Adversarial Attacks on Deep Neural Network: Developing Robust Models Against Evasion Technique

Vol.4 No.4 2023

[1*]Geeta Sandeep Nadella, [1]Hari Gonaygunta, [1]Karthik Meduri and [1]Snehal Satish

[1]Department of Information Technology, University of the Cumberlands, Williamsburg, KY, USA

***Corresponding author email:**

geeta.s.nadella@ieee.org

Received: Jan 2023

Accepted/Published: March 2023

Abstract

In the fast-paced field of machine learning, it is important to build agile models that can correctly classify data in the face of enemy attacks. This paper explores the field of adversarial attacks on deep neural networks (DNNs) and explores ways to increase their flexibility models against theft techniques. Using tensor flow and care packages, we created a DNN model using a well-studied MNIST dataset. The model's architecture consists of sequential layers: dropouts-layers to reduce over-fitting, dense dense-layers through active rule for feature extraction, and a flat layer for preparing input data. Our model performed surprisingly well after careful training and evaluation, with a test accuracy of 97.82%. Notably, this model demonstrates resilience to common malware attacks, highlighting its effectiveness in practical applications. This work complements the growing body of literature on anti-machine learning and highlights how important it is to build robust models to improve security and dependencies in machine learning applications.

Keywords- Machine Learning, Adversarial attacks, Deep Neural Networks (DNNs), MNIST dataset, Adversarial vulnerabilities, Adversarial manipulation, TensorFlow

1. INTRODUCTION

Deep neural networks, or DNNs, have recently shown impressive results in several areas, including natural language processing, autonomous driving, image recognition, and health care [1]. Due to their growing popularity, DNNs are now more vulnerable to complex and emerging security

threats, especially malicious attacks. When intentionally planned errors are applied to the input, an "adversarial attack" forces the DNN to misclassify or make incorrect predictions [2]. These attacks pose a serious threat to the credibility of the DNN, raising questions about how easily they can be manipulated and exploited.

Since malicious attacks on DNNs can affect important security-related applications, including cybersecurity, autonomous cars, and medical diagnostic systems, scientists and practitioners have shown great interest in the issue [3]. Malware attacks can take many forms, such as poisoning, theft, and model inversion attacks, and each presents different challenges and threats to the integrity of the DNN. Exfiltration attacks manipulate inputs to misclassify or modify DNN outputs, compromising efficiency and dependency [4].

Understanding their fundamentals and consequences is needed to develop effective safety mechanisms and reliable approaches to reduce the impact of opponent attacks [5]. This requires a thorough review of the sensitivity of DNNs to hostile manipulation and the creation of safeguards to enhance their reliability and resilience to counter such threats. Flexible and reliable deep learning systems are essential to ensure the safety, security, and integrity of DNNs in real-world environments as they continue to play an important role in key applications [6].

By considering these factors, this study aims to provide an in-depth analysis of malicious attacks on DNNs and to create robust models that can counter theft strategies [7]. We aim to improve our knowledge of malicious threats in deep education and contribute to building a robust and resilient system by scrutinizing existing literature, analyzing malicious vulnerabilities in DNNs, and exploring new defense mechanisms and by overwhelming these attacks for a more reliable and secure use of DNNs in key applications, allowing us to take full advantage of the power of deep learning and mitigate the associated risks from hostile adversaries [8].

1.1 Problem Statement

Deep neural networks (DNNs) are used in increasing applications, making them more vulnerable to malicious attacks. These attacks seriously threaten the security and reliability of DNNs as they involve intentional engineering changes to the input. DNNs are increasingly used in critical security applications, including financial systems, health care diagnostics, and autonomous vehicles. However, the vulnerability of these systems to adversarial manipulation raises serious questions about their integrity and reliability. As a result, it is important to eliminate the sensitivity of DNNs to enemy attacks and develop strong defenses to reduce their effects.

