Kapil Yadav (B22AI024)

Arsewad Bhagwan (B22AI010)

Development and Analysis of Adversarial Defense Mechanisms for Multi-Label Classification Models

1. Introduction

This report presents a comprehensive investigation into adversarial defense mechanisms for multi-label classification models trained on the **IAPRTC-12** dataset. Building upon the vulnerability assessment conducted in Task 1, we implemented and evaluated five distinct defense strategies to improve model robustness against adversarial attacks:

1. **Adversarial Training**: Incorporating adversarial examples into the training process
2. **Feature Squeezing**: Reducing input precision to eliminate adversarial perturbations
3. **Gaussian Data Augmentation**: Training with noise-augmented data to improve robustness
4. **Ensemble Methods**: Combining predictions from multiple models to mitigate attacks
5. **Input Transformation**: Preprocessing inputs to neutralize adversarial perturbations

The effectiveness of these defense mechanisms was evaluated on five classification algorithms:

- Linear SVM
- Logistic Regression
- Softmax Regression
- Decision Tree
- Weighted KNN

2. Methodology

2.1 Dataset and Base Models

We utilized the IAPRTC-12 dataset, containing:

- Training set: 17,665 samples × 2,048 features, with 291 possible labels
- Testing set: 1,962 samples × 2,048 features, with 291 possible labels

For computational efficiency, we used PCA dimensionality reduction to 200 components and evaluated defenses on a consistent subset of test samples.

2.2 Defense Implementation

2.2.1 Adversarial Training

To implement adversarial training:

1. We generated adversarial examples using FGSM with $\varepsilon = 0.1$ for training samples
2. We combined original and adversarial examples in the training set with a ratio of 50:50
3. We retrained each model on this augmented dataset
4. We saved checkpoints to enable resuming training if interrupted

This approach followed the principle that exposure to adversarial examples during training helps models learn to recognize and correctly classify them during testing.

2.2.2 Feature Squeezing

Feature squeezing reduces the precision of input features to eliminate adversarial perturbations:

1. We normalized features to the [0,1] range
2. We quantized values to specific bit depths (3, 5, and 7 bits)
3. We rescaled quantized values back to their original range

For example, with 3-bit squeezing, each feature is mapped to one of $2^3 = 8$ possible values, effectively removing small perturbations below the quantization threshold.

2.2.3 Gaussian Data Augmentation

This defense trains models on noise-augmented data to improve robustness:

1. We generated multiple noisy copies of each training example by adding Gaussian noise with $\sigma = 0.1$
2. We trained new models on the augmented dataset combining original and noisy examples
3. We used checkpointing to save trained models for efficient evaluation

This approach helps models learn to be less sensitive to small perturbations in the input space.

2.2.4 Ensemble Methods

Ensemble defense leverages multiple models' predictions to increase robustness:

1. For each target model, we created an ensemble of all other trained models
2. We implemented both hard voting (majority vote) and soft voting (probability averaging)
3. We applied the ensemble to both clean and adversarial examples

This defense exploits the fact that different model architectures often have uncorrelated vulnerabilities to adversarial examples.

2.2.5 Input Transformation

Input transformation preprocesses inputs to neutralize adversarial perturbations:

1. **Robust Scaling**: Normalizes features using statistics resistant to outliers
2. **PCA Transformation**: Projects data to principal components and reconstructs it

3. **Quantile Transformation**: Transforms features to follow a normal distribution

Each transformation aims to preserve the underlying signal while removing adversarial perturbations.

2.3 Resilient Evaluation Framework

To ensure robust and reliable evaluation, we implemented:

1. **Comprehensive checkpointing**: Saved intermediate states for all processing steps
2. **Error handling**: Implemented try-except blocks with appropriate fallbacks
3. **Incremental result saving**: Stored results after each evaluation step
4. **Progress tracking**: Maintained logs to avoid redundant computation

This framework allowed the extensive evaluation to proceed despite occasional interruptions or computational constraints.

3. Results and Analysis

3.1 Feature Squeezing

Feature squeezing showed varying effectiveness across different bit depths and models.

Table 1: Feature Squeezing Results (ε = 0.5)

Model	No Defense F1	3-bit F1	5-bit F1	7-bit F1	Best Improvement
Linear SVM	0.3718	0.4172	0.3901	0.3754	12.2% (3-bit)
Logistic Regression	0.1235	0.1285	0.1252	0.1239	4.0% (3-bit)
Softmax Regression	0.1235	0.1285	0.1252	0.1239	4.0% (3-bit)
Decision Tree	0.3726	0.3895	0.3794	0.3753	4.5% (3-bit)
Weighted KNN	0.4295	0.4527	0.4385	0.4321	5.4% (3-bit)

1. Lower bit depths (3-bit) consistently provided the most effective defense
2. All models showed improvement with optimal bit-depth selection
3. Linear SVM achieved the greatest relative improvement (12.2%)
4. As bit depth increased, performance approached the undefended baseline
5. Weighted KNN maintained the highest absolute F1 score (0.4527)

3.2 Adversarial Training

Adversarial training improved robustness for most models when tested against adversarial examples.

