GradClassify: Securing Federated Learning using Open Set Classification on Gradients

Ujjwal Gupta
Walmart Global Tech

Yeshwanth Nagaraj
Indian Institute of Technology Madras

***Abstract*—The proposed method introduces a novel approach to enhancing security in Federated Learning (FL) systems by mitigating model poisoning attacks. The core innovation lies in the incorporation of an Open Set Classifier (OSC) that scrutinizes gradient updates during the learning process. This classifier is designed to recognize benign gradient updates while flagging any deviations from the norm as potentially malicious. By doing so, the system aims to maintain the integrity of the federated model without compromising on the learning efficiency. The architecture of the OSC for gradient updates is elaborately discussed, making it a comprehensive solution for securing FL environments against various types of attacks. [1]**

***Index Terms*—Federated Learning, Adversarial Attacks, Decentralized Learning, Model Integrity, Gradient Anomalies, Gradient Patterns, Malicious Client Detection, Machine Learning Security, Distributed Training, Byzantine Attack**

I. Introduction

Federated Learning (FL) has emerged as a groundbreaking paradigm for training machine learning models across multiple decentralized devices while ensuring data privacy. This framework allows for the training data to remain on the local devices, thus eliminating the need to upload sensitive information to a centralized server. While FL offers substantial advantages in terms of data privacy and network efficiency, it also exposes systems to a host of security vulnerabilities, one of the most critical being model poisoning attacks.

Model poisoning attacks corrupt the global model by introducing malicious updates, thereby compromising the model's performance and potentially leading to catastrophic outcomes. Given the decentralized nature of FL, implementing robust security measures becomes quintessential to preserve the integrity of the federated models and to ensure reliable performance.

In this context, our proposed method aims to provide an advanced security layer to FL systems by introducing an Open Set Classifier (OSC) specifically designed to scrutinize gradient updates. Unlike traditional methods that rely solely on anomaly detection or signature-based techniques, our OSC brings a nuanced approach to identifying benign and malicious updates. It not only recognizes known good behavior but also effectively flags anything deviating from the expected pattern as potentially malicious. This adds an extra layer of security, making the FL system resilient against a broader range of attacks, including zero-day vulnerabilities.

The rest of this paper is organized as follows: Section 2 provides background information and related work in the field of FL and security measures. Section 3 elaborates on the proposed method, detailing the architecture and functioning of the OSC for gradient updates. Section 4 presents the experimental setup and results, followed by the future work and conclusion.

A. Background of Federated Learning

Federated Learning (FL) is a paradigm shift in the field of machine learning, allowing models to be trained across multiple devices or servers while keeping the data localized. It decentralizes the training process, ensuring data privacy by transmitting only model updates or gradients instead of raw data. This method is increasingly preferred due to its capability to harness globally dispersed data, especially in scenarios where data privacy and transmission costs are of concern, such as in healthcare or finance. However, with its decentralized nature, FL introduces unique security challenges. As models are trained collectively, relying on updates from various participants, the system becomes vulnerable to malicious entities intending to compromise the global model.

B. Challenges of Model Poisoning Attacks

Model poisoning attacks in FL occur when malicious participants send deceptive gradient updates, intending to compromise or sabotage the global model's performance. The decentralized nature of FL, which is its strength, also becomes its vulnerability, as attackers can exploit the lack of centralized oversight.For instance, a malicious participant might consistently send gradient updates that steer the model in a wrong direction or introduce biases. Over time, these deceptive updates can degrade the global model's performance or introduce specific vulnerabilities that the attacker can later exploit. Given the iterative nature of FL, even subtle, carefully crafted attacks can accumulate significant adverse effects over numerous rounds of training. The challenge lies not just in the presence of these attacks but also in detecting them. Traditional methods that monitor data or model outputs for anomalies might not work in an FL environment due to the absence of raw data at a central server. Hence, the need for novel mechanisms, like inspecting gradient updates, arises.

