International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V74I2P114

Volume 74 Issue 2, 204-214, February 2026

Original Article

Research on a Hybrid Security Model for Distributed Neural Networks

Timur V. Jamgharyan

Department of Information Security and Software Development, National Polytechnic University of Armenia, Yerevan, Armenia.

Corresponding Author : t.jamharyan@polytechnic.am

Received: 22 October 2025 Revised: 12 January 2026 Accepted: 20 January 2026 Published: 14 February 2026

Abstract *- This research proposes a hybrid method for protecting distributed Machine Learning (ML) systems that combines neural network interconnection strengthening with model architecture obfuscation. Experimental results demonstrate that the combined approach maintains the accuracy of the global learning task while outperforming isolated methods (obfuscation-only or strengthening-only). The proposed method significantly enhances the robustness of inter-network communication by preserving the consistency of feature transmission among neural networks. Both obfuscation and hybrid strategies significantly reduce the effectiveness of various model attacks, such as model stealing, model inversion, and membership inference, thereby lowering the fidelity score of surrogate models. The overall evaluation indicates that the proposed approach achieves a well-balanced trade-off between accuracy, confidentiality, and attack resistance within the defined system constraints.*

Keywords *- Dolev-Yao threat model, Machine Learning Model Security, Model-level attack, Model Obfuscation, Hyperparameters.*

1. Introduction

The growing adoption of ML models across various domains has made ML systems themselves a primary target for adversarial attacks. Consequently, the protection of ML models has emerged as a distinct area of research. Existing defense methods can be broadly categorized into three groups.

1.1. Cryptographic Methods

These approaches provide a high level of formal security but are often associated with considerable computational overhead and limited scalability when applied to neural networks. Typical techniques include encryption, homomorphic computations, and differential privacy mechanisms [1-3].

1.2. Protection Methods against Interaction-Level Attacks

To enhance the resilience of distributed systems, secure protocols for feature transmission have been developed [4,5]. Such protocols reduce the risk of data interception or modification during transmission; however, they remain vulnerable to attacks targeting individual model components.

1.3. Model Obfuscation and Architectural Protection Methods

Several studies [6-8] have explored obfuscation techniques, structural redundancy, and parametric masking. These approaches hinder the extraction of architectural and parametric details but often degrade the accuracy of the original learning task.

Various studies point to the growing threat posed by model-based attacks (model stealing, model inversion, membership inference) [9-11]. Consequently, numerous studies have been devoted to addressing this problem within the stated constraints [12-14].

However, the proposed solutions either provide high security at the expense of reduced efficiency [15-17] or focus on specific aspects of the problem [18-21], again creating new links.

A separate challenge that requires a solution is the detection of malware obfuscated using neural networks. The challenge of increasing the stealth of a model's internal architecture while simultaneously enhancing its resilience is becoming increasingly relevant.

1.4. Research Gap

A review of existing work revealed three key unresolved issues:

- Lack of integrated mechanisms.

Modern defenses typically protect either communication channels or model architecture. No widely deployed solution

provides combined protection against attacks at both the communication and model levels.

- Precision instability during obfuscation.

Most obfuscation approaches significantly distort feature propagation, leading to uneven accuracy loss, especially in distributed systems.

- Insufficient resilience to cross-layer attacks.

Attackers increasingly use hybrid attack vectors (e.g., simultaneous model theft and channel fuzzing). Existing mitigation methods mostly operate in isolation and do not provide a unified optimization framework.

Existing methods and solutions based on them address this challenge with a number of limitations. The method proposed in this paper is distinguished by its combination of model obfuscation and strengthening of connections between networks, which allows for simultaneous enhancement of the architecture's stealth and the resilience of a distributed system to attacks.

The scientific novelty of the proposed approach lies in developing a hybrid model that allows for variable changes in system parameters, thereby enabling adaptive model tuning across operating scenarios and improving forecasting accuracy under changing input data.

This approach enables:

- Dynamically accounting for the influence of external factors on system behavior.
- Reducing forecast uncertainty through real-time parameter calibration.
- Ensuring scalability and portability of the model to different data types and environments.
- Reducing the risk of overfitting by combining multiple calibration and optimization methods.

The aim of this research is to develop a method that combines strengthening of connections between neural networks and model obfuscation.

