Improving the adversarial robustness of ConvNets by reduction of input dimensionality

Akash V. Maharaj
Department of Physics, Stanford University
amaharaj@stanford.edu

Abstract

We show that the adversarial robustness of deep neural networks can be improved by reducing the dimensionality of input data. Adversarial examples are generated by small perturbations of regular input data, and while these changes are typically imperceptible to the human eye, they can result in catastrophic losses of accuracy in neural networks. This has been shown to be a generic property of quasi-linear classifiers in high dimensional spaces; thus, it is intuitively possible that reducing the dimensionality of the inputs leaves less 'space' for adversarial examples to exist. We present detailed results demonstrating this improved robustness (with small sacrifices in accuracy) for 5 layer networks on the CIFAR10 dataset, as well as for the deeper CaffeNet and GoogLeNet architectures on the Tiny-Imagenet-200 dataset. In particular, a reduction in the number of allowed pixel intensities from 256 to 4, which corresponds to robustness to perturbations of up to 32 units of intensity, results in an accuracy reduction of only 6% for CIFAR10 and 8% for Tiny-Imagenet.

1. Introduction and related work

Convolutional neural networks (ConvNets) have by now exhibited better-than-human performance[1] in vision classification problems. It was therefore both surprising, and a bit depressing, when Szegedy et al.[2] discovered that these classifiers are remarkably vulnerable to so called adversarial examples - small, deliberate changes of the pixel intensities, designed to fool the classifier. While such small changes are virtually imperceptible to humans, state-of-the-art ConvNets typically classify such images incorrectly, but with very high confidence. Even more worryingly, ConvNets can be convincingly fooled by tuning the pixel intensities of random noise [3]. Furthermore, adversarial examples tend to be similarly misclassified by different, separately trained ConvNets leading to fears for both the security and generalizability of deep neural networks.

From the subsequent explosion of literature investigating this problem, several themes have emerged. First, susceptibility to adversity has been shown to be a property of high dimensional dot products[4] and so will plague any sufficiently linear model, as many ConvNets are thought to be. Second, while including adversarial examples in the training phase can act as a form of regularization, there are relatively few successful strategies for combatting adversarial examples in vanilla ConvNets, trained on natural images.[4]

Finally, as emphasized by [3], at the heart of the issue is the difference between discriminative and generative models. ConvNets are discriminative models, which for a label vector $\boldsymbol{y}$ and input vector $\boldsymbol{x}$ learn the probability distribution $p(\boldsymbol{y}|\boldsymbol{x})$. Essentially, these models partition a very high dimensional input space into some set number of classes, with class boundaries partitioning the space into hyper-volumes that are *much* larger than the volume occupied by the training examples, as demonstrated in Fig. 1(a). Because natural input images occupy a very restricted manifold in this high dimensional space, the class predictions for vast regions of the input space are completely arbitrary. Thus, adversarial examples are essentially synthetic images which end up 'far' from typical data, and will be incorrectly classified with a high degree of confidence. What is needed is a generative model, where the complete joint density $p(\boldsymbol{y}, \boldsymbol{x})$ is known; in such a case the prior probability $p(\boldsymbol{x})$ may be used to rule out adversarial examples as being too improbable. These considerations have led to the development of Generative Adversarial Nets (GANs) [5].

Given the above discussions of dimensionality, an alternative—and we believe hitherto unexplored—possibility is to *discretize the manifold from which images are classified*. Eliminating portions of the input manifold can be viewed as an inflexible prior; we are essentially reducing the regions into which data can be perturbed and misclassified by reducing the hyper-volume which must be partitioned. The resulting robustness to adversarial examples is then trivial: there is now a minimum allowed distortion of the image (due to the discretization, as shown in Fig. 1(b)), and so a proposed adversarial image will be