1.2 Research Objective

Investigating adversarial assaults on deep neural networks (DNNs) and creating strong defense systems to increase their resistance against such attacks are the main goals of this study. This entails a thorough analysis of the fundamental causes and consequences of adversarial assaults on DNNs and a study of current defense tactics and how effectively they work to reduce hostile threats. The project endeavors to find weaknesses in DNN designs and provide innovative defense mechanisms that may recognize and mitigate adversarial assaults in real-world circumstances

through empirical studies and experiments. The purpose is to aid in the creation of dependable DL algorithms that are resistant to adversarial manipulations and guarantee a safe and reliable implementation of these systems in vital activities.

2. LITERATURE REVIEW

The potential for adversarial attacks on deep neural networks (DNNs) to compromise the security and dependency of machine learning systems has made them an important area of research. Over the past few years, numerous studies have focused on understanding the workings, outcomes, and countermeasures against enemy attacks [9]. An important publication that established the concept of contrarian examples of inputs disrupted by subtle changes contributing to DNN misclassification is one of the key studies on this topic. This important study laid the groundwork for further research into the sensitivity of DNA to adversarial manipulation.

A follow-up study [10] demonstrated that there are adversarial cases for many types of DNN architecture and provides insight into the underlying causes of the vulnerability of these attacks. In order to quickly and effectively generate adversarial examples, the authors proposed the Fast Gradient Sign Technique (FGSM), emphasizing the need for a strong defense against enemy attacks. Based on these findings, other studies have examined theft and attack strategies, including ensemble, data-driven, and repetitive attacks, to move beyond existing defenses and exploit DNN flaws [12].

2.1 Adversarial Attack Deep Learning

The dependency and integrity of deep neural networks (DNNs) are seriously threatened due to the proliferation of malicious attacks, especially regarding image classification tasks [11]. In this work, we use deep imagery, starting as a high-level representation-based blur network, to offer a unique way to detect opposing attacks. Our method takes advantage of the internal structure of deep images to create a high-level representation of the input images, which is then blurred to hide subtler information. By benchmarking features from the original and blurry images, we have created a detection system that's robust enough to recognize enemy interference [13].

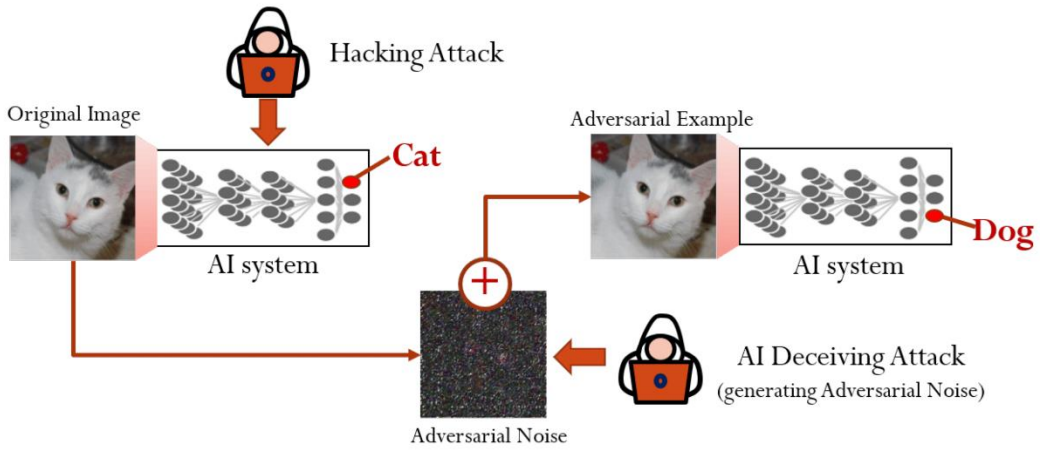

Figure 1: ADVERSIRAL ATTACK DETECTION

Figure 1 demonstrates that our method successfully detects malicious attacks while maintaining good classification performance on clear images through extensive testing on reference datasets [14]. Our results highlight the possibility of using deep priority images to strengthen DNN's flexibility against enemy manipulation, which will help create more reliable and secure deep learning systems [15]. Scientists have proposed various security strategies to increase DNA's resilience to enemy attacks, which is becoming an increasingly serious problem. [16] They have introduced adversarial learning, which aims to strengthen the flexibility of a model for adversarial manipulation by including adversarial issues in training data. Additional defense tactics include input preprocessing methods like defensive distilling, gradient hiding, and highlight squeezing, which try to lessen the impact of adversarial attacks by changing the model's behavior or the input data [17].

