Efficient Adversarial Training with Transferable Adversarial Examples

Haizhong Zheng  Ziqi Zhang  Juncheng Gu  Honglak Lee  Atul Prakash
University of Michigan, Ann Arbor
{hzzheng, ziqizh, jcguz, honglak, aprakash}@umich.edu

Abstract

Adversarial training is an effective defense method to protect classification models against adversarial attacks. However, one limitation of this approach is that it can require orders of magnitude additional training time due to high cost of generating strong adversarial examples during training. In this paper, we first show that there is high transferability between models from neighboring epochs in the same training process, i.e., adversarial examples from one epoch continue to be adversarial in subsequent epochs. Leveraging this property, we propose a novel method, Adversarial Training with Transferable Adversarial Examples (ATTA), that can enhance the robustness of trained models and greatly improve the training efficiency by accumulating adversarial perturbations through epochs. Compared to state-of-the-art adversarial training methods, ATTA enhances adversarial accuracy by up to 7.2% on CIFAR10 and requires $12 \sim 14 \times$ less training time on MNIST and CIFAR10 datasets with comparable model robustness.

1. Introduction

State-of-the-art deep learning models for computer vision tasks have been found to be vulnerable to adversarial examples [27]. Even a small perturbation on the image can fool a well-trained model. Recent works [7, 25, 26] also show that adversarial examples can be physically realized, which can lead to serious safety issues. Design of robust models, which correctly classifies adversarial examples, is an active research area [5, 9, 14, 21, 30], with adversarial training [18] being one of the most effective methods. It formulates training as a game between adversarial attacks and the model: the stronger the adversarial examples generated to train the model are, the more robust the model is.

To generate strong adversarial examples, iterative attacks [13], which use multiple attack iterations to generate adversarial examples, are widely adopted in various adversarial training methods [4, 18, 28, 34]. Since adversarial perturbations are usually bounded by a constrained space $S$ and attack perturbations outside $S$ need to be projected back to $S$, $k$-step projected gradient descent method [13, 18] (PGD-$k$) has been widely adopted to generate adversarial examples. Typically, using more attack iterations (higher value of $k$) produces stronger adversarial examples [18]. However, each attack iteration needs to compute the gradient on the input, which causes a large computational overhead. As shown in Table 1, the training time of adversarial training can be close to 100 times larger than natural training.

Recent works [17, 19, 20] show that adversarial examples can be transferred between models: adversarial examples generated for one model can still stay adversarial to another model. The key insight in our work, which we experimentally verified, is that, because of high transferability between models (i.e., checkpoints) from neighboring training epochs, attack strength can be accumulated across epochs by repeatedly reusing the adversarial perturbation from the previous epoch.

We take advantage of this insight in coming up with a novel adversarial training method called ATTA (Adversarial Training with Transferable Adversarial examples) that can be significantly faster than state-of-the-art methods while achieving similar model robustness. In traditional adversarial training, when a new epoch begins, the attack algorithm generates adversarial examples from the original input images, which ignores the fact that these perturbations can be reused effectively. In contrast, we show that it can be advantageous to reuse these adversarial perturbations through epochs. Even using only one attack iteration to generate adversarial examples, ATTA-1 can still achieve comparable robustness with respect to traditional PGD-40.

We apply our technique on Madry’s Adversarial Training method (MAT) [18] and TRADES [34] and evaluate the performance on both MNIST and CIFAR10 dataset. Compared with traditional PGD attack, our method improves training efficiency by up to $12.2 \times (14.1 \times)$ on MNIST (CIFAR10) with comparable model robustness. Trained with ATTA, the adversarial accuracy of MAT can be improved by up to 7.2%. Noticeably, for MNIST, with one attack iteration, our method can achieve 96.31% adversarial accuracy within 5 minutes (compared to 65 minutes with MAT). For CIFAR10, compared to MAT whose training time is...
more than one day, our method achieves comparable adversarial accuracy in about 2 hours (compared to 33 hours with MAT). We also verify that our method is practical for large dataset: we evaluate our method with ImageNet, which can achieve comparative robustness within 97 hours with just one 2080 Ti GPU. We also compare our method with two fast adversarial training methods, YOPO [32] and Free [23]. Evaluation results show that our method is both more effective and efficient than these two methods.

