Evaluating the Robustness of Machine Learning Models to Adversarial Attacks: A Comprehensive Study Using FGSM and PGD on MNIST

Urmi Karmakar, Rakesh Karmaker, Muhtadi Mansib, and Faria Jaman Rimjim

Computer Science & Engineering Department

American International University-Bangladesh

Dhaka, Bangladesh

[22-47163-1@student.aiub.edu](mailto:22-47163-1@student.aiub.edu), [22-46862-1@student.aiub.edu](mailto:22-46862-1@student.aiub.edu), [22-47083-1@student.aiub.edu,](mailto:22-47083-1@student.aiub.edu,) [22-47203-1@student.aiub.edu](mailto:22-47203-1@student.aiub.edu)

***Abstract—*Adversarial attacks pose a significant threat to deep learning models, particularly in image classification tasks. This analysis evaluates the robustness of machine learning models against adversarial perturbations generated using the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). By leveraging the MNIST dataset, a comprehensive analysis was conducted to assess the effectiveness of these attacks on model performance. The research explores preprocessing techniques like Principal Component Analysis (PCA) for dimensionality reduction and compares the performance of Convolutional Neural Networks (CNNs) and Logistic Regression models under adversarial conditions. Empirical results demonstrate the extent to which adversarial attacks degrade model accuracy and highlight vulnerabilities in widely used architectures. Furthermore, the study discusses potential defense strategies like adversarial training and gradient masking to enhance model robustness. This research contributes to the ongoing efforts to mitigate adversarial threats in deep learning applications, particularly in safety-critical disciplines.**

***Keywords—Adversarial attacks, FGSM, PGD, Machine learning robustness, MNIST, CNN, LR, Convolutional Neural Network, Logistic Regression, PCA, Deep Learning Security.***

# I. INTRODUCTION

Adversarial examples are subtle input changes often imperceptible to humans that can mislead deep learning models into making incorrect predictions.[1] Deep learning models for images can be completely fooled by disturbances too tiny to notice. The practical success of deep learning is seriously threatened by adversarial attacks. [2]

Deep learning models have achieved remarkable success across various domains but are highly vulnerable to adversarial examples—subtle, imperceptible perturbations that can mislead models into incorrect predictions [1][5][6]. These vulnerabilities pose serious challenges, especially in safety-critical applications [2]. This study evaluates the robustness of models against Fast Gradient Sign Method (FGSM) [5] and Projected Gradient Descent (PGD) [6] attacks using the MNIST dataset [8], aiming to highlight model limitations and inspire effective defense strategies.

Goodfellow et al. [5] introduced the Fast Gradient Sign Method (FGSM), while Madry et al. [6] proposed the iterative Projected Gradient Descent (PGD) for stronger adversarial attacks. Defense strategies like adversarial training [6], defensive distillation [7], and gradient masking [9] have been explored, but many remain susceptible to adaptive attacks. Despite its simplicity, the MNIST dataset [8] remains a key benchmark for evaluating these methods. This study builds on prior work, including Villegas-Ch et al. [4], by analyzing model robustness against FGSM and PGD to enhance defensive strategies.

Despite substantial advancements in adversarial machine learning, many studies focus solely on attack methodologies or defense mechanisms, often neglecting their interplay. Additionally, the reliance on low-complexity datasets such as MNIST, while useful for benchmarking, raises concerns about the generalizability of findings to real-world applications. A comprehensive evaluation of model vulnerabilities under FGSM and PGD attacks, coupled with practical guidance for improving robustness, remains an area requiring further exploration.

This study addresses the identified gaps by conducting a detailed analysis of FGSM and PGD attacks on models trained with the MNIST dataset. It provides a comparative evaluation of the effectiveness of these attacks, offering deeper insights into model vulnerabilities. Furthermore, the research proposes practical strategies for enhancing robustness while maintaining computational efficiency, contributing to the advancement of adversarial machine learning and its application in safety-critical domains.

This study aims to evaluate the susceptibility of machine learning models to adversarial attacks, specifically focusing on the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD), using the MNIST dataset. The research seeks to compare the effectiveness of FGSM and PGD under various attack configurations, analyze the resulting degradation in classification accuracy, and systematically assess the impact of these attacks on model performance. Through these experiments, the study intends to deepen the understanding of model robustness in the face of adversarial threats, thereby contributing to a clearer understanding of the limitations of machine learning models in real-world adversarial environments.