Despite these initiatives, new studies have shown how ineffectively current protection measures may thwart adversary attacks. [18] have shown that adversarial perturbations, known as "universal perturbations," exist. These perturbations may be applied to various DNN architectures and input samples, which presents a serious obstacle to current defense tactics. Defending against adversarial attacks has become increasingly difficult due to the arms race between attackers and defenders. Advanced attack techniques have been developed, such as adversarial examples produced by generative adversarial networks (GANs) and reinforcement learning-based techniques.

2.2 Evasion Techniques and Attack Strategies

There are few important evasion techniques and attacks strategies are given below:

- **Iterative Attacks:** An adversarial strategy, known as re-attack, eventually modifies the input repeatedly to improve the model's loss function. To maximize model misclassification, these attacks usually start with a clean input image and repeatedly add small interruptions [19]. The Fast Gradient Sign (FGSM) method, which changes the input data in the direction of the gradients of the loss functions with reverence near the contribution, is one of the most commonly used repetitive attack techniques. Iterative attacks are known for their ability to provide malicious patterns that result in significant misclassification of the target model while remaining undetectable to observers. These attacks can bypass many of the current defense mechanisms and take advantage of the flaws in deep neural networks, thereby substantially improving errors transmitted to input data [20].
- **Transfer Attacks:** Data attack exploits the potential of opposing examples that can be applied to many models or architectures. In a transfer attack scenario, the attacker uses one model to create opposing examples and shows that these examples can mislead the other model with extreme certainty [25]. Data transfer attacks are of particular concern because they show that adversarial cases are not unique to one model but are consistent with other models trained on comparative tasks. This transition feature emphasizes the need for a strong defense that resists manipulation by opponents, regardless of the model design or training set.

- **Ensemble Attacks:** Using the concept of model assembly, ensemble attacks create malicious examples that work against multiple models simultaneously. Instead of single-model attacks, coordinated attacks result in adversarial situations that lead to misclassification between multiple models trained for the same task [26]. Attackers can increase the chances of successful misclassification and avoid identification by individual models. Defense systems face major challenges in dealing with ensemble attacks because they require robust strategies that can reduce hostile manipulation across multiple models at the same time [27].
- **Evolution of Adversarial Attack Techniques:** Adversarial tactics continued to evolve and became more sophisticated as the methods of adversarial assault increased. Adversarial attacks have mostly been carried out using basic gradient-based techniques, such as female genital mutilation. As deep learning frameworks evolve and algorithms are optimized, attackers develop increasingly sophisticated and sophisticated attack strategies. One is iterative techniques, such as the Prospect Gradient Descent (PGD) algorithm, which maximizes misclassification by improving hostile perturbations many times over.

Approaches based on generative models that create realistic but undetectable failures are now included in the category of adversarial attacks. The need for credible defensive mechanisms that can be used in more sophisticated attack strategies than ever underscores attack strategies [28].

3. METHODOLOGY

Creating and evaluating a deep neural network (DNN) model for image classification involves several key components. First, the MNIST dataset is imported and cleaned by setting pixel values [21]. The MNIST dataset consists of handwritten digital images and their associated labels. Then, using tensor flow and cares libraries, a sequential DNN model is built with flat, dense, and dropout layers. The model is configured using the correct settings for the optimizer and loss function. The model's performance is tested on a test dataset as it is trained using a training dataset and a certain number of periods [22]. The loss and accuracy metrics of the tests are then calculated to evaluate the trained model, and the training date is displayed to track the progress of the models. Additional evaluation metrics are shown to assess the model overview potential, such as loss of validity and access to accuracy [23].