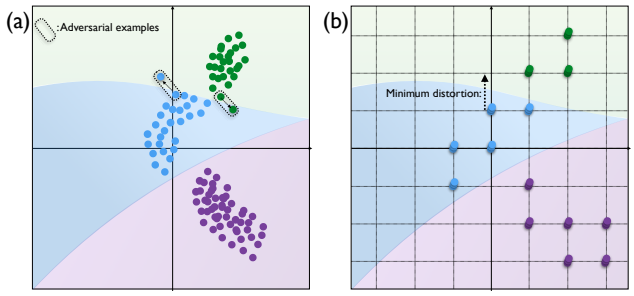


Figure 1: A cartoon demonstration of the intuition behind discretizing inputs. True data (dots) lie on a narrow manifold in the high dimension space, and the classifier is trained to separate this data into categories (shaded regions). However the decision boundaries in the vast swathes of "unpopulated" space are arbitrary and untrained. Adversarial examples are images perturbed into this naturally empty region and are thus, misclassified. Discretizing inputs introduces a minimum allowable distortion, and helps prevent small adversarial perturbations.

perceptibly different from a natural one.

From a human perspective, this discretization is perhaps more intuitive than one may at first imagine. For example, humans can typically classify grayscale images, suggesting that all three color channels may not be necessary for categorical data. Indeed, the higher convolutional layers of ConvNets have been shown to be efficient feature extractors[6] which should be robust to color. Furthermore, there is a coarse graining procedure that is inherent to our vision - the presence of displaced pixels is irrelevant to our perception of an image. Thus, while color is certainly important to classification tasks, knowledge of all 256^3 RGB colors is probably unnecessary. Both these 'intuitive' observations lead to the hypothesis that *not every single dimension of the input space is relevant*.

Discretization has the added attraction of improved computational efficiency and decreased memory requirements, which is naturally a desirable feature for any real world deployment of deep neural networks. Indeed, in a very recent preprint, Courbariaux et al.[7] introduced and discussed deep binary neural networks (BNNs), where *all* the weights and activations are binary. Despite the severe non-analyticities introduced to the loss function by such 'binarization', the authors report near state of the art results with impressive speedups in the training of such networks. While Courbariaux et al.[7] did not discuss discretizing of inputs as we do here, this report can be viewed as a natural generalization of the BNN concept.

In Sec. 2.1 of this report, we describe how we achieve the reduction in dimensionality of the input by quantizing RGB pixel intensities into a variety of allowed discretizations, ranging from 2^2 to the full $2^8 = 256$ allowed intensities. For the CIFAR10 dataset[8], we train a 5 layer neural network from scratch for each discretization as is described in Sec. 2.4.1, and report in detail on the changes in accuracy and robustness of the network in Sec. 3.2. We then generalize these methods to the deeper, pretrained CaffeNet[9] (a modified AlexNet[10]) and GoogLeNet architectures[11]. We employ transfer learning[12] to fine tune these networks (Sec. 2.4.2) for discretizations of $2^{\{2,5,8\}}$, and report the results in Sec.3.3. In addition to these quantitative metrics, we also include visualizations of the effect of these adversarial perturbations on images from Tiny-Imagenet[13] in Sec. 3.3. We end with a broader discussion of the potential role for dimensional reduction techniques in image classification, and discuss future avenues of research in Sec. 4.

2. Methods

Here we discuss the discretization of inputs as well as the different network architectures and training methods used throughout this report.

2.1. Discretization of pixel intensities

While there are $256^3 \approx 2 \times 10^7$ possible colors which can be represented by an image array, it is thought that the human eye can only perceive around half of these. Furthermore, everyday experience suggests that there is a huge amount of redundancy in these colors - one does not need to know the difference between cobalt and azure (shades of blue) to recognize an image of the ocean. A simple way to represent this redundancy is by discretizing the intensities of each color channel. Throughout this work, we convert the raw integer pixel intensities $0 \leq I \leq 255$ to δ possible values I_δ using the equation

$$I_\delta = \left\lfloor \frac{I}{(256/\delta)} \right\rfloor \left(\frac{256}{\delta}\right) + \frac{1}{2}\left(\frac{256}{\delta}\right), \qquad (1)$$

