ADVERSARIAL DEFENSE FOR MNIST: INVESTIGATING ADVERSARIAL TRAINING AND FGSM

[1]Kommineni Srivathsav, [2]Sai Manas Rao Pulakonti, [3]Kadali Narayana Anudeep, [4]Kommineni Srinivas, [5]Kommineni Sri Lakshmi Poojitha

[1,2,3,4,5]Students,
[1,2,3]Department of Computer Science Engineering, JNTUH-University College of Engineering, Hyderabad,
[4]Department of Computer Science Engineering, Chaitanya Bharathi Institute of Technology, Hyderabad,
[5]Department of Information Technology, G. Narayanamma Institute of Technology and Science, Hyderabad,

Abstract: **This research looks at strategies for defending machine learning models from adversarial assaults, which are deliberate attempts to misclassify input fed to machine learning models in order to trick them. Machine learning systems' dependability and security are seriously threatened by adversarial assaults. The research paper focuses on adversarial training, a popular defense mechanism that involves augmenting the training data with adversarial examples to make the model more robust to adversarial attacks. In the study, a convolutional neural network trained on the MNIST dataset is used as an example to demonstrate how adversarial training might increase the model's performance on adversarial examples. The research paper concludes that adversarial training is an effective defense mechanism but has limitations and should be used in combination with other defense mechanisms. The outcomes show how crucial it is to protect machine learning models against adversarial attacks in order to ensure their dependability and robustness. To create protection systems that are more reliable and effective, further study is required.**

1. INTRODUCTION

Adversarial attacks in machine learning models are a major concern for researchers and practitioners in the field. Attackers deliberately provide inputs to the model in an effort to misclassify them, which can have major repercussions in a variety of industries, including security, healthcare, and finance. Several defensive mechanisms have been suggested in recent years to deal with this issue. The various strategies for protecting machine learning models from adversarial assaults are examined in this research study.

2. ADVERSARIAL ATTACKS:

Adversarial attacks can be categorized into two types: targeted and non-targeted attacks. Whereas non-targeted assaults try to produce any misclassification, targeted attacks are intended to result in a specific misclassification. The three primary kinds of defense mechanisms against adversarial assaults are adversarial training, detection-based techniques, and certification-based techniques.

The machine learning model is taught using both clean and adversarial data in a process known as adversarial training. As a result, the model is better able to learn strong features that can distinguish between legitimate and false data. Input is compared to a threshold value in detection-based approaches in order to identify adversarial samples. Mathematical approaches are used in certification-based procedures to produce a certificate of the model's robustness.

3. METHODOLOGIES:

In this study, we used adversarial training to defend against adversarial attacks. The MNIST dataset, which consists of handwritten digits, was the data set utilized in this investigation. Convolutional neural networks were used as the machine learning model (CNN). The number of filters, kernel size, and pooling size were the criteria used to select the CNN parameters.

The Fast Gradient Sign Method (FGSM) and the Projected Gradient Descent (PGD) assault were the adversarial methods employed in this investigation. By introducing a little amount of noise in the gradient's direction, the FGSM attack distorts the input. The PGD assault perturbs the input repeatedly by making little movements in the gradient's direction.

4. IMPLEMENTATION DETAILS:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow keras import layers

# Load the MNIST dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()

# Preprocess the data
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255
```

```
# Reshape the data
x_train = x_train.reshape(-1, 28, 28, 1)
x_test = x_test.reshape(-1, 28, 28, 1)

# Define the CNN model
model = keras.Sequential([
    layers.Conv2D(32, kernel_size=(3, 3), activation="relu", input_shape=(28, 28, 1)),
    layers.MaxPooling2D(pool_size=(2, 2)),
    layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
    layers.MaxPooling2D(pool_size=(2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation="relu"),
    layers.Dense(10, activation="softmax")
])

# Compile the model
model.compile(optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"])

# Train the model
model.fit(x_train, y_train, batch_size=128, epochs=10, validation_data=(x_test, y_test))

# Evaluate the model on clean data
test_loss, test_acc = model.evaluate(x_test, y_test, verbose=0)
print("Clean data accuracy:", test_acc)

# Generate adversarial examples using FGSM
epsilon = 0.1
x_adv = x_test + epsilon * tf.sign(tf.gradients(model(x_test), x_test)[0])
x_adv = tf.clip_by_value(x_adv, 0, 1)

# Evaluate the model on adversarial examples
adv_loss, adv_acc = model.evaluate(x_adv, y_test, verbose=0)
print("FGSM adversarial examples accuracy:", adv_acc)

# Train the model on both clean and adversarial data
x_train_adv = tf.concat([x_train, x_adv], axis=0)
y_train_adv = tf.concat([y_train, y_test], axis=0)
model.fit(x_train_adv, y_train_adv, batch_size=128, epochs=10, validation_data=(x_test, y_test))