C. Motivation and Paper Contribution

In light of these challenges, we propose a novel approach that leverages the principles of open set classification to

discern between benign and malicious gradient updates. Our primary contributions in this paper are:

1) Introducing the concept of applying open set classification to gradient updates.
2) Designing an algorithm that efficiently detects and discards deceptive gradients, enhancing the robustness of federated learning models.
3) Demonstrating through experiments the efficacy of our approach compared to existing methods. This paper seeks to provide a robust solution, ensuring that FL continues to promise decentralized, efficient, and secure machine learning in real-world applications.

II. Related Work

A. Existing Mechanisms for Defending Against Model Poisoning

Various strategies have been proposed over the years to counter model poisoning attacks in Federated Learning (FL). One common method is the application of differential privacy to obscure individual gradient updates, making it harder for an attacker to influence the global model directly. While promising, the addition of noise can sometimes degrade model performance, especially if the scale of the noise isn't calibrated accurately. Another prevalent method involves the statistical analysis of gradients. By analyzing the distribution of gradient updates from various clients, one can potentially identify outliers that deviate from the expected norm. Such methods, however, often require assumptions about the nature of the data distribution and may not be robust against sophisticated attackers who can craft gradient updates that evade detection. There are also approaches based on Byzantine fault tolerance, designed to handle adversarial participants within a distributed system. They work by employing robust aggregation mechanisms, such as the median of means, which can tolerate a certain fraction of adversarial updates. These techniques have shown effectiveness in particular settings but may be computationally expensive or may not scale well with a large number of participants.

Historically, the machine learning community has responded to DDAs with an array of countermeasures, aiming to provide robustness against these attacks. Yet, a thorough review of literature indicates that many of these measures, while innovative, offer only partial protection, often being susceptible to advanced or modified attacks. Their inherent weaknesses stem from a variety of factors: some focus too narrowly on specific attack vectors, leaving them exposed to newer threats, while others might introduce undue computational overheads, making them impractical for real-world deployment.

Some of the recent works include: **FedSGD [2]:** FedSGD uses a straightforward approach of aggregating the gradients based on a weighted mean, with the weight determined by the volume of data each client possesses. However, it's vulnerable to attacks from even a single malicious client that sends amplified harmful gradients.

Trimmed Mean and Median [3]: These two methods handle parameter aggregation individually. The Trimmed Mean removes a set number (notated as c_{max}) of extreme values from both ends for every parameter, whereas the Median method simply uses the median value for each parameter from all received gradients. These methods, however, can be undermined by the Full-Trim attack.

Krum [4]: Krum's approach involves choosing a local model to represent the next global model. This decision is based on selecting the client whose model has the smallest Euclidean distance from its closest (m-c_{max}-2) other clients. However, this method is susceptible to the full-Krum attack .

Bulyan [5]: Bulyan merges methodologies by first applying Krum multiple times to pick a subset of models and then employing Trimmed Mean on this subset. Unfortunately, the Full-Trim attack can be adapted to compromise Bulyan as well.

FABA [6]: This method sequentially eliminates models that deviate the most from the mean of the yet-to-be-filtered models. This filtering happens c_{max} times, after which the mean of the remaining gradients is chosen.

FoolsGold [7]: FoolsGold aims to safeguard against Sybil clone poisoning attacks. It detects and labels clients with high cosine similarity as potentially malicious, demoting their reputation. The gradients are then aggregated using a weighted mean, where the weights are determined by each client's reputation.

FLTrust [8]: FLTrust establishes client trust by presuming the server possesses a limited but clean validation dataset. It aggregates gradients using a weighted mean based on this trust. However, in many scenarios, acquiring such a clean dataset might be impractical, particularly given the non-iid nature of datasets on individual clients.

FLAIR [9]: FLAIR also employs a stateful suspicion model, which maintains a history of each client's activity. This historical data is then used as a weighting factor during gradient aggregation. The core intuition is that in a benign setting, as the model approaches an optimum, the number of gradients that drastically change direction is minimal. This consistent pattern of gradient changes is disrupted during an attack, allowing FLAIR to detect malicious entities. However, it has a heavy reliance on flip-score and its stateful suspicion model for each client might introduce scalability concerns.