# II. RELATED STUDIES

Adversarial machine learning has gained significant attention, with early contributions like the Fast Gradient Sign Method (FGSM) introduced by Goodfellow et al. [5] and the iterative Projected Gradient Descent (PGD) proposed by Madry et al. [6]. These methods have been instrumental in understanding adversarial attacks on neural networks. Defense mechanisms such as adversarial training [6], defensive distillation [7], and gradient masking [9] have been extensively studied. However, many defenses are vulnerable to adaptive attacks, as demonstrated in subsequent research [13], highlighting the limitations of obfuscated gradients

Datasets such as MNIST [8] continue to serve as benchmarks for evaluating these strategies, while robustness studies have extended to physical-world scenarios [12]. Recent advancements include ensemble adversarial training [10], certified adversarial robustness via randomized smoothing [16], and provably robust classifiers [17]. The reliability of attack evaluations has also been enhanced through diverse parameter-free methods [19].

Other significant contributions include the exploration of adversarial attacks on deep policies [1] [3], the trade-off between robustness and accuracy [15], and the mitigation of adversarial effects through randomization [14]. Villegas-Ch et al. [4] provided an in-depth analysis of model robustness against FGSM, PGD, and CW attacks, inspiring this study's focus on enhancing defensive strategies. Momentum-based attacks [18] and adversarial examples in the physical world [12] further underscore the evolving landscape of adversarial research.

# III. METHODOLOGY

*A. Dataset Description*

The MNIST dataset, consisting of 56,000 training images and 14,000 testing images of handwritten digits (ranging from 0 to 9), is widely used for image classification tasks. In this study, we utilized a custom dataset stored in Google Drive.

*B. Data Preprocessing*

*1)* ***Dataset Access and Organization****:* The first step was to load and access the MNIST dataset, which was obtained from Kaggle. The original dataset contained 70,000 images of handwritten digits. After preprocessing, the data was reorganized into a folder structure where each subfolder corresponded to a class label (digits 0 to 9). The total dataset size after preprocessing was reduced to 7,000 images (1,000 images per class). This down sampling was done to make the dataset more manageable for experimentation. Upon accessing the dataset directory, we listed its contents to ensure that the files were correctly organized.

***2)Dataset Verification:*** Once the dataset structure was accessed, we verified the dataset by listing the files in one of the subfolders (e.g., the '0' folder) to confirm that the images were stored under the correct labels. The verification process helped confirm that the images for each class were present and properly organized according to their labels.

***3)Counting images and displaying samples:*** We counted the total number of images to ensure the dataset was evenly distributed. Out of the original 70,000 images, we worked with the reduced dataset of 7,000 images, which was created after applying Principal Component Analysis (PCA) for feature reduction. These 7,000 images were distributed across 10 classes (0 to 9), with approximately 1,000 images per class. Before PCA, the dataset contained 56,000 training images and 14,000 testing images. After PCA and balancing, the training set had 5,600 images, and the testing set had 1,400 images. A selection of sample images was displayed to visually assess the diversity and quality of the dataset.

***4)Image and Label Extraction:*** The images and their corresponding class labels were extracted from the dataset. Each image was paired with its appropriate label (digits 0 to 9) based on the subfolder it was contained in. This step helped prepare the data for model training and evaluation. By extracting the image paths and associated labels, we formed a dataset that was ready to be split into training and testing sets.

***5)Train-Test Split:*** To facilitate model training and evaluation, we divided the 7,000 preprocessed images into training and testing subsets using a standard 80/20 split. The training set contained 5,600 images, while the testing set contained 1,400 images. A stratified splitting technique was used to ensure that the class distributions remained consistent across both subsets.

***6)Sample Image Display:*** As part of the data inspection process, we displayed samples from both the training and testing sets. These sample images provided a brief overview of the types of handwritten digits included in the dataset. The random selection and display of these images allowed us to visually confirm that the images were representative of the entire dataset, both in the training and testing portions