**Contribution.** To the best of our knowledge, our work is the first to enhance the efficiency and effectiveness of adversarial training by taking advantage of high transferability between models from different epochs. In summary, we make the following contributions:

- We are the first to reveal the high transferability between models of neighboring epochs in adversarial training. With this property, we verify that the attack strength can be accumulated across epochs by reusing adversarial perturbations from the previous epoch.

- We propose a novel method (ATTA) for iterative attack-based adversarial training with the objectives of both efficiency and effectiveness. It can generate the similar (or even stronger) adversarial examples with much fewer attack iterations via accumulating adversarial perturbations through epochs.

- Evaluation result shows that, with comparable model robustness, ATTA is 12.2 × (14.1 ×) faster than traditional adversarial methods on MNIST (CIFAR10). ATTA can also enhance model adversarial accuracy by up to 7.2% for MAT on CIFAR10.

2. Adversarial training and transferability

In this section, we introduce relevant background on adversarial training and transferability of adversarial examples. We also discuss the trade-off between training time and model robustness.

2.1. Adversarial Training

Adversarial training is an effective defense method to train robust models against adversarial attacks. By using adversarial attacks as a data augmentation method, a model trained with adversarial examples achieves considerable robustness. Recently, lots of works [2, 10, 11, 18, 22, 33, 34] focus on analyzing and improving adversarial machine learning. Madry et al. [18] first formulate adversarial training as a min-max optimization problem:

$$\min_{f \in \mathcal{H}} \mathbb{E}_{(x, y) \sim \mathcal{D}} \left[ \max_{\delta \in \mathcal{S}} L(f(x + \delta), y) \right]$$

where $\mathcal{H}$ is the hypothesis space, $\mathcal{D}$ is the distribution of the training dataset, $L$ is a loss function, and $\mathcal{S}$ is the allowed perturbation space that is usually selected as an $L_p$ norm ball around $x$. The basic strategy in adversarial training is, given a original image $x$, to find a perturbed image $x + \delta$ that maximizes the loss with respect to correct classification. The model is then trained on generated adversarial examples. In this work, We consider adversarial examples with a high loss to have **high attack strength.**

**PGD-$k$ attack based adversarial training:** Unfortunately, solving the inner maximization problem is hard. Iterative attack [13] is commonly used to generate strong adversarial examples as an approximate solution for the inner maximization problem of Equation 1. Since adversarial perturbations are usually bounded by the allowed perturbation space $\mathcal{S}$, PGD-$k$ ($k$-step projected gradient descent [13]) is adopted to conduct iterative attack [18, 23, 32, 34]. PGD-$k$ adversarial attack is the multi-step projected gradient descent on a negative loss function:

$$x_{t+1} = \Pi_{x \in \mathcal{S}}(x_t + \alpha \text{sgn}(\Delta_{x_t}(\theta, x_t), y))$$

In the above, $x_t$ is the adversarial example in the $t$-th attack iteration, $\alpha$ is the attack step size, and $\Pi$ is the projection function to project adversarial examples back to the allowed perturbation space $\mathcal{S}$.

With a higher value of $k$ (more attack iterations), PGD-$k$ can generate adversarial examples with higher loss [18]. However, there is a trade-off between training time and model robustness in adversarial training. On one hand, since each attack iteration needs to calculate the gradient for the input, using more attack iterations requires more time to generate adversarial examples, thus causing a large computational overhead for adversarial training. As shown in Table 1, compared to natural training, adversarial training may need close to 100x more training time until the model converges. Most of the training time is consumed in generating adversarial examples (attack time). On the other hand, reducing the number of attack iterations $k$ can reduce the training time, but that negatively impacts robustness of the trained model (Table 2).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Natural Training</th>
<th>Adversarial Training</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Training</td>
</tr>
<tr>
<td>MNIST</td>
<td>39.7 sec</td>
<td>210 sec</td>
</tr>
<tr>
<td>CIAFR10</td>
<td>55 min</td>
<td>214 min</td>
</tr>
</tbody>
</table>

Table 1: Training time of natural training and adversarial training. **Attack** column shows the time consumed in adversarial example generation.

2.2. Transferability of Adversarial Examples

Szegedy et al. [27] show that adversarial examples generated for one model can stay adversarial for other models. This property is named as transferability. This property is usually leveraged to perform a black-box attack
training epochs, we adversarially train a model should lead to high transferability between these models. If the number of attack iterations in each epoch is very small, we think that they are likely to have epochs of the same training program has not been studied.

