IMPROVING FREE ADVERSARIAL TRAINING USING SCHEDULING, AUGMENTATIONS, AND DENOISING

Oleksii Volkovskyi
volkovskyi@berkeley.edu

Nadia Hyder
nhyder@berkeley.edu

Michael Zhu
michaelbzhu@berkeley.edu

1 Introduction

We build a robust classifier, one that is able to correctly label adversarially perturbed images. We specifically work with the Tiny ImageNet dataset that consists of 100,000 64x64 images broken down into 200 classes. Robustness is a broad generalization of the many perturbations that can occur in real world scenarios. Some of these perturbations may include distribution shifts such as changes in lighting, perception angles, rendition style, geography, deep learning augmentation techniques, and others [Hendrycks et al., 2020].

As a baseline for accuracy, we evaluated our model using the public Tiny ImageNet validation set which contains 50 images per class. Additionally, we performed a set of adversarial augmentations to the validation set and recorded the performance of a baseline model as a benchmark.

We performed adversarial training in a low compute regime, which poses unique challenges and limits the range of models and training procedures we are able to implement. We do this at no cost to the training accuracy, trading off longer training time for training accuracy. All of the presented results were trained on 1-3 Tesla P100 GPUs in under 24 hours. We did not find any significant improvement from longer training since the validation accuracy stopped improving after approximately 20 epochs.

2 Related Work

There are many different distribution shifts possible in real world data and as a result, the field of image classification robustness has varied opinions on which evaluation metrics are the most important. Most research focuses on improving robustness for a few of the possible distribution shifts but no current method consistently improves robustness across all distribution shifts.

A simple and popular approach is to use geometric data augmentation during training [Shorten and Khoshgoftaar, 2019]. This process comprises randomly applying transformations such as cropping, flipping, rotation, and color shifts to the training data. These are often used as a baseline for evaluation because they are commonly seen in real world data, are easy to implement, and have low training speed/memory costs. Another data augmentation approach proposed by Hendrycks et al. [2020] called DeepAugment uses an image to image autoencoder to augment training data. They incorporate random distortions to the network weights to generate diverse but semantically consistent training images.

These data augmentation techniques do not protect against adversarial attacks, since the random noise vectors added to the images are rarely in the adversarial direction [Warde-Farley, 2016]. This leads the underlying trained models to be highly susceptible to FGSM and other gradient based attacks. Adversarial attacks such as the fast gradient sign method [Goodfellow et al., 2015] have been shown to be effective at thwarting image models with minor perturbations that most human observers would be robust to. Kurakin et al. [2017] extends the fast gradient sign method into an iterative attack (known as BIM) that performs multiple iterations of maximizing the loss while keeping the perturbation to a small scale. Although these attacks are much stronger and don't have a fully robust defense, it is also much harder to construct such an attack against a model without accessing it's gradients.

To defend against BIM, Shafahi et al. [2019] propose a fast training procedure that defends against adversarial gradient attacks by perturbing training images with a running average of the adversarial gradient direction. One problem with this algorithm is the moving target optimization objective prevents the classifier from learning without expensive hyper-parameter tuning. In our work, we will extend the "Free" Adversarial Training algorithm to have lower variance while training by using scheduling tricks and exploring an auto-encoder architecture to train a noise filter and classifier together end-to-end.

3 Background

We built upon ResNet18 with weights pre-trained on the ImageNet dataset for transfer learning. This convolutional neural network is 18 layer deep, pretrained on more than a million images from the ImageNet database. To avoid overfitting and exploding and vanishing gradients, residual networks use skip connections that add the input to the output of the affine layers. This allows gradients to flow through shortcut connections to previous layers and allows for deeper models.

Empirically, we found that allowing the entire network to be fine-tuned on the TinyImageNet dataset resulted in much better performance. We hypothesize that this may be because the convolutional kernels need to be adapted to the lower resolution images and strong adversarial inputs. Additionally, we replaced the final softmax layer with three 512x512 linear layers with dropout in order to prevent overfitting by reducing the variance in the final layers, thus being able to deal with gaussian noise in the inputs.

Because ResNet18 has a 224x224 input image size, we scaled the 64x64 Mini ImageNet input images up using bilinear interpolation. Naturally, this magnified noise that we trained our model to deal with.

We heavily reference the "Free" adversarial training algorithm by Shafahi et al. [2019] and build upon it with the goals of getting better training properties and faster convergence under a low-compute regime.

4 Methods

For our baseline model, we used a pretrained ResNet-18 without any data augmentation. We also tested larger ResNet models such as ResNet-50 and ResNet-101, but found that ResNet-18 had the best performance on the validation set. We hypothesize that the larger models performed worse because of the smaller image sizes and possible overfitting.