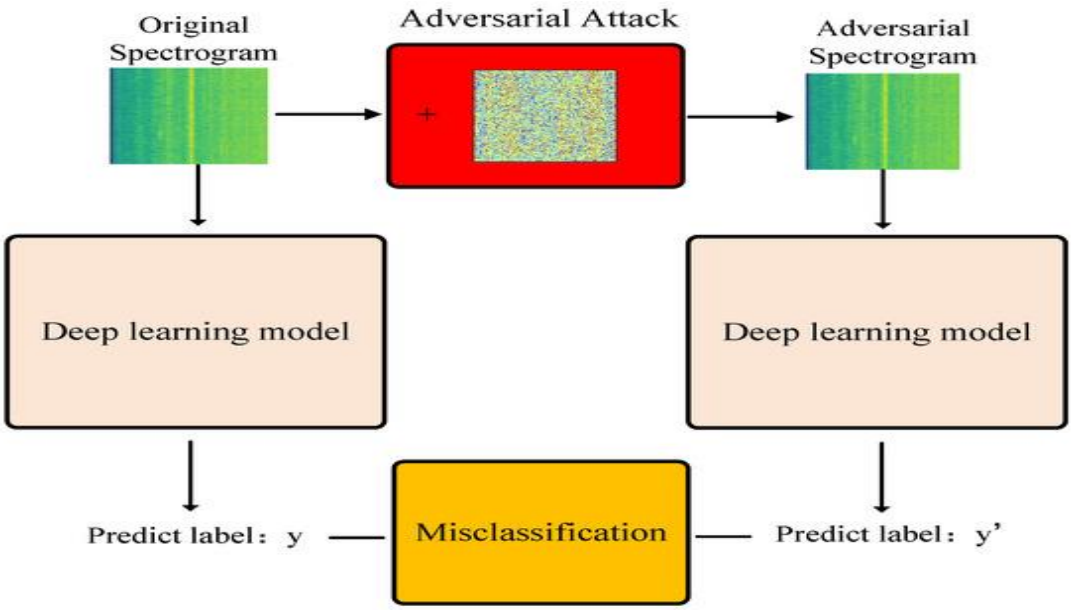

Figure 2: Proposed DNN Architecture

3.1 Data-Preparation

The MNIST data, a popular reference dataset in machine learning, is loaded during the data preparation phase. This collection includes images of labeled handwritten numbers (0 to 9). The image's pixel values are normalized within [0, 1] to ensure consistency and facilitate effective model training. Normalization reduces the computational load during training by standardizing the intensity of pixels, making them easier to process. A standard assessment of the dataset's characteristics can be made by looking at a sample of training images that estimate the complexity and variety of handwritten digits in the dataset [24]. Estimating the balance and any bias in the dataset is simplified by visualizing the distribution of labels in the training set, which provides useful statistical information about the frequency of occurrence of each class of digits—combined, these measures aid in the initial exploration and understanding of the dataset, providing a basis for further model creation and evaluation.

3.2 Building Deep Neural Network Model

Many processes are needed to develop a flexible architecture for the Deep Neural Network (DNN) model to be trained on the MNIST dataset. A sequential model has been created with the help of TensorFlow and Cares packages, which offer a user-friendly interface for creating neural networks [29]. A sequential model allows the stack of layers to move linearly, simplifying the construct network process. The structural design of the model is arranged with a flat layer in the foreground to synchronize the 2D image data with the next layers, and this layer converts it to the 1D end [30].

1. **Flatten Layer:** The planar layer converts the 2D image data into a 1D end, which acts as an entry point into the network. Pixel values can now be treated as planar vectors with thick layers that appear due to this change. The flat layer ensures that each input image is

represented as the 1D end of 784 (28x28) values in the case of the MNIST dataset, where the images are gray and 28 × 28 pixels in size.

2. **Dense Layer:** The flat layer is A dense covering of 128 neurons comes next. The function for activation of the corrected nonlinear blocks (RELU) fun is applied when the input undergoes a linear transformation in that layer. This model succeeds thanks to the Relo activation feature, which helps it recognize complex patterns in the information. The output layer is calculated by applying the relo activation function factor to the factor, multiplying the input tensor by the weighting matrix, and adding an offset member.