2. Problem Definition

Let there be a set of neural networks $N = \{N_1, N_2, \dots, N_m\}$ united into a distributed system, where N_i denotes an individual neural network in the distributed system that solves a subtask of the global objective. Data exchange between networks is performed over untrusted communication channels.

In particular, the following types of attacks are possible:

2.1. Interaction-Level Attacks

Interception, modification, or forgery of the data transmitted between neural networks (an adversary with capabilities described by the CIA (CIA, confidentiality, integrity, availability) threat model [22].

2.2. Model-Level Attacks

Extraction of the architecture and parameters of individual networks with the aim of reverse engineering, cloning, or circumventing protection mechanisms (an adversary with capabilities described by the Dolev-Yao threat model) [23].

It is required to develop a method $M = \{F, O\}$ (where: M - denotes the proposed method, F - the interaction operator, and O - the obfuscation operator that modifies a network) that provides protection of inter-network interactions and concealment of the internal structure of neural networks while preserving the functional effectiveness of the system.

3. Boundary Conditions

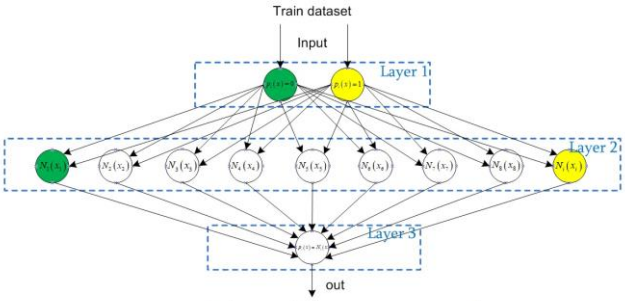
- Obfuscation was applied to only one neural network within the distributed system.
- The experiments were conducted using convolutional, recurrent, capsule, and generative-adversarial networks.
- The obfuscation process must not degrade the robustness of inter-network interactions, while connection strengthening must not reduce the degree of model concealment.
- Each neural network N_i must be transformed into a modified network (N_i'), $N_i' = O(N_i)$that preserves the output distribution (E) within an acceptable error margin $E\left[L\left(N_i(x), N_i'(x)\right)\right] < \varepsilon$, while concealing its architectural and parametric characteristics from external analysis (L - the loss function defined for the primary learning task,ε - the upper bound on the allowable divergence between the output distributions of the original and obfuscated neural networks).

4. Proposed Method

Let each neural network Ni produce an output probability distribution $p_i(x) = N_i(x), p_i(x) \in \Delta^{k-1}$, where Δ^{k-1} is a simplex of dimension k-1, k is the dimension of the output simplex (the number of classes in the classification problem).

Definition of the output matching operator: $F\left(N_i, N_j\right) = soft\,max\left(W_{ij}\phi(p_i)\right)$, where: $\varphi(x)$ is the projection function, W_i is the interaction weight matrix, *softmax* is the normalizing function, and is the latent dimension in the interaction operator.

Optimization is achieved by minimizing the functional $L_{link} = \sum_{i \neq j} D_{KL}\left(F(N_i), F\left(N_J\right)\right)$, where L_{link} is the internetwork matching functional, and D_{KL} is the Kullback–Leibler (KL) divergence [24] measuring the discrepancy in the matching of distributed models (Figure 1).



$$L_{link} = \sum_{i \neq j} D_{KL}\left(F(N_i), F(N_J)\right)$$

Fig. 1 Alignment of neural network outputs

The proposed model alignment approach brings the output probability distributions of different neural networks into a unified representation. This method reduces the system's vulnerability to attacks on individual communication channels. Consequently, it ensures output consistency and enhances robustness against targeted channel-level attacks. The interaction between networks is synchronized iteratively via the F_{match} functional, and obfuscation O is applied to one network selected randomly at each training step. Unlike known approaches, the proposed method optimizes both structural and probabilistic consistency of distributed models in a single functionality.

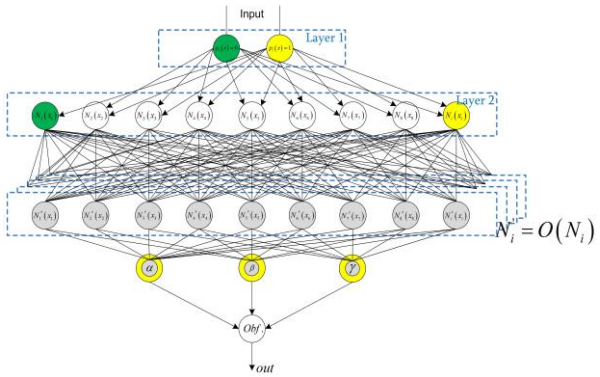
4.1. Model Obfuscation

Each network *Ni* is transformed by an obfuscation operator *O*, which modifies the network's structure and parameters while preserving its functionality.