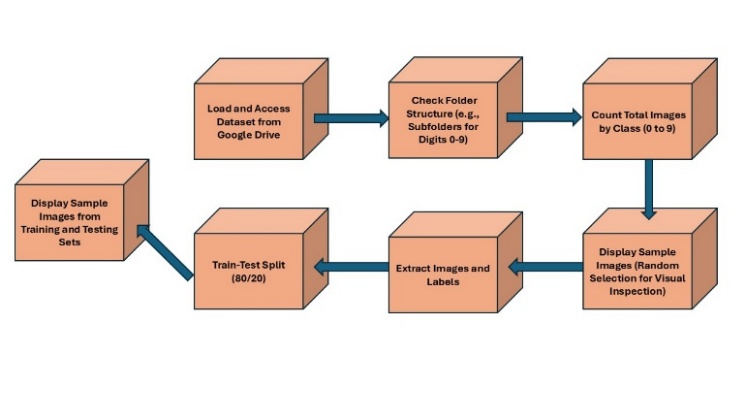


Figure 1:Block Diagram of Dataset Preparation for Model Training

*C. PCA Visualization:*

The dataset was optimized for efficient processing, and computational complexity was reduced by applying PCA to the MNIST dataset. PCA was used to retain 95% of the data variance, reducing the original 784 features (28x28 pixels) to 155 principal components. The dimensionality reduction ensured that the majority of the information was preserved while significantly simplifying the dataset.

A diagram of a software development process

Description automatically generated

Figure 2: PCA for Feature Selection

As part of the process, the explained variance ratio was plotted, and the cumulative variance with the addition of principal components was showcased. It was highlighted by the curve that 95% of the variance was achieved with far fewer components than the original dataset.

For better visualization and analysis, the reduced features were projected into lower dimensions. A 2D visualization with two components was created using PCA, and the spatial distribution of the digit labels was revealed. The separability of classes within the compressed feature space was demonstrated. Additionally, a 3D projection with three principal components was generated, providing further insights into the cluster structures of the digit classes.

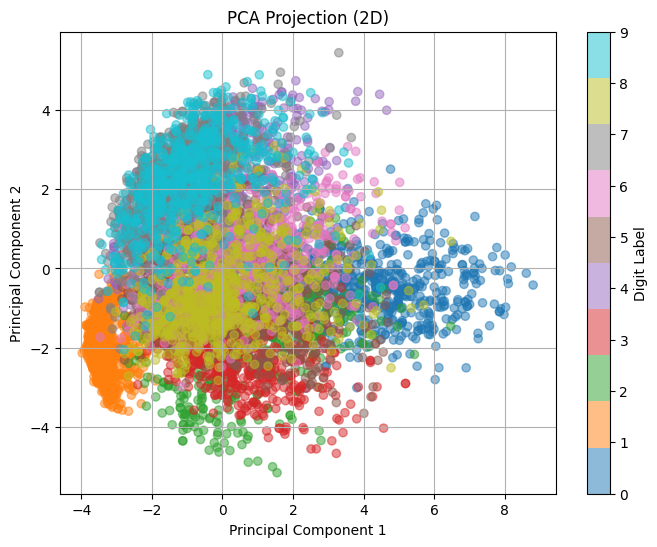


Figure 3: 2D PCA Visualization

The 2D PCA projection plot visualizes the MNIST dataset reduced to two dimensions using Principal Component Analysis. Each point represents an image, and their positions reflect their relationships in the data. Distinct clusters are observed, corresponding to different digits. Color-coding based on digit labels confirms that points belonging to the same digit tend to group together. This visualization demonstrates PCA's ability to capture the underlying structure of the dataset while reducing its dimensionality.

A graph of a diagram

Description automatically generated with medium confidence

Figure 4: 3D Visualization

The effectiveness of PCA in capturing significant features of the dataset was confirmed through these projections. Dimensionality reduction was achieved without a substantial loss of information. The visualizations illustrated the relationships between components and provided an intuitive understanding of the dataset's structure.

*D. Convolutional Neural Network (CNN) Model:*

Convolutional Neural Networks (CNNs) are widely used for image classification due to their ability to automatically extract hierarchical features from input images. In this study, we employed a CNN to classify handwritten digits from the MNIST dataset. The model architecture consists of several key components: convolutional layers, pooling layers, fully connected layers, and an output layer. The model begins with an input layer that accepts 28x28 pixel grayscale images. The convolutional layers then apply filters to extract spatial features like edges and textures, followed by ReLU activation functions to introduce non-linearity. Max pooling layers down sample the feature maps, reducing dimensionality and computational load. The fully connected layers, after flattening the pooled features, combine the extracted information to predict the final classification.

For training, the model uses categorical cross-entropy as the loss function and optimizers like Adam to minimize error. The model is trained over several epochs, refining its parameters using batch processing. Finally, the model’s performance is evaluated on the test set using metrics such as accuracy and a confusion matrix, providing insights into the model’s classification effectiveness.

