Dataset	Mean Classification %			Standard Deviation		
	Correct	Other	Background	Correct	Other	Background
Fast Gradient Sign						
Digits	8.26	91.74		1.92	1.92	
Digits+Background	2.26	68.39	29.35	0.99	12.50	13.22
Change	-6.00	-23.35	+ 29.35			
Fast Gradient						
Digits	5.91	94.09		1.42	1.42	
Digits+Background	1.21	55.99	42.80	0.80	6.82	7.11
Change	-4.70	-38.10	+42.80			

the cheapest thing to learn and that the model could not afford to learn anything more complex. Yet we have not pushed at keeping the capacity of the model under pressure. How do the results change if we remove capacity from the network until the point when we can no longer maintain normal image accuracy?

B. *Background images close to and between key classes*

We used letters as background images for simplicity. Yet, with random locations, there is little guarantee that they fill the crucial space between our key classes. If the idea behind our approach is a good one, ideally we would like to sprinkle background images throughout the non-key parts of the space. And especially all along the border of each key class, but a bit separated from it so as to not degrade performance on images within the key class, such as figure 1.

One approximation of the around/between and close part would be to use various morphing algorithms to change images from one key class into images of another key class. Use a human to judge when the image stops being the first class and when it starts being the second class. Test that this method does not cause your image paths to cross over any other key classes along the way. Hopefully you find a pattern for these change points, such as 30% and 70% along the way, and use those points, or just 50%, as a new background image. As you fill in the space, classification of new ones as background images might reliably allow you to avoid creating auto generated background images that were actually key class images. A complication is that each key class may have several separate areas in the feature space.

A possible experiment to test this would be to pick a small subset, or one, of our key classes and use this to sprinkle the area surrounding those classes with close background images. Then evaluate whether attacks involving this subset were more difficult than attacks involving only other key classes.

Such artificial construction of background images is almost certainly required for image spaces with higher dimensionality or with far more classes, though at some point this may cause the background images to no longer appear normal, being just a synthetic morph between two points in the image space.

C. *Use more background images*

We could expand the types of images used for the background class or use standard data augmentation techniques. For our background images we stayed within the same domain that our model had to learn, simple things that were hand written. This was deliberate in an attempt to not change too much what the network needed to learn but instead to just add a new large class. Staying within that domain, letters could be chopped into subareas and rearranged to form random 'characters' that were not digits. This would add extra sharp edges that are not otherwise present in our data. Using random images – trees, cars, faces – would expand the data set even more. Also the more complex domains, represented by such datasets as CIFAR, GTSRB, and ImageNet, likely do not have a simple and similar domain from which to draw such images, so use of more random background images would be necessary for our method in harder image domains.

VIII. Related Work

Bendale and Boult [BB16] examine detecting images from the part of the image space not trained on, which they call the open set. They examine open set and fooling images, detecting and rejecting images that look to humans like an unknown class. They do not examine adversarial images. Their open set is essentially the same as our background class. However, a crucial difference is that they do not use open set images in training, while we train with images from the background class to try to avoid adversarial examples.

Melis et. al propose a similar defense, where the classifier has the option to reject an image as not belonging to any of the trained classes [MDB$^+$17]. During training, the classifier learns a region for each class that encompasses most or all of the examples of that class in the training set. At test time, the classifier rejects the image if it does not fall into any of these regions. Thus, their defense can be considered another open-set scheme. Like Bendale and Boult, they train only on normal images, but do not use open set images in training; in contrast, our defense trains on examples of the background class.

Goodfellow et al. discuss rubbish classes, "degenerate inputs that a human would classify as not belonging to any of the categories in the training set", and the problem of not classifying such images as belonging to one of the expected classes [GSS15]. They discuss rubbish classes in connection with the fooling images of Nguyen et al.[NYC15], abstract or apparently random images that are classified with high confidence. Goodfellow et al. show that on CIFAR-10 it is easy to create a fooling image from a random image by adding a few gradient steps toward a target class. Neither group trains the classifier on such rubbish images, nor do they discuss normal images that are just not in the classes of interest. We train on background class images that are normal, not random, images.

Hosseini et. al introduce NULL labeling to mark adversarial examples added in training, so that under attack labeling shifts to the NULL class [HCK+17]. We hope for a similar shift, but without relying on distinguishing adversarial examples. We rely on background being the available misclassification.

IX. Conclusion

Adding a background class to image classification training significantly increases detection of simple adversarial example attacks, but we have not found it to stop advanced attacks. For the weak attacks, we have reduced the success rate significantly, suggesting that we have filled the space between our key classes with the background class. For the strong Carlini attack, we see no effect, suggesting that the trained model is still very convoluted near each of the training examples, and still vulnerable to attack. While not a successful defense in itself, adding the background class to the training data may be a useful tool. Further work could investigate constructing background images between the key classes and artificially expanding the background data.

References

[BB16] Abhijit Bendale and Terrance E. Boult. Towards Open Set Deep Networks. *CVPR*, 2016, arXiv:1511.06233 [cs.CV].

[CATv17] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. EMNIST: an extension of MNIST to handwritten letters. *CoRR*, 2017, arXiv:1702.05373 [cs.CV].

[CW17] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *Security and Privacy*, 2017, arXiv:1608.04644 [cs.CR].

[GSS15] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *ICLR*, 2015, arXiv:1412.6572 [stat.ML].

[HCK+17] Hossein Hosseini, Yize Chen, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. Blocking Transferability of Adversarial Examples in Black-Box Learning Systems. 2017, arXiv:1703.04318 [cs.LG].

[LCLS17] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into Transferable Adversarial Examples and Black-box Attacks. *ICLR*, 2017, arXiv:1611.02770 [cs.LG].

[MDB+17] Marco Melis, Ambra Demontis, Battista Biggio, Gavin Brown, Giorgio Fumera, and Fabio Roli. Is Deep Learning Safe for Robot Vision? Adversarial Examples against the iCub Humanoid. *ViPAR*, 2017, arXiv:1708.06939 [cs.LG].

[NYC15] A. Nguyen, J. Yosinski, and J. Clune. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. *CVPR*, 2015, arXiv: 1412.1897 [cs.CV].

[RDS+15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vision*, 115(3), December 2015, arXiv: 1409.0575 [cs.CV].

[RHW88] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning Representations by Back-propagating Errors. In James A. Anderson and Edward Rosenfeld, editors, *Neurocomputing: Foundations of Research*. MIT Press, 1988.

[SSSI11] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *IEEE Int. Joint Conference on Neural Networks*, 2011.

[SYN15] Uri Shaham, Yutaro Yamada, and Sahand Negahban. Understanding adversarial training: Increasing local stability of neural nets through robust optimization. *CoRR*, 2015, arXiv:1511.05432 stat.ML.

[SZS+14] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *ICLR*, 2014, arXiv:1312.6199 [cs.CV].

[Zei12] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, 2012, arXiv:1212.5701 [cs.LG].