The obfuscation process was implemented using the method described in research [25]. Formally, $N_i' = O(N_i)$where: $N_i'$is the modified version of the original network. The obfuscation criterion L_{obf} is defined as $L_{obf} = L_{task}(N_i') + \lambda \cdot C(N_i')$, where L_{task} denotes the task-specific loss function, $C(N_i')$ represents the complexity measure (e.g., the number of unique layers or the network depth), and $\lambda$is the balancing coefficient controlling the trade-off between model accuracy and concealment degree. Thus, the operator O minimizes the combined functional L_{obf} , ensuring that task accuracy is preserved while enhancing the concealment of the model's architecture and parameters.

4.2. Joint Optimization

The final goal of the proposed method is the simultaneous optimization of interaction and obfuscation processes. $L_{total} = \alpha \cdot L_{task} + \beta \cdot L_{link} + \gamma \cdot L_{obf}$ task represents the global task error, L_{link} is the inter-network consistency functional, and L_{obf} denotes the obfuscation functional. The coefficients α, β, γ are hyperparameter**s** controlling the balance between accuracy, interaction robustness, and model concealment. Thus, the training of the entire system reduces to finding the set of network parameters that minimizes the overall optimization functional L_{total} (Figure 2).



$$L_{total} = \alpha \cdot L_{task} + \beta \cdot L_{link} + \gamma \cdot L_{obf}$$

Fig. 2 Final optimization functional

Each type of neural network is described by its own analytical expressions, which, by varying the parameters, can improve a particular indicator for that network.

An equation for the functioning of a neural network classification model (1) is given in [26].

$$Y(z) = \varphi\left(\sum_{i=1}^{N_2} \omega_{i1}^{(3)} . \varphi\left(\sum_{j=1}^{N_1} \omega_{ij}^{(2)} . \varphi\left(\sum_{k=1}^{n} \omega_{jk}^{(1)} . Z_k + \Theta_j^{(1)}\right) + \Theta_i^{(2)}\right) + \Theta_1^{(3)}\right) \quad (1)$$

where φ the neural network activation function, $\omega_{ij}^{(k)}$ weights at the input of the neuron of the i[th] layer, $\Theta_i^{(n)}$ output layer offset parameter, and Z_k signals at the input/output of neural network layers.

Analytical expression (1) is necessary for the final evaluation of the overall effectiveness of the proposed model and allows for the output distributions of different neural networks to be unified. This approach reduces the system's vulnerability to attacks on individual communication channels. This ensures output consistency and reduces vulnerability to attacks on individual communication channels. The proposed method enables achieving a balance between efficiency and security. This results in a practical compromise: the system maintains high prediction accuracy while reducing the likelihood of interaction-based attacks and concealing the architectures of individual models.

4.3. Algorithm Environment

Algorithm: Hybrid training with random obfuscation

Input: Dataset D, networks {N_i}, obfuscation operator O, interaction weight matrix W, hyperparams α, β, γ, epochs T

for epoch = 1 to T do

for each mini-batch B ∈ D do

for each node i:

compute y_i = N_i(x; θ_i) # forward
select j ← Uniform({1..n}) # random node to obfuscate
θ_j' = O (θ_j; λ) # apply obfuscation with param λ
compute F_match = Σ_{i,k} KL(softmax (W·proj(y_i)), softmax(W·proj(y_k)))
compute L_task = Σ_i Loss_task (y_i, y_true)
compute L_obf = ObfCriterion (θ_j', θ_j)
L_total = α·L_task + β·F_match + γ·L_obf
update θ ← θ − η ∇_θ L_total
end for
end for
Output: Trained parameters {θ_i}

The training process jointly minimizes the task loss, the inter-network matching functional, and the obfuscation criterion within a single optimization loop. At each iteration, all networks compute their outputs, and one network is randomly selected for structural obfuscation, ensuring stochastic perturbation of its parameters under the constraint of bounded output divergence.