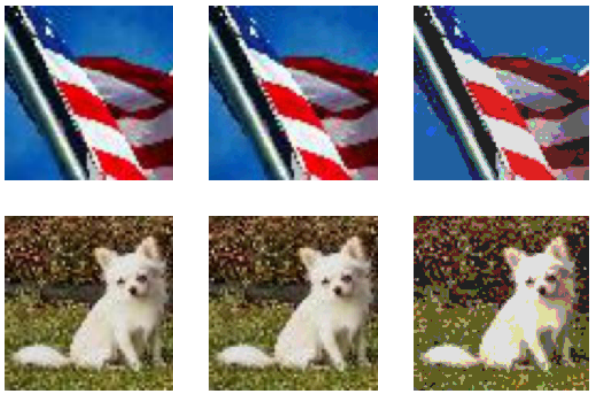
where $\lfloor\ \rfloor$ denotes integer (floored) division. For an RGB image of size $N \times N$ pixels, this reduces the input manifold from 256^{3N^2} to δ^{3N^2}. Some examples of what discretization does to input images is shown in Fig. 2

2.2. Generating adversarial examples

In this work, we generate adversarial examples by a method known as the 'fast gradient method' [4] which computes the gradient of the loss function $\mathcal{L}(\theta, \boldsymbol{y}, \boldsymbol{x})$, with respect to the input image $\boldsymbol{x}$,

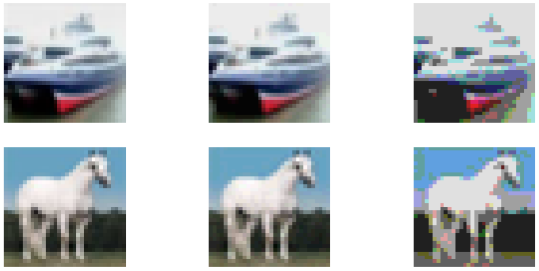
$$\boldsymbol{x}' \rightarrow \boldsymbol{x} - (\epsilon \times 256)\,\mathrm{sign}[\nabla_{\boldsymbol{x}}\mathcal{L}(\theta, \boldsymbol{y}, \boldsymbol{x})] \qquad (2)$$

where $\boldsymbol{y}$ is the classification vector and θ are the parameters (weights, biases etc.) of the network, and $\mathcal{L}$ is the loss function. Note $\nabla_{\boldsymbol{x}}\mathcal{L}(\theta, \boldsymbol{y}, \boldsymbol{x})$ is direction of steepest descent,



(a) $\delta = 256$ (b) $\delta = 32$ (c) $\delta = 4$

Figure 2: The effect of increasing discretization of pixel intensities (δ is the number of allowed intensities) on two randomly chosen (upscaled) 224×224 RGB images from the Tiny-Imagenet dataset



(a) $\delta = 256$ (b) $\delta = 32$ (c) $\delta = 4$

Figure 3: Same as Fig 2 but for the CIFAR10 dataset (32×32 RGB images)

and as such a distortion of the input image in this direction causes the fastest change in the loss function (and hence misclassification). The parameter ϵ quantifies the magnitude of the distortion, and we will plot accuracies as a function of this parameter to quantify the robustness of the network. Note that ϵ's of ~ 0.1 have been found to generate errors rates of 87% on CIFAR10 [4].

Adversarial examples $\boldsymbol{x}'$ are quantified by their magnitude of distortion $|r|$ from an original image $\boldsymbol{x}$ (both of which are vectors of dimension D) using the expression

$$|r| = \frac{1}{D}\sum_{n=1}^{D} ||x'_n - x_n||_2^2 \tag{3}$$

Because a given distortion of the image involves changing *all* pixel values by $\epsilon \times 256$ according to Equation 2, this measure of the distortion corresponds well to a human's perception of the distortion to the image (as opposed to displacing a single pixel by a huge amount, which would not affect our perception of the image, but would correspond to large $|r|$).

2.3. Dataset

We use two standard computer vision datasets throughout this project. Tests were initially carried out on the CIFAR10 data set [8] which contains 50 000 training and 10 000 test images, corresponding to 10 categories. Each image has a resolution of 32×32 pixels, and some examples are shown in Fig. 3.