3.1. High transferability between epochs

PGD-attack achieves much higher attack strength with fewer number of attack iterations to generate the same strong adversarial examples. Hypothesis: Repeatedly reusing perturbations from the previous epoch can accumulate attack strength epoch by epoch. Compared to current methods that iterate from original images in each epoch, this can allow us to use few attack iterations to generate the same strong adversarial examples.

3.2. Accumulative PGD-k attack

To validate the aforementioned hypothesis, we design an accumulative PGD-k attack. As shown in Figure 2b, we longitudinally connect models in each epoch by directly reusing the attack perturbation of the previous epoch. Accumulative PGD-k attack Figure 2b generates adversarial examples for $f_{n-m+1}$ first. Then, for the following epochs, the attack is performed based on the accumulated perturbation.
tions from previous epochs.

Figure 2: The traditional PGD-\(k\) attack (a) and the accumulative PGD-\(k\) attack through \(m\) epochs (b). The PGD-\(k\) attack is performed on the model in the red rectangle to get adversarial examples.

To compare the attack strength of two attacks, we use Madry’s method [18] to adversarially train two models on MNIST and CIFAR10 and evaluate the loss value \(L(f_n(x^*_n), y)\) of adversarial examples generated by two attacks. Figure 3 summarises the evaluation result. We can find that, with more epochs \(m\) involved in the attack, accumulative PGD-\(k\) attack can achieve a higher loss value with the same number of attack iterations \(k\).

Especially, when adversarial examples are transferred through a large number of epochs, even accumulative PGD-1 attack can cause high attack loss. For MNIST, accumulative PGD-1 attack can achieve the same attack loss as traditional PGD-35 attack when \(m = 40\). For CIFAR10, accumulative PGD-1 attack can achieve the same attack loss as traditional PGD-12 attack when \(m = 10\).

This result indicates that, with high transferability between epochs, adversarial perturbations can be reused effectively, which allows us to use fewer attack iterations to generate the same or stronger adversarial examples. Reuse of perturbations across epochs can help us reduce the number of attack iterations \(k\) in PGD-\(k\), leading to more efficient adversarial training. Next section describes our proposed algorithm, ATT A (Adversarial Training with Transferable Adversarial examples), based on this property.

4. Adversarial training with transferable adversarial examples

The discussion on transferability in Section 3 suggests that adversarial examples can retain attack strength in subsequent training epochs. The results of accumulative attack in Section 3 suggest that stronger adversarial examples can be generated by accumulating attack strength. This property inspires us to link adversarial examples between adjacent epochs as shown in Figure 4. Unlike starting from a natural image \(x\) to generate an adversarial example in each epoch as shown in Figure 4(a), we start from a previously saved adversarial example \(x^*_i\) from the previous epoch to generate an adversarial example (Figure 4(b)). To improve the transferability between epochs, we use a connection function \(C\) to link adversarial examples, which transforms \(x^*_i\) to a start point \(x_{i+1}\) for the next epoch. During the training, with repeatedly reusing adversarial example between epochs, attack strength can be accumulated epoch by epoch:

\[
x_{i+1}^* = A(f_{i+1}, x, y, C(x^*_i))
\]

where \(A\) is the attack algorithm, \(C\) is a connection function (described in the next section). \(f_i\) is the model in the \(i\)-th epoch, \(x^*_i\) is the adversarial examples generated in the \(i\)-th epoch and \(x, y\) are natural image and benchmark label \(^1\).

As shown in the previous section, adversarial examples can achieve high attack strength as they are transferred across epochs via the above linking process, rather than

\(^1\)Note that the \(x_{i+1}^*\) is still bounded by the natural image \(x\), rather than \(C(x^*_i)\).
starting from natural images. This, in turn, should allow us to train a more robust model with fewer attack iterations.

4.1. Connection function design

Designing connection function can help us overcome two challenges that we encountered in achieving high transferability of adversarial examples from one epoch to the next:

1. **Data augmentation problem**: Data augmentation is a commonly used technique to improve the performance of deep learning. It applies randomly transformation on original images so that models can be trained with various images in different epochs. This difference can cause a mismatch of images between epochs. Since the perturbation is calculated with the gradient of image, if we directly reuse the perturbation, the mismatch between reused perturbation and new augmented image can cause a decrease in attack strength. Simply removing data augmentation also hurts the robustness. We experimentally show this negative influence of these two alternatives in Section 5.3.