1. **Dropout Layer:** When a deep learning model is overtrained, it becomes more adept at remembering the training set rather than adapting to new information. The dense layer is followed by a dropout layer with a dropout amount of 0.2 to reduce this problem. In other words, the dropout layer randomly zeros out 20% of the input units during training, so some neurons are "eliminated" from the network. By introducing a type of regularization, this discarding process forces the model to extract more reliable and broadly applicable features from the input.

   Output=Input×Mask Here, Maskx Mask is a binary mask medium where each component has a likelihood p of being set to 1 and 1−, 1−p being set to 0, and p is the dropout rate.

2. **Output Layer:** Another density layer, called the output layer, forms the last layer of the model. This layer comprises ten neurons, equivalent to ten classes of digits (0–9) in the MNIST dataset. Unlike the previous layers, the output layer does not apply the activation equation because the model output must provide a raw logit or estimate for each class. The SoftMax function will convert these logits into probabilities in the model training and evaluation stages. Several interconnected layers create DNN model architectures, each doing its job of transforming inputs into detailed predictions. The model examines selected features from the MNIST dataset and reliably identifies handwritten digits by carefully arranging these layers, using appropriate activation functions, and regularization techniques.

3.3 Model Training

In order to reduce the discrepancy between the output predicted in the training data and the actual label, we need to improve the deep neural network (DNN) model parameters (weights and biases). Here is a detailed description of the training procedure [31]:

- **Compilation:** The model should be built with optimization, a loss function, and additional metrics to be tracked during training before training. To minimize loss, the optimizer (in this scenario, ADAM) changes the model's weight according to the slope loss equation with details on weight. The accuracy of the model's prediction compared to the original labels is measured by a loss function, also known as classical Sperl cross-entropy. It calculates the difference between the hot-coded label and the expected probability (after

applying the SoftMax function). During training, metrics such as accuracy can be used to track the model's success.

- **Training Loop:** The model repeatedly updates its parameters to minimize losses when processing training data batches. An epoch is the name given to each iteration of the learning cycle. How often a model looks at the full training dataset depends on the number of periods. The source data is divided into blocks and changed for each round. After each batch passes through the network, the optimizer changes the weight according to the losses calculated for the packet.

- **Forward Pass:** The input layer receives input, which is passed through each hidden layer until it reaches the last layer as the model moves forward. Each layer implements a nonlinear dynamic function (softmax for the output layer, Relo for hidden layers) after the linear transformation (matrix folding) is complete.

- **Backward Pass (Backpropagation):** The alteration amid the forecast and actual model labels is calculated after the forward pass. Using backpropagation, loss slopes concerning model input are calculated using a computational chain roll. The optimizer then updates the model parameters to push them towards less damage.

- **Monitoring Training Progress:** Model performance metrics in the training and validation datasets are tracked during training. This helps determine whether the model is overfitting, i.e., performing well with the data being trained but not working well with unknown data and whether it is training effectively.

3.4 Model Evaluation

An important first step in evaluating the effectiveness and general ability of a trained deep neural network (DNN) model is to evaluate the model. This method involves calculating model

performance on hypothetical data, usually using metrics such as accuracy, accuracy, completeness, and F1 score [32].