For subsequent analysis using deeper networks, we used the Tiny-Imagenet dataset [13] which consists of 100 000 training and 10 000 validation images corresponding to 200 categories. For use with CaffeNet and GoogLeNet architectures, we first upscaled all images to a resolution of 224×224 using SciPy's bilinear interpolation scheme[14]. Some example images with different discretizations are shown in Fig. 2.

2.4. Network architectures and training

All networks were implemented in Caffe[9], and with training done using Nvidia GPUs with 4GB memory (on AWS servers). The details of each network are presented here.

2.4.1 5-Layer ConvNet on CIFAR10

For the CIFAR10 dataset we use a five layer network, consisting of three repeats of a [CONV3-32→BATCHNORM→RELU→POOL2] layer, followed by two fully connected layers. The outputs of the top fully connected layer are 10 numbers f_j, $j \in [0, 10)$, which we treat as unnormalized log probabilities, and so we use a softmax classifier to compute the loss function $\mathcal{L}(\boldsymbol{y}, \boldsymbol{x}) = \frac{1}{M}\sum_i \mathcal{L}_i$ where i indexes each of M images, and

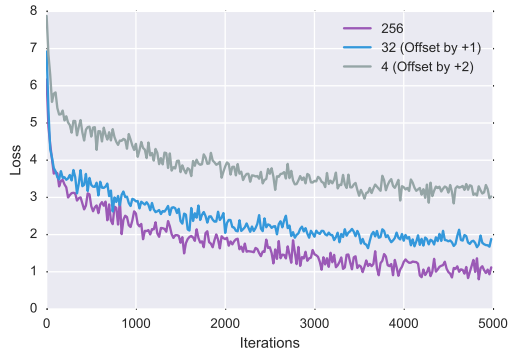
$$\mathcal{L}_i(y_i, x_i) = -\log\left(\frac{e^{y_i}}{\sum_j e^{f_j}}\right), \tag{4}$$

where y_i is the correct class label of image x_i.

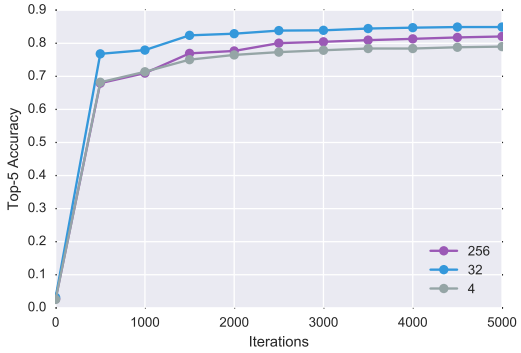
For each discretization of the input images, we use batch gradient descent with the 'Adam' update rule[15]. This is currently an industry recommended default learning rate method[16] because of its per-parameter tuning of the learning rate, and successful combination of several previous update rules like RMS Prop and Momentum. Batch sizes of 100 images were used, and we decreased the learning rate by 90% every 2 epochs. Good results were obtained after training for 10 epochs (5000 iterations).

2.4.2 Transfer learning with CaffeNet and GoogLeNet

For the richer Tiny-Imagenet dataset, a deeper network architecture is necessary. While training deep networks usu-



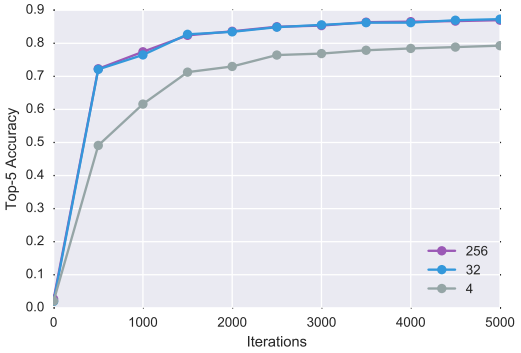
(a) Evolution of the loss for different discretizations of the input.



(b) Top-5 Accuracy for different discretizations.

Figure 4: Loss and accuracy evolution of CaffeNet in the training phase.

