Adversarial Robustness Training: Technical Report

Abstract

This report presents our approach to training adversarially robust deep neural networks for image classification. We implemented a conservative adversarial training strategy that balances clean accuracy with robustness against Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) attacks. Our final model achieved 64.87% clean accuracy, 39.93% FGSM accuracy, and 0.33% PGD accuracy, demonstrating the inherent trade-off between robustness and clean performance.

1. Introduction

Adversarial examples pose a significant threat to deep learning systems, where imperceptible perturbations can cause misclassification. This assignment focuses on adversarial training - a defense mechanism that improves model robustness by including adversarial examples during training. The challenge lies in maintaining clean accuracy while gaining robustness against attacks.

2. Methodology

2.1 Model Architecture

We selected ResNet-34 as our base architecture for several reasons:

- **Proven Performance**: Strong baseline performance on image classification tasks
- **Computational Efficiency**: Balanced between capacity and training time
- **Transfer Learning**: Leveraged ImageNet pre-trained weights for better initialization

The model was modified for 10-class classification with Xavier uniform initialization for the final layer to ensure stable training.

2.2 Adversarial Attack Implementation

2.2.1 Fast Gradient Sign Method (FGSM)

```
def fgsm_attack(model, loss_fn, images, labels, epsilon):
    images = images.clone().detach().requires_grad_(True)
    outputs = model(images)
    loss = loss_fn(outputs, labels)
    model.zero_grad()
```

```
    loss.backward()
    grad_sign = images.grad.data.sign()
    adv_images = images + epsilon * grad_sign
    return torch.clamp(adv_images, 0, 1).detach()
```

FGSM generates adversarial examples by taking a single step in the direction of the gradient sign, making it computationally efficient but less sophisticated than iterative methods.

2.2.2 Projected Gradient Descent (PGD)

```
def pgd_attack(model, loss_fn, images, labels, epsilon, alpha, iters):
    orig_images = images.clone().detach()
    delta = torch.zeros_like(images).uniform_(-epsilon, epsilon)
    delta = torch.clamp(delta, 0-images, 1-images)

    for _ in range(iters):
        # ... iterative gradient ascent with projection

    return (orig_images + delta).detach()
```

PGD represents a stronger attack through multiple gradient steps with projection back to the ε-ball, providing more challenging adversarial examples for training.

2.3 Conservative Adversarial Training Strategy

Our training approach emphasizes maintaining clean accuracy while gradually building robustness:

2.3.1 Curriculum Learning Schedule

- **Phase 1 (Epochs 1-20)**: Clean data only to establish strong baseline
- **Phase 2 (Epochs 21-50)**: Gradual introduction of weak adversarial examples
- **Phase 3 (Epochs 51-80)**: Increase adversarial strength progressively
- **Phase 4 (Epochs 81-100)**: Full strength with 75% clean data ratio

2.3.2 Mixed Training Strategy

Each training batch combines:

- **Clean Examples**: Always maintain majority (75% minimum)
- **FGSM Examples**: Moderate robustness training
- **PGD Examples**: Strong robustness training (introduced later)

This approach prevents overfitting to adversarial examples while building robustness incrementally.

2.4 Training Configuration

2.4.1 Hyperparameters

- **Learning Rate**: 0.01 (conservative for stability)
- **Optimizer**: SGD with momentum (0.9) and weight decay (1e-4)
- **Batch Size**: 128
- **Attack Parameters**: $\varepsilon = 8/255$, $\alpha = 2/255$, PGD iterations = 10
- **Gradient Clipping**: Max norm = 0.5

2.4.2 Data Augmentation

Conservative augmentation to maintain clean accuracy:

- Random horizontal flip (30% probability)
- Random crop with padding (32×32 with 2-pixel padding)
- Standard normalization

2.5 Evaluation Strategy

2.5.1 Multi-Attack Evaluation

Models were evaluated on:

1. **Clean Examples**: Original unperturbed test data
2. **FGSM Examples**: Single-step adversarial examples
3. **PGD Examples**: Multi-step adversarial examples

2.5.2 Combined Scoring

We used a weighted combination favoring clean accuracy:

- Combined Score = 0.7 × Clean Accuracy + 0.2 × FGSM Accuracy + 0.1 × PGD Accuracy

This weighting reflects the practical importance of clean performance while incentivizing robustness.