2. **Drastic model parameter change problem**: As discussed in Section 3, similar parameters between models tends to cause a similar decision boundary and thus high transferability. Unfortunately, model parameters tend to change drastically at the early stages of training. Thus, adversarial perturbations in early epochs tend to be useless for subsequent epochs.

**Overcoming challenges: Inverse data augmentation.**

To address the first issue, we propose a technique called inverse data augmentation so that adversarial examples retain a high transferability between training epochs despite data augmentation. Figure 5 shows the workflow with inverse data augmentation. Some transformations (like cropping and rotation) pad augmented images with background pixels. To transfer a perturbation on background pixels, our method transfers the padded image \( x_i \) rather than standard images so that the background pixels used by data augmentation can be covered by these paddings.

After the adversarial perturbation of augmented image is generated, we can perform the inverse transformation\(^2\) \( T^{-1}(\cdot) \) to calculate the inverse perturbation \( T^{-1}(\Delta_i) \) on the padded image \( x_i \). By adding \( T^{-1}(\Delta_i) \) to \( x_i \), we can store and transfer all perturbation information in \( x_{i+1} \) to next epoch. (Note that, when we perform the adversarial attack on the augmented image \( x_{i,aug} \), the perturbation is still bounded by the natural image rather than \( x_{i,aug} \).)

**Periodically reset perturbation.** To solve the second issue, we propose a straightforward but effective solution: our method resets the perturbation and lets adversarial perturbations be accumulated from the beginning periodically, which mitigates the impact caused by early perturbations.

4.2. Attack Loss

Various attack loss functions are used in different adversarial training algorithms. For example, TRADES [34] uses robustness loss (Equation 2) as the loss to generate the adversarial examples.

\[
\mathcal{L}(f(x^*), f(x))
\]  

where \( \mathcal{L} \) is the loss function, \( f \) is the model and \( x, x^* \) are natural and adversarial example respectively. This loss represents how much the adversarial example diverges from the natural image. Zhang et al. [34] shows that this loss has a better performance in the TRADES algorithm.

In our method, we use the following loss function:

\[
\mathcal{L}(f(x^*), y)
\]  

It represents how much the adversarial examples diverges to the benchmark label.

\(^2\)In adversarial training, most data augmentation methods used are linear transformations which are easy to be inverted.
The objection \( f(x) \) of Equation 2 for adversarial attack varies across epochs, which may weaken the transferability between epochs. Equation 3 applies a fixed objection \( y \), which doesn’t have this concern. In addition, Equation 3 has a smaller computational graph, which can reduce computational overhead during training.

The overall training method is described in Algorithm 1.

**Algorithm 1** Adversarial Training with Transferable Adversarial Examples (ATTA)

1. **Input**: Padded training dataset \( D_{nat} \), model \( f_\theta \), attack algorithm \( A \), perturbation bound \( \epsilon \), the number of epochs to reset perturbation \( reset \)
2. Initialize \( \theta \)
3. Initialize \( D \) by cloning \( D_{nat} \)
4. for epoch = 1 \( \cdots \) \( N \) do
5. for \( x_{nat}, y \) in \( D_{nat} \) and corresponding \( x \in D \) do
6. if epoch \% reset = 0 then
7. \( \beta \) \( \leftarrow \) a small random perturbation
8. \( x \leftarrow x_{nat} + \beta \)
end if
9. Store the transformation \( T_{aug} \) for the inverse augmentation:
10. \( x_{aug}, x_{nat, aug}, T_{aug} \leftarrow data_{aug}(x, x_{adv}) \)
11. \( x^* \leftarrow A(f_\theta, x_{nat, aug}, y, x_{aug}, \epsilon) \)
12. \( \theta \leftarrow \theta - \nabla f_\theta \phi_L(f_\theta, x^*, y) \)
13. \( x \leftarrow inverse_{aug}(x, x^*, T_{aug}) \)
end for
end for

5. Evaluation

In this section, we integrate ATTA with two popular adversarial training methods: Madry’s Adversarial Training (MAT) [18] and TRADES [34]. By evaluating the training time and robustness, we show that ATTA can provide a better trade-off than other adversarial training methods. To understand the contribution of each component to robustness, we conduct an ablation study.