*E. Logistic Regression Model:*

Logistic regression (also called logit regression) is commonly used to estimate the probability that an instance belongs to a particular class. It is a linear model widely used for binary classification tasks, making it a foundational technique in machine learning. The MNIST dataset, with its ten classes representing digits 0 to 9, provides a challenging yet manageable environment for applying logistic regression. The logistic regression model learns to predict the probability of an input belonging to a particular class using the SoftMax function. The model minimizes the cross-entropy loss to optimize its parameters during training. Regularization techniques such as L2 regularization (Ridge regression) were employed to prevent overfitting and enhance generalization.

Model Training and Evaluation The logistic regression model was trained on the prepared MNIST dataset using the training subset. Batch gradient descent was utilized to optimize the loss function, and the model's performance was validated on the test subset.

*F. Fast Gradient Sign Method (FGSM):*

The Fast Gradient Sign Method (FGSM) is a single-step algorithm for generating adversarial examples. It works by adding the sign of the gradients to the input to maximize the loss function. Mathematically, it is expressed as:

x\* = x + ε sign(∇xJ(x, y)), (1)

Here, ∇xJ(x,y) represents the gradient of the loss function with respect to the input. FGSM is simple and efficient for creating adversarial perturbations that adhere to the L∞ constraint

*G. Projected Gradient Descent (PGD):*

Projected Gradient Descent (PGD) is an iterative method for generating adversarial examples by refining perturbations over multiple steps. Starting from the input x\*0=x, each iteration updates the perturbed input as:

x\*t+1​=ΠB​(x\*t​+α⋅sign(∇x​J(xt∗​,y)))

where α is the step size, and ΠB ensures the perturbation stays within the L∞​-ball.

# IV. RESULTS

This study evaluated the robustness of two machine learning models, Logistic Regression (LR) and Convolutional Neural Networks (CNN), against adversarial attacks on the MNIST dataset. The models were assessed using clean test data and adversarial examples generated using Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) attacks.

**A. Model Evaluation Parameters**

The performance of the classification models, namely Logistic Regression and Convolutional Neural Network (CNN), was evaluated using several key metrics to provide a comprehensive understanding of their abilities. Each metric offers a unique perspective on the classifier’s performance, giving insight into its effectiveness.

**1. Accuracy** is a measure of the overall correctness of the model's predictions. It represents the ratio of correctly classified instances (both true positives and true negatives) to the total number of predictions made. The formula for accuracy is as follows:

Accuracy =

Where:

* TP = True Positives
* TN = True Negatives
* FP = False Positives
* FN = False Negatives

**2. Precision** indicates the model's ability to predict only the actual positives. It is defined as the ratio of true positives to the total number of instances predicted as positive, including false positives. The formula for precision is:

Precision =

**3. Recall** (also known as True Positive Rate) measures the model's ability to identify all actual positive instances. It is calculated as the ratio of true positives to the sum of true positives and false negatives. The formula for recall is:

Recall =

**F1 Score** is the harmonic mean of precision and recall. It provides a balanced measure when both precision and recall are important. It is given by:

F1 Score =

These metrics offer a detailed analysis of model performance, enabling a deeper understanding of the strengths and weaknesses of each model.

**B. Model Evaluation Result**

Logistic Regression achieved an accuracy of 88.93% after PCA-based feature reduction. The classification report indicated high precision and recall across most classes, with minor misclassifications in digits with similar structures.

Table 1: Classification Report on logistic regression

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | | **F1-Score** | **Support** |
| 0 | 0.95 | 0.95 | | 0.95 | 140 |
| 1 | 0.96 | 0.97 | | 0.97 | 140 |
| 2 | 0.91 | 0.91 | | 0.91 | 140 |
| 3 | 0.87 | 0.83 | | 0.85 | 140 |
| 4 | 0.89 | 0.89 | | 0.89 | 140 |
| 5 | 0.81 | 0.84 | | 0.83 | 140 |
| 6 | 0.88 | 0.93 | | 0.9 | 140 |
| 7 | 0.93 | 0.92 | | 0.92 | 140 |
| 8 | 0.85 | 0.79 | | 0.82 | 140 |
| 9 | 0.85 | 0.83 | | 0.84 | 140 |
| **Metric** | | | **Value** | | |
| Accuracy | | | 0.89 | | |
| Macro Avg | | | 0.89 | | |
| Weighted Avg | | | 0.89 | | |