Perturbed Validation Set We constructed two modified validation sets to evaluate the robustness of our model on perturbed data. The first validation dataset contains only geometrically augmented images. The second dataset contains images modified using FGSM method using the baseline model above to derive gradients from, and with additional geometric noise from Torchvision transforms.

4.1 Geometric augmentation

We added geometric augmentations to our pre-training process, which significantly improved our baseline model's accuracy on the validation dataset. We found that hue, saturation, and rotation augmentations to have the strongest effect on accuracy improvements.

Contrastive Loss Siamese loss compares the similarity between two image encodings. It's commonly used in facial recognition, where the model would want to produce similar encodings for the same person's face. We hypothesized that this concept could be applied to perturbed images in order to increase robustness. Along with the standard cross entropy loss in image classification, we added a second loss term that computes the contrastive loss between two perturbations of the same image. For each training image, we would perform random geometric perturbations twice to generate two augmented training images. Each image would be passed into our model to generate an encoding vector and our model would compare the two encoding vectors to compute the contrastive loss. Ideally, this would teach our model to ignore the noisy perturbations and pay closer attention to the semantic information in the images.

Denoising Image to Image Autoencoder Instead of teaching our network to ignore perturbations through contrastive loss, we wondered if we could directly remove harmful noise prior to training the classifier model. Inspired by Rusak et al. [2020], who trained a second network to generate adversarial noise, we decided to go the opposite direction and train a second network to aid the classifier network in noise removal. This was implemented by adding an image-to-image autoencoder that takes in perturbed images as input and outputs a denoised image that is fed into the classifier. We trained the denoising autoencoder using a loss function that compares its output image to the original pre-augmentation training image. The pertubations applied to the input images were both geometric transforms as well as FGSM attacks from the classifier currently being trained. This allowed us to train two models in the same training loop whilst reusing the gradients of the classifier to construct data for the autoencoder.

4.2 Adversarial examples

FGSM Training We implemented the free adversarial training algorithm proposed by Shafahi et al. [2019] that defends against attacks from the free gradient sign method. They use an inner loop during training to compute and keep track of adversarial gradient directions δ. This δ is then summed together with input images prior to training to give the model exposure to adversarial noise that could be used to attack the model at test time.

Improvements on Free Adversarial Training In the original paper by Shafahi et al. [2019], the δ values are either ϵ, 0, or $-\epsilon$. We attempted to smooth out the δ values using an exponential average to build a better adversarial direction throughout the training process and construct stronger adversarial examples.

However, smoothing out the δ using a moving average only reduced the strength of the adversarial attacks, which caused the model to train slower to reach benchmark performance on the FGSM augmented dataset. We hypothesize that this is because the adversarial direction is specific to each image and doesn't generalize across classes. Therefore smoothing the delta and allowing it to persist across different batches only serves to weaken the effectiveness of the FGSM attack by essentially feeding in random noise rather than an adversarial gradient.

One problem we ran into while training using Free Adversarial Training was a moving target between the adversarial image and clean image. When trained using a large m, the FGSM attacks dominated the training process due to a large enough ϵ and stopped the model from training well on the natural images.

We explored two solutions to this issue.

In one solution, we added a γ term that controls how often our model trains using δ. In the implementation proposed by Shafahi et al. [2019], they applied δ to every training example. By limiting the usage of δ in our training loop, we improved our model's accuracy on clean data while retaining the ability to defend against some adversarial attacks, albeit less effectively. We believe that this is a necessary trade-off because adversarial examples are significantly less common in a real world setting.

The second and significantly better solution we discovered was adding scheduling the m and ϵ terms to increase throughout the training process. This would allow the model to train on mostly natural data at the start and slowly be fine-tuned against adversarial attacks throughout training. We were inspired by the slow moving target DQN in RL. We observed a significant improvement in training. This ended up being the final model we submitted.

5 Results

5.1 Geometric augmentation

When evaluating our models on geometrically augmented validation data, we found that the model with high contrastive loss (High CL Loss Aug. on Figure 1) performed the best. We compared it with our baseline model, data augmented model, and low contrastive loss model (Contrastive Aug. on Figure 1). The high contrastive loss model differs from the low contrastive loss model in that it places a larger weight on the contrastive loss. These results show that adding constrastive loss can significantly increase a model's robustness towards geometric perturbations.

5.2 Adversarial Examples

We generated adversarial examples by performing fast gradient sign method on the public Tiny ImageNet validation set with our baseline ResNet18 model. We found that our model trained with a scheduled FGSM inner loop performed the best compared to unscheduled FGSM training, moving average FGSM training, and the denoising autoencoder.