(a) Evolution of the loss for different discretizations of the input.



(b) Top-5 Accuracy for different discretizations.

Figure 5: Loss and accuracy evolution of GoogLeNet in the training phase.

ally takes on the order of weeks, a commonly used strategy is transfer learning. We essentially take a pre-trained network and replace the data and final fully connected layers with our data set and a new fully connected layer. The pre-trained layers are used as feature extractors, and within a couple epochs of training we are able to achieve high top-5 accuracies on the 10k validation images of Tiny-Imagenet.

We first experimented CaffeNet, a modified version of AlexNet where pooling is done before normalization. There are five convolutional layers and 2 fully connected layers, with a softmax classifier once more. We implement transfer learning for 5 epochs (5000 iterations at a batch size of 100), using the Adam update rule once again.

Next, we applied transfer learning to GoogLeNet, which is a much deeper network. We trained only the (3) fully connected layers corresponding to the three different output accuracies of the network, for a just over 1 epoch (5000 iterations with a batch size of 24).

Examples of the loss functions and accuracy vs. iterations for both CaffeNet and GoogLeNet are shown in Fig. 4 and 5

We note that due to the significant expense of training these networks on AWS clusters, there was little room for experimentation with many different hyperparameters. The strategy for picking the overall learning rate (typically near to 0.001) was to tune the learning rate magnitude until a decreasing loss function was obtained. Limited experimentation on randomly chosen learning rates produced roughly the same final top-5 accuracies, and the best results are reported here. For the same reasons, we only tested three different discretizations, corresponding to $2^{2,5,8}$ allowed pixel intensities. No cross-validation was employed.

3. Results and Discussion

3.1. Metrics used

Because we are focused on the adversarial robustness of classifiers, our primary metric is the accuracy of the various networks used, as a function of both the discretization, δ, and the strength of the adversarial perturbation. For the CIFAR10 data set we present the top-1 (regular) accuracy on the test set of 10 000 images, while for the Tiny-Imagenet we use both top-5 and top-1 accuracies on the validation set of 10 000 images. For qualitative results (where we show

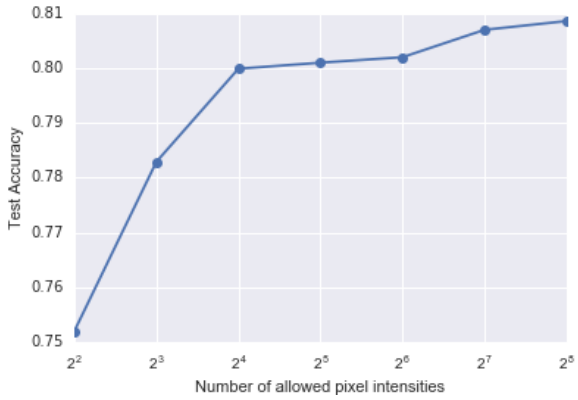


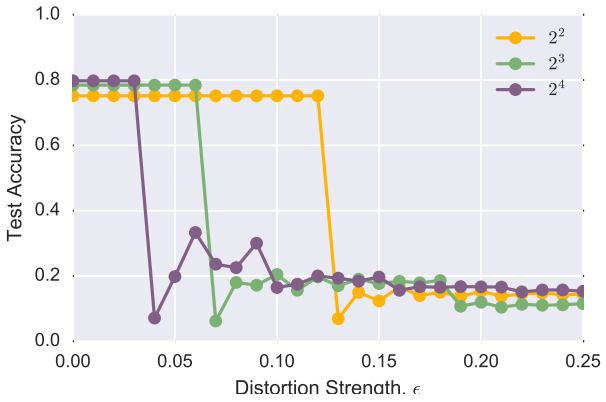
Figure 6: Evolution of the accuracy of the ConvNet as the discretization δ is changed. A remarkable outcome is the fact that with only 16 allowed intensities, the classification accuracy has only dropped by 1%. This corresponds to a minimum allowed perturbation of 8 units of intensity.

adversarial distortions), we also include the magnitude of the distortion $|r|$ as defined in Eq. 3