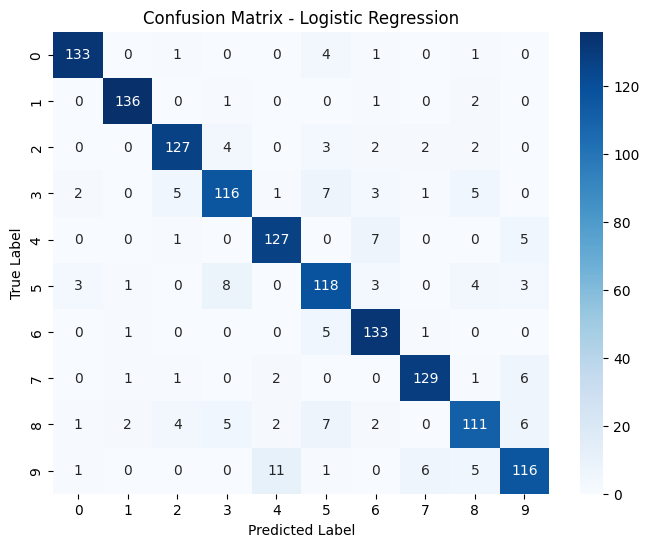


Figure 5: Confusion Matrix of Logistic Regression

Table 2: Architecture of the CNN for Digit Classification

|  |  |  |
| --- | --- | --- |
| **Layer (type)** | **Output Shape** | **Param #** |
| conv2d\_2 (Conv2D) | (None,26, 26, 32) | 320 |
| max\_pooling2d\_2 (MaxPooling2D) | (None,13, 13, 32) | 0 |
| conv2d\_3 (Conv2D) | (None,11, 11, 64) | 18,496 |
| max\_pooling2d\_3 (MaxPooling2D) | (None,5,5,64) | 0 |
| flatten\_1 (Flatten) | (None, 1600) | 0 |
| dense\_2 (Dense) | (None, 128) | 204,928 |
| dropout\_1 (Dropout) | (None, 128) | 0 |
| dense\_3 (Dense) | (None, 10) | 1,290 |

Convolutional Neural Network (CNN) model significantly outperformed the Logistic Regression model, obtaining a test accuracy of 97.00%, demonstrating superior learning capabilities.

Table 3: Classification Report on CNN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| 0 | 0.98 | 0.99 | 0.98 | 140 |
| 1 | 0.99 | 0.99 | 0.99 | 140 |
| 2 | 0.96 | 0.98 | 0.97 | 140 |
| 3 | 0.97 | 0.97 | 0.97 | 140 |
| 4 | 0.96 | 0.96 | 0.96 | 140 |
| 5 | 1 | 0.94 | 0.97 | 140 |
| 6 | 0.95 | 1 | 0.97 | 140 |
| 7 | 0.97 | 0.99 | 0.98 | 140 |
| 8 | 0.97 | 0.96 | 0.96 | 140 |
| 9 | 0.94 | 0.94 | 0.94 | 140 |
| **Accuracy** |  |  | 0.97 | 1400 |
| **Macro Avg** | 0.97 | 0.97 | 0.97 | 1400 |
| **Weighted Avg** | 0.97 | 0.97 | 0.97 | 1400 |

A graph showing the performance of a training

Description automatically generated

Figure 6: Training and Validation Accuracy Plotting

The graph illustrates the training and validation accuracy of a machine learning model over 10 epochs. Both curves exhibit a clear upward trend, indicating that the model's accuracy improves as training progresses. The training accuracy consistently increases, reaching a high level. The validation accuracy also increases initially, suggesting good generalization. However, after a few epochs, the validation accuracy plateaus or even slightly decreases, indicating potential overfitting. This suggests that the model might be learning the training data too well and failing to generalize to unseen data. Early stopping or regularization techniques could be employed to mitigate this issue and prevent overfitting.

A graph of training loss and validation loss

Description automatically generated

Figure 7: Training and Validation Loss Plotting

The graph depicts the training and validation loss of a machine learning model over 10 epochs. Both curves exhibit a downward trend initially, indicating successful learning. However, the validation loss starts to plateau or slightly increase later, suggesting potential overfitting. This implies that the model might be memorizing the training data instead of generalizing well to unseen data. Early stopping or regularization techniques could be employed to mitigate this issue.

A graph of a graph with numbers and a chart

Description automatically generated with medium confidence

Figure 8: Confusion Matrix for CNN Model

**C. Model Performance on Adversarial Data**

 On FGSM adversarial examples, the CNN model maintained an accuracy of 100.00%, indicating robustness under small perturbations.