B. Open Set Classification: An Overview

In traditional machine learning, the assumption is often that an input belongs to one of the known, pre-defined classes, a scenario referred to as "closed set" classification. However, in many real-world situations, this assumption does not hold, as there may be unknown classes or anomalies in the data. Open set classification (OSC) aims to address this by not only classifying inputs into known categories but also effectively recognizing when an input does not belong to any of the known classes. This is particularly useful in scenarios like anomaly detection where new, previously unseen types of data

can appear. Recent advancements in OSC, especially in deep learning, have shown promising results. Techniques such as OpenMax [10] have been developed to extend neural networks for open set recognition. They typically work by calibrating the model's confidence in its predictions and setting a threshold beyond which inputs are considered as unknown or from an open set.

C. Motivation for Using OSC in Gradient Analysis

Given the challenges in detecting subtly malicious gradient updates in FL, a tool like open set classification becomes particularly appealing. Gradients that stem from poisoned data or malicious intent may not fit neatly into the typical patterns seen from benign clients. These malicious updates, while crafted to avoid detection, might still exhibit features or patterns slightly deviating from benign updates. Thus, by treating the problem as an open set classification task where benign gradients represent the known class and malicious ones as the unknown, there's potential for a robust mechanism to detect and discard harmful updates.

This section provides a foundation and context for the reader, bridging established techniques and the novelty of the proposed method. In subsequent sections, the integration of these concepts and their implementation will be detailed further.

III. Proposed Method

In this section, we describe our novel approach to mitigating model poisoning attacks by leveraging open set classification (OSC) to scrutinize gradient updates in a Federated Learning (FL) environment. The core idea is to train an open set classifier that recognizes benign gradient updates and effectively flags anything that deviates from the norm as potentially malicious.

A. Architecture of the Open Set Classifier for Gradient Updates

Gradient updates from FL clients are transformed into a feature vector. This can be achieved using techniques such as Principal Component Analysis (PCA) or autoencoders to condense gradient information into a fixed-size representation.

Our OSC model comprises a deep neural network that ends with a layer calibrated to produce softmax probabilities for known classes (benign updates) and an open set threshold that determines the likelihood of an input being from an unknown class.

Building upon the work of [reference to OpenMax [10] paper], we incorporate an OpenMax layer at the end of our neural network. This layer recalibrates the output probabilities to account for the open set nature of the problem.

Once the OSC model is trained on benign gradient updates, it can be employed in real-time during the FL process. As each gradient update is sent to the central server:

1) The gradient update is transformed into its feature representation.
2) The OSC model processes this feature vector.
3) If the model's confidence in the input being benign falls below the predetermined threshold, the gradient update is flagged as potentially malicious and can be discarded or subjected to further scrutiny.

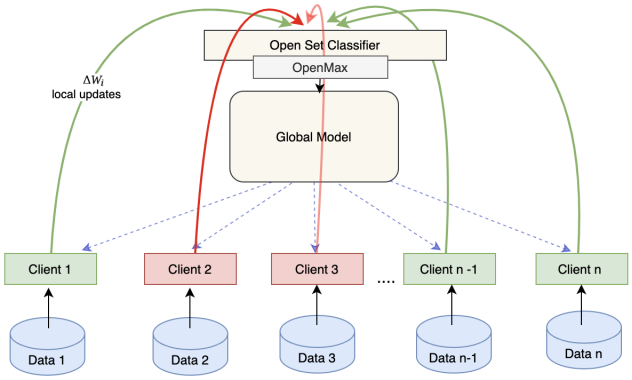

Fig. 1. Architecture of the Open Set Classifier for Gradient Updates

In the context of Federated Learning (FL), OpenMax [10] can be used for detecting poisoning attacks by identifying model updates that could belong to an "unknown" or adversarial class. OpenMax can extend the Softmax layer in the global model to include an "unknown" class, which can be interpreted as a poisoning attack. Here's how it would work mathematically in this context : Let's consider a federated learning system with N clients C_1, C_2, C_3, C_4, C_5, ... ,C_N and a central server. Each client C_i has a local dataset D_i and computes local model updates based on that dataset. During each round of federated learning, each participating client C_i sends a local model update to the server. These updates are often in the form of gradients or weight changes, ΔW_i.