Table 2: Adversarial Training Results (ε = 0.5)

Model	Clean Data F1	Adversarial F1	With Adv. Training F1	Improvement
Linear SVM	0.3475	0.3718	0.4089	10.0%
Logistic Regression	0.1302	0.1235	0.1325	7.3%
Softmax Regression	0.1302	0.1235	0.1325	7.3%
Decision Tree	0.3342	0.3726	0.3921	5.2%
Weighted KNN	0.4647	0.4295	0.4532	5.5%

Key findings:

1. All models showed improved performance with adversarial training
2. Linear SVM achieved the largest relative improvement (10.0%)
3. Performance on clean data was generally maintained or slightly improved
4. The defense was effective across different model architectures

3.3 Gaussian Data Augmentation

Gaussian augmentation showed moderate improvements for most models.

Table 3: Gaussian Augmentation Results (ε = 0.5, σ = 0.1)

Model	Clean Data F1	Adversarial F1	With Gaussian Aug F1	Improvement
Linear SVM	0.3475	0.3718	0.3892	4.7%
Logistic Regression	0.1302	0.1235	0.1297	5.0%
Softmax Regression	0.1302	0.1235	0.1297	5.0%
Decision Tree	0.3342	0.3726	0.3854	3.4%
Weighted KNN	0.4647	0.4295	0.4437	3.3%

1. All models showed modest improvements with Gaussian augmentation
2. Improvements were generally smaller than those from adversarial training
3. Logistic and Softmax Regression showed the largest relative improvements
4. Weighted KNN maintained the highest absolute F1 score (0.4437)

3.4 Ensemble Methods

Ensemble defense showed substantial variations in effectiveness depending on which model was being defended.

Table 4: Ensemble Defense Results (ε = 0.5)

Attack Model	Base F1	Ensemble (Hard) F1	Ensemble (Soft) F1	Best Improvement
Linear SVM	0.3718	0.4124	0.4283	15.2% (Soft)
Logistic Regression	0.1235	0.4137	0.4296	247.9% (Soft)
Softmax Regression	0.1235	0.4137	0.4296	247.9% (Soft)
Decision Tree	0.3726	0.4128	0.4289	15.1% (Soft)
Weighted KNN	0.4295	0.3975	0.4211	-2.0% (Soft)

1. Soft voting consistently outperformed hard voting
2. Logistic and Softmax Regression showed dramatic improvements (247.9%)
3. Linear SVM and Decision Tree showed significant improvements (15.1-15.2%)
4. Weighted KNN showed a slight decrease in performance with ensemble defense
5. The ensemble method produced relatively consistent F1 scores (~0.43) for most models

3.5 Input Transformation

Input transformation techniques showed variable effectiveness across different models and methods.

Table 5: Input Transformation Results (ε = 0.5)

Model	No Defense F1	Robust Scaling F1	PCA F1	Quantile F1	Best Improvement
Linear SVM	0.3718	0.4209	0.3825	0.3973	13.2% (Scaling)

Logistic Regression	0.1235	0.1317	0.1235	0.1272	6.6% (Scaling)
Softmax Regression	0.1235	0.1317	0.1235	0.1272	6.6% (Scaling)
Decision Tree	0.3726	0.3942	0.3781	0.3854	5.8% (Scaling)
Weighted KNN	0.4295	0.4518	0.4342	0.4425	5.2% (Scaling)

1. Robust scaling was consistently the most effective transformation method
2. Linear SVM showed the largest relative improvement (13.2%)
3. PCA transformation provided the smallest benefits
4. All models showed some improvement with robust scaling
5. Quantile transformation was generally the second-most effective method

3.6 Comparative Analysis

Comparing all defense methods reveals which defense works best for each model architecture.

Table 6: Best Defense Method by Model (ε = 0.5)

Model	Best Defense Method	F1 Score	Improvement
Linear SVM	Ensemble (Soft)	0.4283	15.2%
Logistic Regression	Ensemble (Soft)	0.4296	247.9%
Softmax Regression	Ensemble (Soft)	0.4296	247.9%
Decision Tree	Ensemble (Soft)	0.4289	15.1%
Weighted KNN	Feature Squeezing (3-bit)	0.4527	5.4%

1. Ensemble methods with soft voting were the most effective for 4 out of 5 models
2. Weighted KNN uniquely performed best with feature squeezing
3. Different models showed dramatically different degrees of improvement
4. Logistic and Softmax Regression benefited most significantly from defense mechanisms
5. After applying defenses, the performance gap between different model types narrowed significantly

3.7 Defense Efficiency Analysis

We also analyzed the computational efficiency of different defense mechanisms.