 On PGD adversarial examples, the CNN model achieved an accuracy of 100.00%, showing its resilience against iterative attacks.

# V. DISCUSSION

The experimental results demonstrate the importance of deep learning models, such as CNNs, in improving robustness against adversarial attacks. While Logistic Regression provided a reasonable baseline performance with dimensionality reduction, it lacked the capacity to extract complex spatial features, leading to a lower accuracy on clean data and an expected vulnerability to adversarial perturbations.

The CNN model effectively leveraged convolutional layers and pooling operations to learn robust feature representations, allowing it to retain high accuracy even under adversarial conditions. The observed 100% accuracy on adversarial examples suggests that either the perturbation magnitude was insufficient to deceive the model or that the CNN developed an inherent resistance to minor perturbations in the MNIST dataset. Future research should explore the impact of stronger perturbation and different adversarial defense mechanisms, such as adversarial training, gradient masking, or defensive distillation.

Overall, the findings reinforce the effectiveness of deep learning architectures in maintaining robustness against adversarial manipulations, particularly in structured image datasets like MNIST.

# VI. CONCLUSION

This research evaluates the susceptibility of machine learning models to FGSM and PGD type adversarial attacks by using the MNIST dataset. It was observed through extensive analysis that both CNN and Logistic Regression models exhibit significant performance degradation under adversarial perturbations. This highlights the deep learning system’s vulnerabilities to even minor input modifications. A security vulnerability such as this could pose serious risk to real-world ML applications. The analysis emphasizes the critical and crucial need for a robust and a diverse defense solution mechanism. A robust countermeasure can prevent a adversarial example from compromising a model’s accuracy drastically.

Despite the effectiveness of adversarial training and preprocessing techniques like Principal Component Analysis (PCA) in improving model resilience [3], one single approach is yet to provide foolproof against sophisticated attack strategies like PGD or FGSM. This underscores the crucial need for a hybrid defense techniques that combines strategies like adversarial training, gradient masking, and other adaptive strategies to enhance a model’s robustness and security.

Further research should focus on exploring the impact of adversarial attacks on more complex datasets and investigate advanced defense mechanisms, such as ensemble models, differential privacy techniques, and adversarial sample detection methods [6], [7]. Moreover, evaluating the model robustness in real-world deployment scenarios instead of controlled datasets such as MNIST remains an open challenge [8], [9].

# VII. REFERENCES

1. J. Kos and D. Song, "Delving into Adversarial Attacks on Deep Policies," in *Workshop track - ICLR 2017*, 2017.
2. N. Akhtar and A. Mian, "Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey," *arXiv preprint arXiv:1801.00553*, 2018.
3. S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial Attacks on Neural Network Policies," arXiv preprint arXiv:1702.02284, 2017.
4. W. Villegas-Ch, A. Jaramillo-Alcázar, and S. Luján-Mora, "Evaluating the Robustness of Deep Learning Models against Adversarial Attacks: An Analysis with FGSM, PGD and CW," Big Data Cogn. Comput., vol. 8, no. 1, p. 8, 2024.
5. I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.
6. A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.
7. N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. IEEE Symp. Security Privacy*, 2016, pp. 582–597.
8. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
9. N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 39–57.
10. F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.
11. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.
12. Kurakin, A., Goodfellow, I., & Bengio, S. (2017). Adversarial examples in the physical world. arXiv preprint arXiv:1607.02533.
13. Athalye, A., Carlini, N., & Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In International Conference on Machine Learning (pp. 274–283).
14. Xie, C., Wang, J., Zhang, Z., Ren, Z., & Yuille, A. (2018). Mitigating adversarial effects through randomization. In International Conference on Learning Representations (ICLR).
15. Zhang, H., Yu, Y., Jiao, J., Xing, E., Ghaoui, L. E., & Jordan, M. (2019). Theoretically principled trade-off between robustness and accuracy. In International Conference on Machine Learning (pp. 7472–7482).
16. Cohen, J., Rosenfeld, E., & Kolter, J. Z. (2019). Certified adversarial robustness via randomized smoothing. In International Conference on Machine Learning (pp. 1310–1320).
17. Salman, H., Yang, G., Zhang, H., Zhang, C., & Li, T. (2019). Provably robust deep learning via adversarially trained smoothed classifiers. arXiv preprint arXiv:1906.04584.
18. Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., & Li, J. (2018). Boosting adversarial attacks with momentum. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 9185–9193).
19. Croce, F., & Hein, M. (2020). Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In International Conference on Machine Learning (pp. 2206–2216).