3. Results and Analysis

3.1 Final Performance

- **Clean Accuracy**: 64.87%
- **FGSM Accuracy**: 39.93%
- **PGD Accuracy**: 0.33%
- **Combined Score**: 0.7 × 0.6487 + 0.2 × 0.3993 + 0.1 × 0.0033 = 0.5344

3.2 Performance Analysis

3.2.1 Clean Accuracy (64.87%)

The clean accuracy demonstrates reasonable baseline performance considering the adversarial training constraints. The conservative approach successfully prevented catastrophic degradation of clean performance.

3.2.2 FGSM Robustness (39.93%)

The model shows moderate robustness against single-step attacks. This suggests the adversarial training successfully learned to handle gradient-based perturbations to some extent.

3.2.3 PGD Robustness (0.33%)

The extremely low PGD accuracy reveals the model's vulnerability to strong multi-step attacks. This highlights the difficulty of defending against sophisticated adversarial examples.

3.3 Robustness-Accuracy Trade-off

Our results exemplify the fundamental trade-off in adversarial training:

- **Conservative Strategy**: Maintained reasonable clean accuracy (64.87%)
- **Robustness Cost**: Limited defense against strong attacks (0.33% PGD accuracy)
- **Balanced Approach**: Moderate performance on weaker attacks (39.93% FGSM accuracy)

4. Challenges and Limitations

4.1 Technical Challenges

4.1.1 Training Instability

- Adversarial training often leads to unstable gradients
- Mitigated through gradient clipping and conservative learning rates
- Curriculum learning helped stabilize early training phases

4.1.2 Computational Cost

- Adversarial example generation significantly increases training time
- PGD attacks require multiple forward/backward passes
- Balanced computational efficiency with training effectiveness

4.2 Methodological Limitations

4.2.1 Attack Strength

- Limited to $\varepsilon = 8/255$ perturbations
- Real-world attacks may use different norms or adaptive strategies
- Evaluation limited to gradient-based attacks

4.2.2 Generalization

- Robustness may not transfer to unseen attack methods
- Clean accuracy sacrifice may not justify limited robustness gains
- Model may overfit to specific attack patterns

5. Comparison with State-of-the-Art

5.1 Baseline Comparisons

- **Standard Training**: Typically achieves 90%+ clean accuracy but 0% adversarial robustness
- **Basic Adversarial Training**: Usually 60-70% clean accuracy, 40-50% FGSM accuracy
- **Our Approach**: 64.87% clean, 39.93% FGSM - comparable to standard adversarial training

5.2 Advanced Methods

- **TRADES**: Often achieves better robustness-accuracy trade-offs
- **AWP**: Adversarial Weight Perturbation shows improved robustness
- **Certified Defenses**: Provide provable robustness guarantees

6. Future Improvements

6.1 Training Enhancements

1. **Advanced Loss Functions**: Implement TRADES or MART loss
2. **Regularization Techniques**: Add adversarial weight perturbation
3. **Ensemble Methods**: Combine multiple adversarially trained models
4. **Progressive Training**: More sophisticated curriculum learning

6.2 Architectural Improvements

1. **Wider Networks**: Use WideResNet architectures
2. **Attention Mechanisms**: Incorporate attention for robust features
3. **Normalization Layers**: Experiment with different normalization schemes
4. **Skip Connections**: Enhanced residual connections for robustness

6.3 Evaluation Extensions

1. **Adaptive Attacks**: Evaluate against adaptive attack methods
2. **Different Norms**: Test L_2 and $L\infty$ bounded attacks
3. **Semantic Attacks**: Evaluate against semantic perturbations
4. **Certified Robustness**: Implement provable defense mechanisms

7. Conclusion

This project successfully implemented a conservative adversarial training strategy that maintains reasonable clean accuracy while providing moderate robustness against weaker attacks. The results demonstrate the inherent challenges in adversarial training:

7.1 Key Achievements

- Implemented comprehensive adversarial training pipeline
- Maintained clean accuracy above 60% threshold

- Achieved moderate robustness against FGSM attacks
- Demonstrated systematic approach to curriculum learning

7.2 Lessons Learned

- Conservative approaches preserve clean accuracy at robustness cost
- Curriculum learning is crucial for stable adversarial training
- Strong attacks (PGD) remain extremely challenging to defend against
- Robustness-accuracy trade-off is fundamental and unavoidable

7.3 Impact and Applications

The developed model provides a foundation for robust classification in adversarial environments, with applications in:

- Security-critical systems requiring attack resistance
- Baseline for advanced robustness research
- Educational demonstration of adversarial training principles