Table 7: Computational Requirements of Defense Methods

Defense Method	Training Time	Inference Time	Memory Usage	Implementation Complexity
Adversarial Training	Very High	Low	Medium	Medium
Feature Squeezing	None	Low	Low	Low
Gaussian Augmentation	High	Low	Medium	Low
Ensemble Methods	None	Medium	High	Medium
Input Transformation	None	Medium	Medium	Medium

1. Feature squeezing offered the best efficiency-to-effectiveness ratio
2. Adversarial training required the highest computational resources during training
3. Ensemble methods had the highest memory requirements during inference
4. Some defenses (Feature Squeezing, Input Transformation) required no additional training time

4. Challenges and Limitations

Several challenges were encountered during this investigation:

4.1 Technical Challenges

1. **Computational Complexity**: Adversarial training and ensemble methods required significant computational resources, necessitating batch processing and checkpointing.
2. **Memory Management**: Working with high-dimensional data and multiple models simultaneously required careful memory handling to avoid out-of-memory errors.
3. **Numerical Stability**: Some transformations introduced numerical stability issues, particularly for models sensitive to feature scaling.
4. **Implementation Robustness**: Developing a system resistant to interruptions required careful design of checkpoint and recovery mechanisms.

4.2 Methodological Challenges

1. **Parameter Selection**: Finding optimal parameters (e.g., bit depth, noise level, ensemble weights) required extensive experimentation.
2. **Multi-label Complexity**: The multi-label nature of the problem (291 possible labels) made evaluation and optimization more complex than for single-label classification.
3. **Model-Defense Interactions**: Different defenses performed differently across model types, requiring model-specific optimization.
4. **Evaluation Metrics**: Determining which metrics best represent practical robustness was challenging in a multi-label setting.

4.3 Limitations

1. **Attack Diversity**: We focused primarily on FGSM attacks; other attack methods might yield different defense effectiveness patterns.
2. **Dataset Specificity**: Findings may not generalize to all domains or larger feature spaces.
3. **Computational Constraints**: More extensive hyperparameter tuning could potentially yield better results.
4. **Defense Combinations**: We evaluated each defense independently; combinations might provide synergistic benefits.

5. Conclusions and Recommendations

5.1 Key Findings

1. **Defense Effectiveness is Model-Dependent**: Different model architectures benefit from different defense strategies, with ensemble methods excelling for most models except Weighted KNN.
2. **Ensemble Power**: Soft-voting ensemble defense produced the most dramatic improvements, particularly for models with lower baseline robustness.
3. **Simplicity-Effectiveness Trade-off**: Simple methods like feature squeezing provided meaningful improvements with minimal computational overhead.
4. **Transformation Effectiveness**: Input transformations, particularly robust scaling, consistently improved robustness across all models.
5. **Model Architecture Impact**: Linear models (Logistic/Softmax Regression) that were initially most vulnerable benefited most dramatically from defensive measures.

5.2 Practical Recommendations

Based on our findings, we recommend the following strategies for deploying adversarially robust multi-label classification systems:

1. **Model-Specific Defense Selection**:
    - For Linear SVM and regression-based models: Prioritize ensemble methods with soft voting
    - For Decision Tree: Use ensemble methods or adversarial training
    - For Weighted KNN: Implement feature squeezing or input transformations
2. **Resource-Constrained Environments**:
    - When computational resources are limited: Implement feature squeezing (3-bit)
    - When memory is constrained: Avoid ensemble methods in favor of input transformations

    - When deployment speed is critical: Use pre-computed transformations
3. **Defense Layering**:
    - Implement multiple complementary defenses in sequence
    - Consider pipeline: input transformation → feature squeezing → model prediction
4. **Continuous Monitoring**:
    - Regularly evaluate model performance against new attack methods
    - Be prepared to adapt defense strategies as adversarial techniques evolve

5.3 Future Research Directions

Based on our findings, several promising areas for further research include:

1. **Defense Combinations**: Systematically evaluate combinations of multiple defense mechanisms (e.g., feature squeezing + ensemble methods).
2. **Adaptive Defenses**: Develop systems that dynamically select the most appropriate defense based on input characteristics.
3. **Theoretical Understanding**: Investigate why certain defenses work better with specific model architectures.
4. **Efficiency Optimization**: Develop more computationally efficient versions of the most effective defenses.
5. **Transfer Learning for Defense**: Explore whether defensive knowledge can transfer between models and domains.

In conclusion, this study demonstrates that appropriate selection and implementation of defense mechanisms can significantly improve the robustness of multi-label classification models against adversarial attacks. The effectiveness of these defenses varies substantially based on model architecture, highlighting the importance of tailored defense strategies rather than one-size-fits-all solutions.

5. References

- Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). "Explaining and harnessing adversarial examples".
- Xu, W., Evans, D., & Qi, Y. (2017). "Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks."
- Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., & Swami, A. (2017). "Practical black-box attacks against machine learning."
- Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., & McDaniel, P. (2017). "Ensemble adversarial training: Attacks and defenses."
- Guo, C., Rana, M., Cisse, M., & Van Der Maaten, L. (2018). "Countering adversarial images using input transformations."