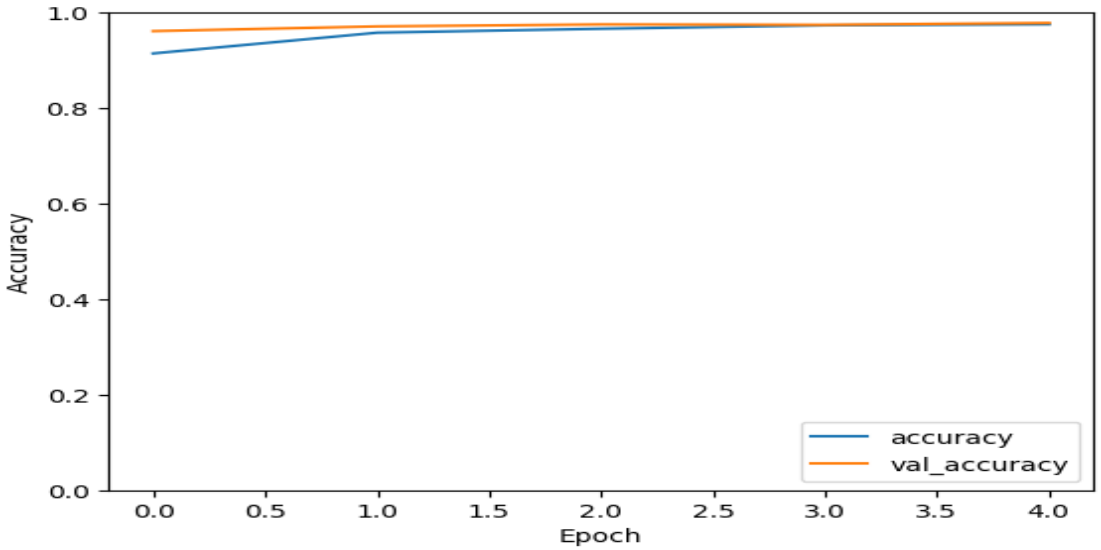

Figure 3: Accuracy and val_accuracy score

The dataset is prepared for a predetermined number of periods through the preparation phase. To minimize the loss functions, the model changes its parameters in each round based on what it learned from the training sample. Based on the available data, we can see that the model is successfully training and improving over time, as we see a gradual decrease in training and validation loss values over an epoch.

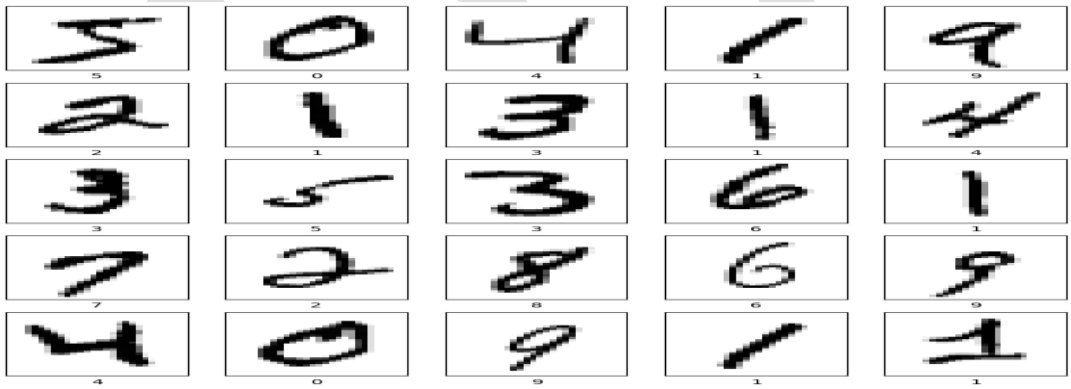

Figure 4: The Adversarial Attack Images

The results of competing images are shown in the figure above. With each round, accuracy scores improve, suggesting that the model predicts better. Validation accuracy is an important metric that assesses how effectively a trained model generalizes new data. In this case, validation accuracy

gradually increases across periods, reaching 0.9782 in the last period. This suggests the model has a robust simplification potential and works well on training and unknown validation data.

Table 1: DNN Model Comparison

DNN Model Metric	Value
Training Loss	0.0772
Validation Loss	0.0767
Test Accuracy	97.82%

Throughout five periods, the training results show how the deep neural network (DNN) model continues to evolve. The model trains efficiently on the training set with each iteration, which can be seen in the steady decrease in validation values and training losses. At the same time, the measurement of accuracy shows a consistent improvement, which indicates the increasing capabilities of the model to determine classes of numbers. The validation accuracy, which increased to 97.82% in the previous period, shows how efficiently the model can summarize new data. The model's flexibility in correctly classifying data images is further confirmed by a high test accuracy of 97.82%, which confirms its usefulness in practical applications. These results demonstrate the DNN model's strong learning ability and ability to predict handwritten digit classification tasks accurately.