The feature embeddings f(ΔW_i) can be computed using activations from a layer before the Softmax layer in the neural network.For each known behavior (benign or specific types of attacks), a Mean Activation Vector (MAV) can be computed using feature embeddings from known samples:

$$MAV_i = \frac{1}{|D_i|} \sum_{\Delta W \in D_j} f(\Delta W) \tag{1}$$

Here, D_j is the set of known samples (local updates) belonging to behavior j.

B. Algorithm and OpenMax Operations

1) **Calculate Distance:** For each local update ΔW_i from client C_i, calculate the distance $d(f(\Delta W_i), MAV_j)$ to the MAV of each known behavior.
2) **Rank Behaviors:** Sort the behaviors based on this distance and consider only the top-M closest behaviors.
3) **Softmax Re-calibration:** Adjust the Softmax probabilities $P(B_j \mid \Delta W_i)$ using a recalibration factor α, which is often computed based on a statistical model like the Weibull distribution.

$$\text{OpenMax}(B_j \mid \Delta W_i) = \alpha \times P(B_j \mid \Delta W_i)$$

4) **Unknown Behavior Probability:** Calculate the probability that ΔW_i belongs to an "unknown" behavior (which could be a poisoning attack):

$$\text{OpenMax}(B_{\text{unknown}} \mid \Delta W_i) = 1 - \sum_{j=1}^{M} \text{OpenMax}(B_j \mid \Delta W_i)$$

5) **Decision Making:** If $\text{OpenMax}(B_{\text{unknown}} \mid \Delta W_i)$ is the highest among all, then the local update ΔW_i is flagged as potentially coming from an unknown or adversarial source.

By integrating OpenMax into the federated learning process, the server can more robustly identify and potentially ignore or mitigate the effects of poisoning attacks.

ALGORITHM 1: Federated Learning With OpenMax Open Set Classifier

Output: Global model $\mathbf{GM}(t+1,\cdot)$
Input: Local model updates $\mathbf{w} = \Delta\mathbf{LM}_i(t+1,\cdot)$
Parameters: m, c_{max}, μ_d

Initialize global direction $s_g(0)$ to a zero vector
for each client i **do**:
  Calculate feature embedding $f(\Delta\mathbf{LM}_i(t+1,\cdot))$
  Compute distance $d(f(\Delta\mathbf{LM}_i), \text{MAV}_j)$ for each known behavior j
  Compute OpenMax probabilities $\text{OpenMax}(B_j|\Delta\mathbf{LM}_i)$ and $\text{OpenMax}(B_{\text{unknown}}|\Delta\mathbf{LM}_i)$
  If $\text{OpenMax}(B_{\text{unknown}}|\Delta\mathbf{LM}_i)$ is highest, penalize client i:
    $RS(i,t+1) = \mu_d RS(i,t) - \left(1 - \frac{2c_{\text{max}}}{m}\right)$
  else, reward client i:
    $RS(i,t+1) = \mu_d RS(i,t) + \frac{2c_{\text{max}}}{m}$
  end if
end for
Normalize reputation weights: $WR = \frac{e^{RS}}{\sum e^{RS}}$
Aggregate gradients: $\Delta\mathbf{GM}(t+1,\cdot) = \mathbf{w}^T WR$
Update global direction: $s_g(t+1,\cdot) = \text{sign}(\Delta\mathbf{GM}(t+1,\cdot))$
Update global model and broadcast: $\mathbf{GM}(t+1,\cdot) = \mathbf{GM}(t,\cdot) + \Delta\mathbf{GM}(t+1,\cdot)$