We were surprised to find that the denoising autoencoder performed so poorly compared to the FGSM inner loop methods. We hypothesize that this may be because the autoencoder trivially learns the identity mapping without necessarily denoising. Instead, it just linearly transforms the noise (since neural networks are locally linear).

The results show that our change to use scheduled FGSM training does improve upon the FGSM training algorithm proposed by Shafahi et al. [2019]. This is likely due to the fact that scheduling m and ϵ doesn't harm the training process in the initial few epochs; but ramps up in later epochs in order to succesfully defend against FGSM attacks.

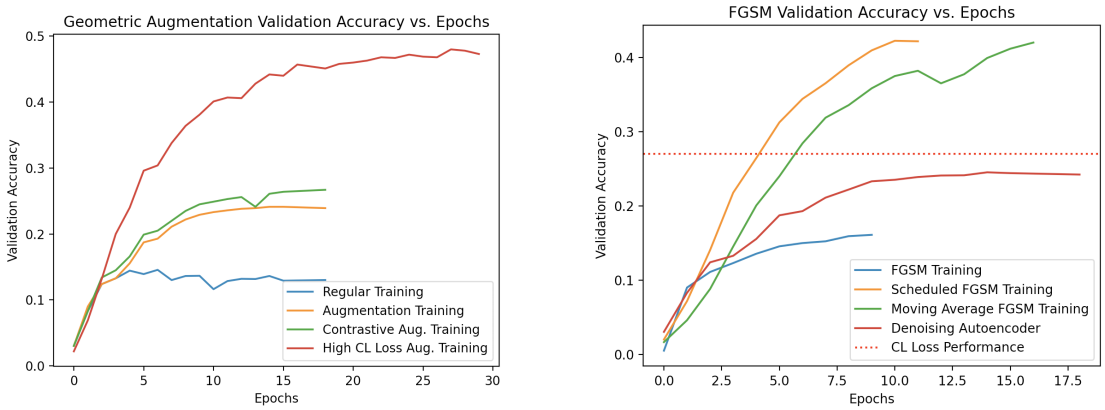

Figure 1: Validation accuracy by model

Table 1: Model Accuracies

Model	Geometrically Augmented Validation Accuracy	FGSM Augmented Validation Accuracy
Fine-Tuned ResNet18	32%	N/A
Fine-Tuned ResNet50	20%	N/A
ResNet18 w/ Data Aug	35%	N/A
ResNet18 w/ Data Aug + Contrastive Loss	**48%**	28%
ResNet18 w/ Data Aug + CL + Inner FGSM	N/A	16%
ResNet18 w/ Data Aug + CL + Scheduled FGSM	N/A	**42%**
ResNet18 w/ Data Aug + CL + Autoencoder	N/A	24%

6 Conclusion

We were able to drastically improve the training loop of Free Adversarial Training by using a scheduled m and ϵ to reduce the initial moving target effect of large adversarial δ's on the training data. This made the training process more stable and allowed us to converge to results much faster.

We proposed training an ensemble using an autoencoder structure to map an adversarially augmented image to the normal image manifold. This relied on training two networks within one training loop - a classifier that also provided us with FGSM adversarial examples and an autoencoder that aimed to reconstruct the perturbed image to the original. We were surprised to find out that it performed poorly and learned a noisy identity transform that did not significantly eliminate geometric or FGSM augmentations.

We learned that the adversarial direction is image specific and did not generalize between classes in our experiments. This significantly reduced the effectiveness of one of our novel additions to FGSM training - a moving average δ term.

Further approaches could look into caching specific adversarial directions or eliminating extremely compute intensive geometric transformations that tanked our training times. We expected that most of the adversarial examples in the code would be geometric transforms so we heavily focused on those.

We were compute bound throughout the project and were not able to train a few of the networks to convergence or run grid-search on hyperparamaters such as those involved in FGSM training scheduling or trade-off between cross-entropy loss and contrastive loss, so it would be a naturally great extension.

7 Team Contributions

Michael and Nadia helped research papers to reference for adversarial training, robustness, and possible evaluation metrics. Oleksii was responsible for implementing data augmentations, contrastive loss, FGSM training, and running experiments to properly ablation test the hypotheses. Everyone helped run experiments to train and evaluate models. Oleksii 45%, Michael 35%, Nadia 25%

References

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization, 2020.

Connor Shorten and T. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6: 1–48, 2019.

David Warde-Farley. 1 adversarial perturbations of deep neural networks. 2016.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world, 2017.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free!, 2019.

Evgenia Rusak, Lukas Schott, Roland S. Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions, 2020.