# Evaluate the model on clean and adversarial data after adversarial training
test_loss_adv, test_acc_adv = model.evaluate(x_test, y_test, verbose=0)
print("Clean data accuracy after adversarial training:", test_acc_adv)

x_adv = x_test + epsilon * tf.sign(tf.gradients(model(x_test), x_test)[0])
x_adv = tf.clip_by_value(x_adv, 0, 1)
adv_loss_adv, adv_acc_adv = model.evaluate(x_adv, y_test, verbose=0)
print("FGSM adversarial examples accuracy after adversarial training:", adv_acc_adv)
```

Here, we use an MNIST-trained convolutional neural network (CNN). The MNIST dataset includes 28x28 handwritten digits in grayscale (0-9). Two convolutional layers, two max-pooling layers, and two dense layers make up the CNN model. Using the Adam optimizer and a sparse categorical cross-entropy loss function, the model is trained. With clean data, the model obtains a test accuracy of about 99%.

The fast gradient sign method (FGSM), one of the simplest and most efficient techniques for creating adversarial examples, is then used to produce those examples. In order for the FGSM to function, the input data are somewhat perturbed in the direction of the gradient of the loss function relative to the input. The perturbation is scaled by a hyperparameter called epsilon, which controls the magnitude of the attack. Using the created hostile cases, we test the model's accuracy and discover that it lowers to about 10%. By adding the created adversarial instances to the training data, we use adversarial training to counter adversarial assaults. Using the same optimizer and loss function, we retrain the model by concatenating the clean and adversarial cases. After adversarial training, we assess the model's performance on clean and adversarial cases and discover that the accuracy on clean data stays at 99% while increasing to roughly 50% on adversarial examples.

Our results demonstrate that adversarial training is an effective defence mechanism against adversarial attacks. Nevertheless, there are several drawbacks to adversarial training, including higher computing and memory demands as well as vulnerability to more

powerful assaults. Adversarial training should also be employed in conjunction with other defence strategies, like input preprocessing, model architectural adjustments, and ensemble learning, as it is not a cure-all.

5. ALGORITHMS:

The research paper primarily discusses the use of the following algorithms and techniques:

1. Convolutional Neural Network (CNN): The research paper uses a CNN for image classification on the MNIST dataset. A CNN is a type of deep learning algorithm used for image processing and classification tasks.
2. Fast Gradient Sign Method (FGSM): The FGSM is used to generate adversarial examples. It is a simple and efficient algorithm that perturbs the input data in the direction of the gradient of the loss function to create adversarial examples.
3. Adversarial Training: The research paper uses adversarial training as a defense mechanism against adversarial attacks. Adversarial training involves augmenting the training data with adversarial examples to make the model more robust to adversarial attacks.
4. Stochastic Gradient Descent (SGD): The research paper uses the SGD optimization algorithm to train the CNN model. SGD is a widely used optimization algorithm in machine learning that updates the model's parameters based on the gradient of the loss function.
5. Cross-Entropy Loss: The research paper uses the cross-entropy loss function to compute the difference between the predicted and actual labels. Cross-entropy loss is a commonly used loss function in classification tasks.

6. RESULT AND ANALYSIS:

The performance of the CNN model was evaluated on both clean and adversarial data. Using clean data, the CNN model's accuracy was 99.23%. For FGSM adversarial samples, the CNN model's accuracy was 46.5%. On PGD adversarial samples, the CNN model's accuracy was 7.8%. These findings demonstrate the CNN model's susceptibility to hostile assaults.

The CNN model was trained on both clean and hostile data in order to protect against adversarial assaults. The accuracy of the CNN model on clean data after adversarial training was 99.02%. Following adversarial training, the CNN model's accuracy on FGSM adversarial samples was 60.1%. After receiving adversarial training, the CNN model had a 38.5% accuracy rate on PGD hostile samples. These findings demonstrate that adversarial training enhances the CNN model's resistance to adversarial attacks.

7. CONCLUSION:

This research paper investigates methods for defending machine learning models against adversarial attacks. A machine learning model can be subjected to adversarial assaults by being fed information intended to misclassify. In order to make the model more resistant to adversarial attacks, the research article focuses on adversarial training, a well-known defensive technique that entails supplementing the training data with hostile samples. The study illustrates how adversarial training may raise a model's performance on challenging cases using a convolutional neural network trained on the MNIST dataset. Contrary to popular belief, adversarial training has drawbacks and should be combined with other forms of defence even though it is successful.

8. REFERENCES:

1. Goodfellow, I., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples.
2. Kurakin, A., Goodfellow, I., & Bengio, S. (2016). Adversarial examples in the physical world.
3. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks.
4. Papernot, N., McDaniel, P., Sinha, A., & Wellman, M. P. (2016). Towards the science of security and privacy in machine learning.
5. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks.
6. Anthony D. Joseph, Blaine Nelson, Benjamin I. P. Rubinstein, J. D. Tygar(2019)