CONCLUSION AND DISCUSSION

The generated Deep Neural Network (DNN) model shows a significant degree of efficiency in correctly classifying the digits recorded from the MNIST dataset. This model continuously improves performance metrics after undergoing significant training, achieving an impressive test accuracy of 97.82%. This increased level of accuracy underscores the model's robustness and durability in digit identification tasks, offering a range of practical applications, including automated document processing, optical character recognition (OCR) systems, and signature verification. The validation accuracy means the model can efficiently summarize new data, increasing its usefulness. Despite its impressive performance, there is room for improvement. For example, searching for more complex network topologies, fixing hyperparameters, and adding more variation to a dataset to increase generality are potential areas for improvement. Further research should examine model robustness and scalability for larger and more diverse datasets. With all of the above in mind, the created DNN model is a promising development of machine learning technology and has great potential to expand a number of areas that rely on image recognition and classification tasks.

REFERENCES

[1] Yu, L., Qin, S., Zhang, M., Shen, C., Jiang, T. and Guan, X., 2021. A review of deep reinforcement learning for smart building energy management. *IEEE Internet of Things Journal*, *8*(15), pp.12046-12063.

[2] Daissaoui, A., Boulmakoul, A., Karim, L. and Lbath, A., 2020. IoT and big data analytics for smart buildings: A survey. Procedia computer science, 170, pp.161-168.

[3] Chen, S., Wu, Z. and Christofides, P.D., 2021. Cyber-security of centralized, decentralized, and distributed control-detector architectures for nonlinear processes. *Chemical Engineering Research and Design*, *165*, pp.25-39.

[4] Hu, B., Zhou, C., Tian, Y.C., Hu, X. and Junping, X., 2020. Decentralized consensus decision-making for cybersecurity protection in multimicrogrid systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *51*(4), pp.2187-2198.

[5] Bonnah, E. and Shiguang, J., 2020. DecChain: A decentralized security approach in Edge Computing based on Blockchain. *Future Generation Computer Systems*, *113*, pp.363-379.

[6] Grechaninov, V., Hulak, H., Hulak, E., Skladannyi, P. and Sokolov, V., 2021. Decentralized Access Demarcation System Construction in Situational Center Network. *Cybersecurity Providing in Information and Telecommunication Systems II 2021*, *3188*(2), pp.197-206.

[7] Bodkhe, U., Mehta, D., Tanwar, S., Bhattacharya, P., Singh, P.K. and Hong, W.C., 2020. A survey on decentralized consensus mechanisms for cyber physical systems. *IEEE Access*, *8*, pp.54371-54401.

[8] Appiah-Kubi, J. and Liu, C.C., 2020. Decentralized intrusion prevention (DIP) against coordinated cyberattacks on distribution automation systems. *IEEE Open Access Journal of Power and Energy*, *7*, pp.389-402.

[9] Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Alazab, M., Bhattacharya, S., Maddikunta, P.K.R. and Gadekallu, T.R., 2022. Federated learning for intrusion detection system: Concepts, challenges and future directions. *Computer Communications*, *195*, pp.346-361.

[10] Campos, E.M., Saura, P.F., González-Vidal, A., Hernández-Ramos, J.L., Bernabe, J.B., Baldini, G. and Skarmeta, A., 2022. Evaluating Federated Learning for intrusion detection in Internet of Things: Review and challenges. *Computer Networks*, *203*, p.108661.

[11] Alazab, M., RM, S.P., Parimala, M., Maddikunta, P.K.R., Gadekallu, T.R. and Pham, Q.V., 2021. Federated learning for cybersecurity: Concepts, challenges, and future directions. *IEEE Transactions on Industrial Informatics*, *18*(5), pp.3501-3509.

[12] Mishra, P., Biancolillo, A., Roger, J.M., Marini, F. and Rutledge, D.N., 2020. New data preprocessing trends based on ensemble of multiple preprocessing techniques. *TrAC Trends in Analytical Chemistry*, *132*, p.116045.

[13] Luengo, J., García-Gil, D., Ramírez-Gallego, S., García, S. and Herrera, F., 2020. Big data preprocessing. *Cham: Springer*.