5.1 Setup

Following the literature [18, 32, 34], we use both MNIST [14] and CIFAR10 dataset [12] to evaluate ATTA.

For the MNIST dataset, the model has four convolutional layers followed by three full-connected layers which is same architecture as used in [18, 34]. The adversarial perturbation is bounded by \( l_\infty \) ball with size \( \epsilon = 0.3 \).

For the CIFAR10 dataset, we use the wide residual network WRN-34-10 [31] which is same as [18, 34]. The perturbation is bounded by \( l_\infty \) ball with size \( \epsilon = 0.031 \).

5.2 Efficiency and effectiveness of ATTA

In this part, we evaluate the training efficiency and the robustness of ATTA, comparing it to state-of-the-art adversarial training algorithms. To better verify effectiveness of ATTA, we also evaluate ATTA under various attacks.

5.2.1 Training efficiency

We select four state-of-the-art adversarial training methods as baselines: MAT [18], TRADES [34], YOPO [32] and Free [23]. For MNIST, the model trained with ATTA can achieve a comparable robustness with up to 12.2 times training efficiency and, for CIFAR10, our method can achieve a comparable robustness with up to 14.1 times training efficiency. Compared to MAT trained with PGD, our method improve the accuracy up to 7.2% with 3.7 times training efficiency.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Attack</th>
<th>Natural</th>
<th>PGD-40</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAT</td>
<td>PGD-1</td>
<td>99.52%</td>
<td>15.82%</td>
<td>226</td>
</tr>
<tr>
<td></td>
<td>PGD-40</td>
<td>99.37%</td>
<td>96.21%</td>
<td>3933</td>
</tr>
<tr>
<td></td>
<td>YOPO-5-10</td>
<td>98.89%</td>
<td>96.54%</td>
<td>6544</td>
</tr>
<tr>
<td></td>
<td>ATTA-1</td>
<td>98.03%</td>
<td>96.10%</td>
<td>460</td>
</tr>
<tr>
<td></td>
<td>ATTA-40</td>
<td>98.21%</td>
<td>96.03%</td>
<td>4660</td>
</tr>
</tbody>
</table>

Table 3: The result of different attacks on MNIST dataset.

**MNIST.** The experiment results of MNIST are summarised in Table 3. For MAT, to achieve comparable robustness, ATTA is about 12.2 times faster than the traditional PGD training method. Even with one attack iteration, model trained with ATTA-1 achieves 96.31% adversarial accuracy within 297 seconds. For TRADES, we get a similar result. With one attack iteration, ATTA-1 achieves 96.1% adversarial accuracy within 460 seconds, which is about 13 times faster than TRADES(PGD-40). Compared to another fast adversarial training method YOPO, ATTA-1 is about 2.7 times faster and achieves higher robustness (96.1% versus YOPO’s 93.69%).