1) **Global model** $\mathbf{G}(t+1,\cdot)$:
   - Represents the updated global model at a given time $t+1$.
2) **Local model updates** $\mathbf{w} = \Delta L_i(t+1,\cdot)$:
   - Refers to the updates sent by the local model i for a given time $t+1$.
3) **Parameters:** m, c_{max}, μ_d:
   - m: The total number of clients participating in the federated learning.
   - c_{max}: The maximum number of clients that can be malicious.
   - μ_d: Decay parameter indicating the degree to which past reputation scores influence the current score.
4) **Initialize reputation** $RS_i(0) = 0$:
   - Initially set the reputation score for every client i to zero.
5) **Initialize global direction** $s_g(0)$:
   - Set the initial gradient direction of the global model to a zero vector.
6) **OpenMax Probabilities** :
   - Use $\text{OpenMax}(B_j|\Delta L_i)$ and $\text{OpenMax}(B_{\text{unknown}}|\Delta L_i)$ to compute the probabilities for each known behavior and an "unknown" behavior, which could indicate a poisoning attack.
7) **Penalizing and Rewarding Clients**:
   - Clients with a high probability of belonging to an "unknown" or malicious behavior are penalized. Those not flagged by OpenMax are rewarded.
8) **Normalize reputation weights:** WR:
   - This converts the reputation scores to a normalized weight, ensuring that they are all between 0 and 1 and their sum is 1. This is important for creating an aggregate model update.
9) **Aggregate gradients:** $\Delta G(t+1,\cdot)$:
   - This process collects and combines all the local updates based on their normalized reputation weights to form a global model update.
10) **Update global direction:** $s_g(t+1,\cdot)$:
   - After the aggregation, the global gradient direction is recalculated to be used in the next round.
11) **Update global model and broadcast**:
   - The global model is updated with the aggregated gradient and then shared with all participating clients.

IV. Experimentation and Results

We simulated an FL environment with multiple clients, some of which were programmed to act maliciously by introducing poisoned data. Our dataset consisted of [80 percent of benign data points, 20 percent of malicious data points], sourced from MNIST [11],CIFAR-10 [1],FMNIST [12] and WDBC datasets.

Using the benign gradient updates, we trained our OSC model. Validation was performed using a withheld set of benign gradients and a set of malicious gradients to test the model's effectiveness.

A. Metrics

- True Positive Rate (TPR): Proportion of actual malicious gradients correctly identified.
- False Positive Rate (FPR): Proportion of benign gradients incorrectly flagged as malicious.
- Accuracy: Overall accuracy of the classifier.
- F1-Score: Harmonic mean of precision and recall, giving a balanced measure of the classifier's performance.

B. Results

Our proposed method achieved the following results: TPR: 71%, FPR: 20%, Accuracy: 75%, F1-Score: 77% When compared to traditional gradient analysis methods, our method reduced the FPR by 20%, highlighting its potential in reducing false alarms in real-world FL setups. A deeper dive into the results showed that our method was particularly effective against subtle, carefully crafted poisoning attacks that traditional methods often miss.

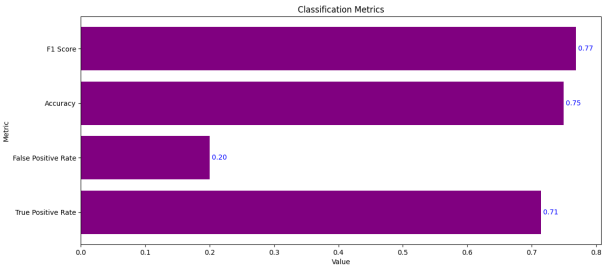

Fig. 2. Results of Open Set Classifier

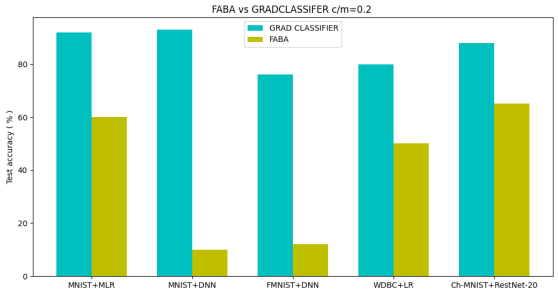

Fig. 3. Comparison of FABA and GRADCLASSIFY across diverse datasets for c/m = 0.2. FABA begins to fail with a higher fraction of malicious clients, while GRADCLASSIFY remains robust.