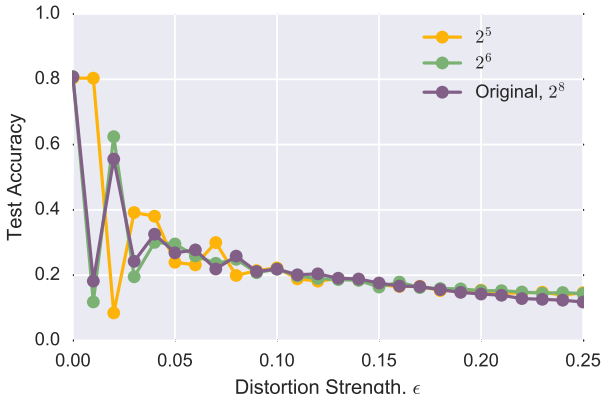
3.2. Fully trained ConvNets on CIFAR10

The results on CIFAR10 are encouraging. As shown in Fig. 6, the accuracy of the classifier only drops by 6% on going from all 256 allowed intensities to just 4. Furthermore, there is a broad regime from 16 to 256 allowed intensities where the accuracy varies by less than 1%. This clearly validates the hypothesis that input dimension of images typically used is unnecessarily large, and can be reduced significantly.

In Fig. 7 we show the classification accuracy on the test set as the strength of the distortion in a fast gradient method is increased. As explained in the caption of this figure, the discretized images are 'trivially' robust to small perturbations - this is an explicit feature of our discretization of the inputs. One obvious feature of these plots is the immediate collapse of the accuracy down to 10% (consistent with random draws) upon perturbing beyond the minimum allowed distortion. While unfortunate, this can be expected on general grounds, following our earlier discussion of the difference between discriminative and generative models. This is still a discriminative method, and so there are still vast regions of the input space which are being classified (despite their lack of natural data). Thus, as soon as images are perturbed into these regions (with a minimum allowed perturbation), the classifier fails completely. A more subtle feature of Fig. 7 is the seeming recovery of the accuracy (jumps in accuracy) just beyond the minimum allowed perturbation. This makes little physical sense, and will be the



(a) Allowed pixel intensities $\delta = 4, 8, 16$



(b) δ =32, 64, 256

Figure 7: **The adversarial robustness upon discretization:** evolution of the accuracy of the classifiers as the parameter ϵ (magnitude of the distortion strength in Eq. 2 is increased. Note that the discretized images are trivially robust. For example for $\delta = 4$, it is only when $\epsilon = 0.125 * 256 = 32$ that a perturbation is allowed. Below this, any small pixel perturbation is clamped back to zero. Note that above $\epsilon = 0.125$, the classifier's all fail completely, with the accuracy of 10% being equivalent to random luck.

subject of future investigation.

Perhaps the most intriguing feature of the CIFAR10 data is shown in Fig. 8. Here, we show the accuracies of ConvNets trained on images with one discretization and tested on images with another discretization. There is a broad region from 16 allowed intensities upwards, where the accuracies are roughly the same. This is further confirmation of the unnecessarily large dimension of images - for classification tasks, only a few colors are necessary, and discretization acts as form of regularization.