[14] Fan, C., Chen, M., Wang, X., Wang, J. and Huang, B., 2021. A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. *Frontiers in energy research*, *9*, p.652801.

[15] Benhar, H., Idri, A. and Fernández-Alemán, J.L., 2020. Data preprocessing for heart disease classification: A systematic literature review. *Computer Methods and Programs in Biomedicine*, *195*, p.105635.

[16] Narkhede, M.V., Bartakke, P.P. and Sutaone, M.S., 2022. A review on weight initialization strategies for neural networks. *Artificial intelligence review*, *55*(1), pp.291-322.

[17] Palanisamy, K., Singhania, D. and Yao, A., 2020. Rethinking CNN models for audio classification. *arXiv preprint arXiv:2007.11154*.

[18] Huang, X.S., Perez, F., Ba, J. and Volkovs, M., 2020, November. Improving transformer optimization through better initialization. In *International Conference on Machine Learning* (pp. 4475-4483). PMLR.

A. Mądry, M. Aleksandar, L. Schmidt, & D. Tsipras, "Towards deep learning models resistant to adversarial attacks", 2017. https://doi.org/10.48550/arxiv.1706.06083.

[19] Mammen, P.M., 2021. Federated learning: Opportunities and challenges. *arXiv preprint arXiv:2101.05428*.

[20] Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W. and Gao, Y., 2021. A survey on federated learning. *Knowledge-Based Systems*, *216*, p.106775.

[21] Li, L., Fan, Y., Tse, M. and Lin, K.Y., 2020. A review of applications in federated learning. *Computers & Industrial Engineering*, *149*, p.106854.

[22] Gafni, T., Shlezinger, N., Cohen, K., Eldar, Y.C. and Poor, H.V., 2022. Federated learning: A signal processing perspective. *IEEE Signal Processing Magazine*, *39*(3), pp.14-41.

[23] Jiang, J.C., Kantarci, B., Oktug, S. and Soyata, T., 2020. Federated learning in smart city sensing: Challenges and opportunities. *Sensors*, *20*(21), p.6230.

[24] Nazah, S., Huda, S., Abawajy, J. and Hassan, M.M., 2020. Evolution of dark web threat analysis and detection: A systematic approach. *Ieee Access*, *8*, pp.171796-171819.

[25] Oz, H., Aris, A., Levi, A. and Uluagac, A.S., 2022. A survey on ransomware: Evolution, taxonomy, and defense solutions. *ACM Computing Surveys (CSUR)*, *54*(11s), pp.1-37.

[26] Caviglione, L., Choraś, M., Corona, I., Janicki, A., Mazurczyk, W., Pawlicki, M. and Wasielewska, K., 2020. Tight arms race: Overview of current malware threats and trends in their detection. *IEEE Access*, *9*, pp.5371-5396.

[27] Ramesh, G., Logeshwaran, J. and Aravindarajan, V., 2022. The Performance Evolution of Antivirus Security Systems in Ultra dense Cloud Server Using Intelligent Deep Learning. *BOHR International Journal of Computational Intelligence and Communication Network*, *1*(1), pp.15-19.

[28] Malhotra, P., Singh, Y., Anand, P., Bangotra, D.K., Singh, P.K. and Hong, W.C., 2021. Internet of things: Evolution, concerns and security challenges. *Sensors*, *21*(5), p.1809.

[29] N. Akhtar and A. Mian, "Threat of adversarial attacks on deep learning in computer vision: a survey", IEEE Access, vol. 6, p. 14410-14430, 2018. https://doi.org/10.1109/access.2018.2807385.

[30] Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., & Müller, K. R. (2021). Explaining deep neural networks and beyond: A review of methods and applications. Proceedings of the IEEE, 109(3), 247-278.

[31] Olu-Ajayi, R., Alaka, H., Sulaimon, I., Sunmola, F., & Ajayi, S. (2022). Building energy consumption prediction for residential buildings using deep learning and other machine learning techniques. Journal of Building Engineering, 45, 103406.

[32] Srinidhi, C. L., Ciga, O., & Martel, A. L. (2021). Deep neural network models for computational histopathology: A survey. Medical image analysis, 67, 101813.