**CIFAR10.** We summarise the experiment results of CIFAR10 in Table 4. For MAT, compared to PGD-10, ATTA-1 achieves 3.89% higher adversarial accuracy with about 15.1 times training efficiency, and ATTA-10 improves adversarial accuracy by 7.26% with 2.9 times training efficiency when the model is trained with 10 attack iterations.
### Table 4: The result of different attacks on CIFAR10 dataset.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Attack</th>
<th>Natural</th>
<th>PGD-20</th>
<th>Time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAT</td>
<td>PGD-1</td>
<td>93.18%</td>
<td>22.3%</td>
<td>435</td>
</tr>
<tr>
<td></td>
<td>PGD-3</td>
<td>89.95%</td>
<td>41.38%</td>
<td>785</td>
</tr>
<tr>
<td></td>
<td>PGD-10</td>
<td><strong>87.49%</strong></td>
<td><strong>47.07%</strong></td>
<td><strong>2027</strong></td>
</tr>
<tr>
<td>YOPO-5-3</td>
<td>Free(m = 8)</td>
<td>85.54%</td>
<td>47.68%</td>
<td>640</td>
</tr>
<tr>
<td></td>
<td>YOPO-3-4</td>
<td>86.43%</td>
<td>48.24%</td>
<td>335</td>
</tr>
<tr>
<td></td>
<td>ATTA-1</td>
<td><strong>85.71%</strong></td>
<td><strong>50.96%</strong></td>
<td><strong>134</strong></td>
</tr>
<tr>
<td></td>
<td>ATTA-3</td>
<td>85.44%</td>
<td>52.56%</td>
<td>267</td>
</tr>
<tr>
<td></td>
<td>ATTA-10</td>
<td>83.80%</td>
<td>54.33%</td>
<td>690</td>
</tr>
<tr>
<td>TRADES</td>
<td>PGD-1</td>
<td>93.58%</td>
<td>35.52%</td>
<td>592</td>
</tr>
<tr>
<td></td>
<td>PGD-3</td>
<td>88.52%</td>
<td>54.20%</td>
<td>861</td>
</tr>
<tr>
<td></td>
<td>PGD-10</td>
<td><strong>84.13%</strong></td>
<td><strong>56.6%</strong></td>
<td><strong>2028</strong></td>
</tr>
<tr>
<td>YOPO-3-4</td>
<td>Free(m = 4)</td>
<td>86.43%</td>
<td>48.24%</td>
<td>335</td>
</tr>
<tr>
<td></td>
<td>ATTA-1</td>
<td>85.04%</td>
<td>54.50%</td>
<td>199</td>
</tr>
<tr>
<td></td>
<td>ATTA-3</td>
<td><strong>84.23%</strong></td>
<td><strong>56.36%</strong></td>
<td><strong>320</strong></td>
</tr>
<tr>
<td></td>
<td>ATTA-10</td>
<td>83.67%</td>
<td>57.34%</td>
<td>752</td>
</tr>
</tbody>
</table>

For TRADES, ATTA-3 achieves comparable adversarial accuracy to PGD-10 with 5.3 times faster training efficiency. By comparing the experiment results with YOPO and Free, for MAT, our method is 3.78 (1.5) times faster than Free (YOPO) with 3.28% (2.72%) better adversarial accuracy.

To better understand the performance of ATTA, we present the trade-off between training time and robustness of different methods in Figure 6. We find that our method (indicated by the solid markers in the left-top corner) gives a better trade-off on efficiency and effectiveness on adversarial training.

#### 5.2.2 Defense under other attacks

To verify the robustness of our method, we evaluate models in the previous section with other attacks: PGD-100 [13], FGSM [8], CW-20 [3].

As shown in Table 5, models trained with ATTA are still robust to other attacks. Compared to baselines, our method still achieves a comparable or better robustness under other attacks. We find that, although ATTA-40 has a similar robustness to PGD-40 under PGD-20 attack, with a stronger attack (e.g. PGD-100), ATTA-40 shows a better robustness.

### Table 5: The robustness comparison between ATTA and PGD under other attacks. The first ‘M’ and ‘T’ stand for MAT and TRADES, respectively.

<table>
<thead>
<tr>
<th>Defense</th>
<th>PGD100</th>
<th>FGSM</th>
<th>CW20</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-PGD-40</td>
<td>94.69%</td>
<td>97.37%</td>
<td>99.06%</td>
</tr>
<tr>
<td>M-ATTA-40</td>
<td>96.85%</td>
<td>98.55%</td>
<td>98.02%</td>
</tr>
<tr>
<td>CIFAR10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-PGD-10</td>
<td>46.77%</td>
<td>63.5%</td>
<td>56.7%</td>
</tr>
<tr>
<td>M-ATTA-10</td>
<td>52.6%</td>
<td>63.49%</td>
<td>75.37%</td>
</tr>
<tr>
<td>T-PGD-10</td>
<td>55.36%</td>
<td>63.02%</td>
<td>79.4%</td>
</tr>
<tr>
<td>T-ATTA-10</td>
<td>56.39%</td>
<td>64.1%</td>
<td>82.01%</td>
</tr>
</tbody>
</table>

### 5.2.3 ATTA with ImageNet

Table 6 shows the results for training an ImageNet Classifier [6] with ATTA for \(l_\infty(\epsilon = 2/255)\) attack with the loss used in MAT. We also show previously reported results of Free \((m = 4)\) from [23]. Shafahi et al. [23] report requiring about 50 hours with four P100s to adversarially train their model. ATTA required 97 hours with one 2080Ti.