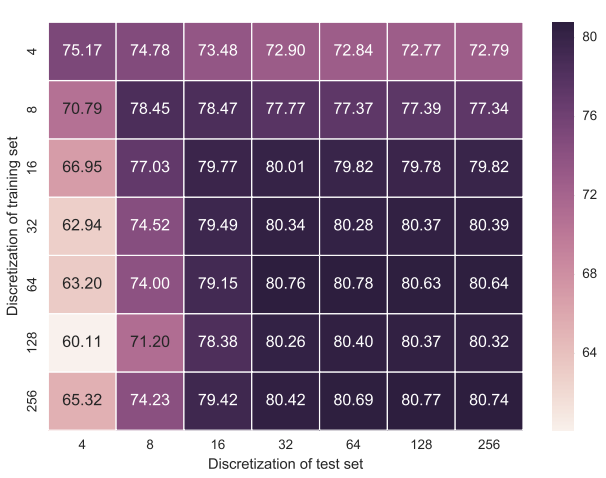


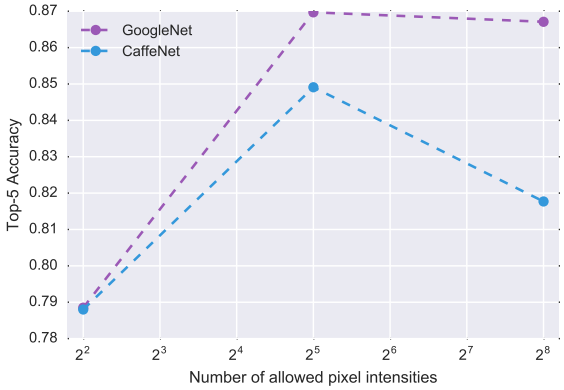
Figure 8: The accuracies for classifiers trained on one level of dicretization and tested on another. Classifiers trained on inputs with a 16 or more allowed intensities all appear to produce similar accuracies (indicating that the weights and biases in these models are similar!)

3.3. Transfer learning on Tiny-ImageNet 200

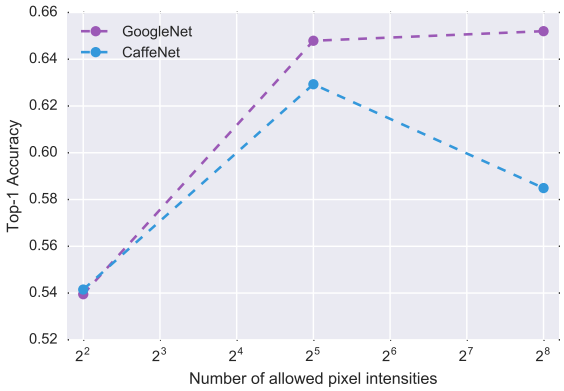
For the Tiny-ImageNet data set, the results are remarkably similar, despite the 20 fold increase in the number of categories being classified. In Fig. 9 we show the top-5 and top-1 accuracies for both CaffeNet and GoogLeNet trained via transfer learning. First note that GoogLeNet outperforms CaffeNet in both top-5 and top-1 accuracies as can be expected - it is a deeper network, and when trained on Imagenet achieved a top-5 accuracy of 89% (as opposed to 80% for CaffeNet).

However, note that there is a remarkable *increase* in the accuracies for CaffeNet upon going from 256 to 32 pixels, while the eventual decrease in top 5 accuracies at 4 allowed pixel intensities is moderate. While a more comprehensive exploration of training hyperparameters may change this result, initial attempts to change it were unsuccessful. This possibly supports the notion of discretization as a form of regularization. On the other hand, GoogLeNet's top-5 and top-1 accuracies were barely different between 256 and 32 allowed intensities, with a moderate (but not catastrophic) drop in accuracy upon going to 4 allowed intensities.

Finally, a useful qualitative evaluation of the effectiveness of discretization to adversarial examples is shown in Fig. 10, where we show the raw images, the minimum perturbation as defined by Eq. 2 and the resulting distorted image. It is clear that the minimum distortion is largest for the discretized dataset, and the resulting distorted image is least recognizable (to humans) for this large discretization. This



(a) Top-5 accuracies



(b) Top-1 accuracies

Figure 9: Accuracies for CaffeNet and GoogLeNet upon changing the discretization of input images.

is a concrete manifestation of the adversarial robustness we have achieved by discretization.