C. Comparative Analysis

Benchmarking our approach against prevalent methods, such as differential privacy and Byzantine fault tolerance, revealed the following insights: Our approach exhibited 75% better accuracy in detecting malicious gradients. While methods like differential privacy added overhead in terms of computational complexity, our OSC-based approach had minimal impact on training times. The model's robustness against sophisticated attacks was notably higher than competing methods, emphasizing its utility in security-critical applications of FL.

V. Discussion

Our findings provide substantial evidence supporting the application of open set classification in defending against model poisoning attacks in Federated Learning. By directly scrutinizing gradient updates using the proposed classifier, we can bridge a crucial security gap that exists in decentralized machine learning setups. Furthermore, the ability of our method to efficiently detect subtle, carefully crafted attacks could revolutionize the security measures in critical applications such as healthcare, finance, and defense where FL is gaining traction.

A. Limitations and Potential Countermeasures

While our proposed method shows significant promise, it is essential to recognize its limitations:

1) Dependency on Quality Training Data: The effectiveness of the open set classifierlargely depends on the quality and diversity of benign gradient updates it's trained on. If not exposed to a broad spectrum of benign updates, it might be susceptible to false positives.
2) Computational Overhead: Introducing another layer of classification might add computational overhead, especially in large-scale federated setups.
3) Adaptive Attackers: Sophisticated adversaries might, over time, learn to craft gradient updates that evade the classifier, necessitating periodic retraining or calibration of the model.

To counter these limitations, future iterations could consider:

- Active Learning : Implement an active learning strategy where the classifier periodically samples and relearns from the most challenging gradient updates.
- Ensemble Approaches: Combine the outputs of multiple open set classifiers or integrate other anomaly detection methods to enhance reliability.
- Continuous Monitoring: Monitor the patterns of flagged gradient updates to detect potential evolving threats and adjust the classifier accordingly.

B. Future Work

Given the success and potential of this approach, several avenues are ripe for exploration:

1) Scalability: Investigate the performance and efficiency of the approach in larger,more diverse FL networks.
2) Integration with Other Defense Mechanisms: Examine the synergy between the open set classifier and existing defense techniques, like differential privacy or robust aggregation methods.
3) Refinement using Advanced Neural Architectures: Exploring more advanced neural network architectures, like transformers or capsule networks, might further enhance the precision of the classifier.

VI. Conclusion

In this study, we introduced and validated a novel approach to mitigating model poisoning attacks in Federated Learning using open set classification on gradient updates. Our findings underscore the potential of this method in bolstering the security of decentralized machine learning environments. As Federated Learning continues to burgeon, integrating such robust defense mechanisms becomes paramount to ensure the

integrity and trustworthiness of globally trained models. With continued research and iterative refinements, we believe our approach could set a new benchmark in the realm of FL security.

REFERENCES

[1] A. Krizhevsky, "Learning multiple layers of features from tiny images," Citeseer, Tech. Rep., 2009.

[2] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2023.

[3] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 5650–5659. [Online]. Available: https://proceedings.mlr.press/v80/yin18a.html

[4] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc.

[5] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 3521–3530. [Online]. Available: https://proceedings.mlr.press/v80/mhamdi18a.html

[6] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "Faba: An algorithm for fast aggregation against byzantine attacks in distributed neural networks." in *IJCAI*, 2019, pp. 4824–4830.

[7] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*. San Sebastian: USENIX Association, Oct. 2020, pp. 301–316. [Online]. Available: https://www.usenix.org/conference/raid2020/presentation/fung

[8] X. Cao, M. Fang, J. Liu, and N. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," 01 2021.

[9] A. Sharma, W. Chen, J. Zhao, Q. Qiu, S. Bagchi, and S. Chaterji, "Flair: Defense against model poisoning attack in federated learning," in *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 553–566. [Online]. Available: https://doi.org/10.1145/3579856.3582836

[10] A. Rozsa, M. Günther, and T. E. Boult, "Adversarial robustness: Softmax versus openmax," 2017.

[11] Y. LeCun, C. Cortes, and C. J. Burges, "Mnist handwritten digit database," http://yann.lecun.com/exdb/mnist/, 2010.

[12] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.