### Table 6: Evaluation of ATTA on ImageNet

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>PGD10</th>
<th>PGD50</th>
<th>PGD100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free(m = 4)</td>
<td>64.44%</td>
<td>43.52%</td>
<td>43.39%</td>
<td>43.40%</td>
</tr>
<tr>
<td>ATTA-2</td>
<td>60.70%</td>
<td>44.57%</td>
<td>43.57%</td>
<td>43.51%</td>
</tr>
</tbody>
</table>
5.3. Ablation study

To study the contribution of each component to robustness, we do an ablation study on inverse data augmentation and different attack loss functions.

**Inverse data augmentation.** To study the robustness gain of inverse data augmentation (i.d.a.), we use ATTA-1 to adversarially train models by reusing the adversarial perturbation directly. As shown in Table 7, for both MAT (ATTA-1) and TRADES (ATTA-1), models trained with inverse data augmentation achieve about 6% higher accuracy, which means that inverse data augmentation does help improve the transferability between training epochs. As discussed in 4.1, another alternative is to remove data augmentation. However, Table 7 shows that removal of data augmentation hurts both natural accuracy and robustness.

<table>
<thead>
<tr>
<th>Attack</th>
<th>Natural</th>
<th>PGD-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAT(w/o d.a., w/o i.d.a.)</td>
<td>82.53%</td>
<td>41.64%</td>
</tr>
<tr>
<td>MAT(w/ d.a., w/o i.d.a.)</td>
<td>91.65%</td>
<td>42.55%</td>
</tr>
<tr>
<td>MAT(w/ d.a., w/ i.d.a.)</td>
<td>85.71%</td>
<td>50.96%</td>
</tr>
<tr>
<td>TRADES(w/o d.a., w/o i.d.a.)</td>
<td>81.87%</td>
<td>41.65%</td>
</tr>
<tr>
<td>TRADES(w/ d.a., w/o i.d.a.)</td>
<td>90.46%</td>
<td>48.38%</td>
</tr>
<tr>
<td>TRADES(w/ d.a., w/ i.d.a.)</td>
<td>85.04%</td>
<td>54.50%</td>
</tr>
</tbody>
</table>

Table 7: The accuracy under PGD attack of models trained by ATTA-1 with or without d.a.(data augmentation) and i.d.a. (inverse data augmentation).

**Attack loss.** Zhang et al. [34] show that, for TRADES, using Equation 2 leads to a better robustness. However, in this attack loss, we noticed that both inputs to the loss function $\mathcal{L}$ are related to the model $f$. Since the model $f$ is updated every epoch, compared to Equation 3 whose $y$ is fixed, the instability of $f(x^*)$ and $f(x)$ may have a larger influence on transferability. To analyze the performance difference between these two attack losses in ATTA, we train two TRADES/ATTA-1 models with different attack losses. In Table 8, we find that Equation 3 leads to 2.48% higher accuracy against PGD-20 attack. This result suggests that the higher stability of Equation 3 helps ATTA increase transferability between training epochs.

### 6. Related work

Several works [2, 4, 10, 15, 18, 23, 24, 28, 29, 32] focus on enhancing either the efficiency or effectiveness of the adversarial training that was first proposed in [13]. We highlight most relevant work here. YOPO [32], TRADES [34] improves the robustness of an adversarially trained model by adding a robustness regularizer in the loss function.

### 7. Conclusion

ATTA is a new method for iterative attack based adversarial training that significantly speeds up training time and improves model robustness. The key insight behind ATTA is high transferability between models from neighboring epochs, which is firstly revealed in this paper. Based on this property, the attack strength in ATTA can be accumulated across epochs by repeatedly reusing adversarial perturbations from the previous epoch. This allows ATTA to generate the same (or even stronger) adversarial examples with fewer attack iterations. We evaluate ATTA and compare it with state-of-the-art adversarial training methods. It greatly shortens the training time with comparable or even better model robustness. More importantly, ATTA is a generic method and can be applied to enhance the performance of other iterative attack based adversarial training methods.

### 8. Acknowledgments

We thank Fan Lai, Rui Liu, Jimmy You, Peifeng Yu, and Yiwen Zhang for helpful discussions. We thank Ryan Feng and Renuka Kumar for feedback on paper drafts. This work is supported by NSF Grant No.1646392.
References


In International Conference on Machine Learning (ICML), 2018.