4. Conclusions and Future Work

We have demonstrated that the accuracies of deep neural networks are not adversely affected by discretization of input data. For a reduction of input pixel intensities from 256 to 16, there is a minimal drop in accuracy (1% for CIFAR10), and for 4 allowed intensities the accuracy drops by less than 10%. This type of discretization then renders images trivially robust to adversarial perturbations - because only discrete values of intensities are permitted, there is a minimum allowed distortion to each image before it can change. There is then a natural tradeoff between increasing the level of discretization (i.e. reducing the number of allowed pixel intensities), and the decreasing accuracy of the ConvNet. The broad regime of similar performance that is evident in CIFAR10 results (and supported by the Tiny-Imagenet experiments) suggests that a 'sweet-spot' of dis-

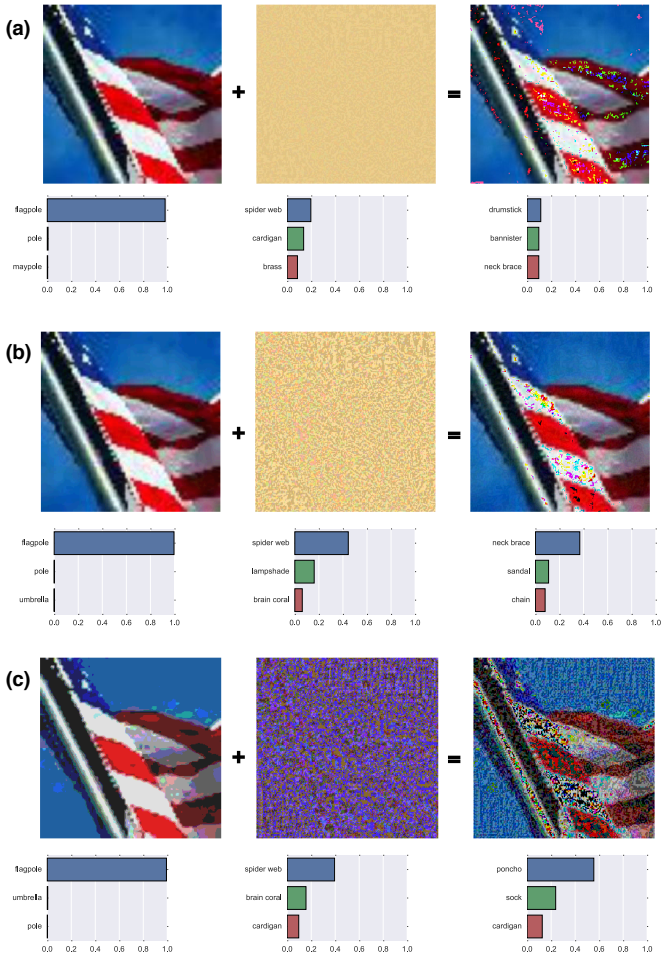


Figure 10: The minimum allowed distortion for different discretizations of a randomly chosen Tiny-Imagenet validation image, with the probabilities as assigned by GoogLeNet. The values of δ are (a): 256, (b):32 and (c):4. The distortion magnitude $|r|$ is respectively 1, 4 and 32. The noise images have been right shifted by 128.

cretization around 16-32 allowed pixel intensities will result in minimal loss of accuracy and maximal robustness to adversarial perturbations.

More broadly, it is clear that this discretization procedure has acted as a form of regularization. By reducing the dimensionality of the input data, we have even improved the top-5 accuracy of CaffeNet and GoogLeNet on Tiny-ImageNet. This reduction in input dimensionality may also be viewed as an inflexible prior distribution on the input data distribution. We note that unlike other forms of input dimensional reduction (e.g. PCA) the discrete form of the data is crucial, as it results in stability to perturbations.

Future areas of investigation include completing this analysis on Imagenet data, and investigating more values of the discretization to search for an optimal value which maximizes robustness without sacrificing accuracy. This form of discrete compression and improved robustness of images may also be valuable for industries where security and high throughput of data are important (e.g. in self driving cars). More generally, techniques for learning the true underlying distributions of image data in high dimensional spaces (and perhaps with the inclusion of dimensional reduction techniques such as that presented here) appears to be an important avenue for research into deep neural networks.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[2] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[3] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 427–436. IEEE, 2015.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[7] Matthieu Courbariaux and Yoshua Bengio. Binarynet: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016.

[8] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images, 2009.

[9] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[12] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[13] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[14] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[15] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Andrej Karpathy, Justin Johnson, and L. Fei-Fei. Cs231n lecture notes, 2015.