4.4. Description of the Experiment

In a virtual environment based on a High-Performance Cluster (HPC) (Figure 3), the Windows Server 2019 Operating System (OS), with the activated Hyper-V virtualization role [27].

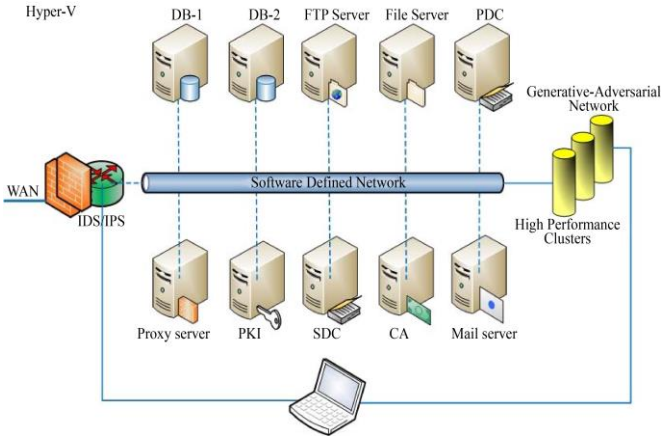


Fig. 3 Software-Defined Network based on an HPC

Virtual environment was configured a software-defined network (SDN), containing the following: primary and secondary domain controllers (PDC, SDC), File Server, FTP server, Proxy server, Public Key Infrastructure (PKI), Certification Authority (CA), Mail server, ML IDS and Snort IDS, Suricata (Snort IDS intentionally had an unmitigated vulnerability *CVE-2022-2068*) and Database servers (DB-1, DB-2). OS Ubuntu v20.04 OS installed a generative adversarial network, connected to the ML IDS, which was also deployed separately. The solutions proposed in [28] were taken into account during deployment.

A total of k instances (k = 4) of virtual infrastructures simulating a distributed security system using ML are deployed. These include the Snort IDS with an ML plug-in, the aforementioned services, and the Kali Linux OS with Metasploit software.

The IDS protects its infrastructure object at each level: level 1 – FTP, level 2 – DB-1, 2, PDC, level 3 – PKI, CA, SDC, etc. Network connectivity is configured between the instances using a Hyper-V virtual adapter. Connectivity between instance elements is configured using Hyper-V private adapters.

The architecture of the infrastructure deployment within the SDN is shown in Figure 4.

The environment shown in Figures 3-4 enables attack emulation and practical evaluation of theoretical results. The study used *Python 3.11*, *scikit-learn 1.5*, *torchvision 0.17*, and secure communication channels between models built using the *cryptography* library version 41.1. The training dataset was *CIC IDS 2017* [29] with embedded malware samples *abc*, *cheeba*, *december_3*, *stasi*, *otario*, *dm*, *v-sign*, *tequila*, *flip* obtained from sources [30-32].

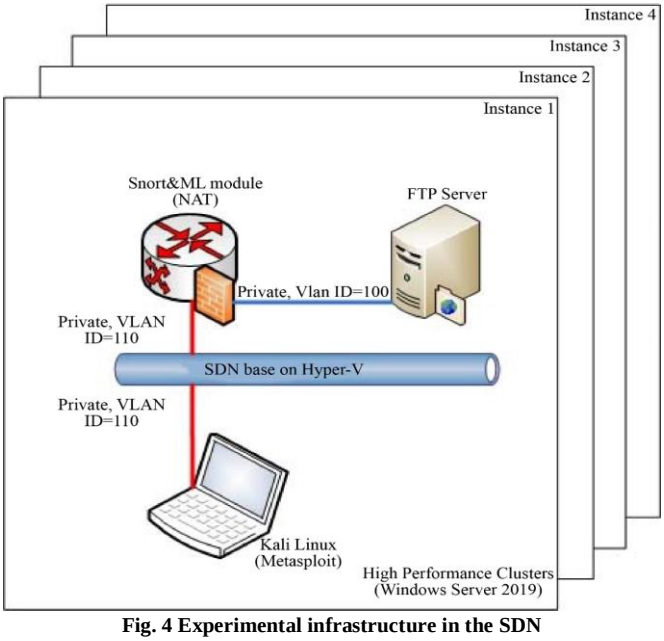


Fig. 4 Experimental infrastructure in the SDN

4.5. Obfuscation was Performed using

Structural permutation of equivalent blocks (permutation of channels/layers with compensating matrices) [33-34], stochastic sparsity [35-36], and parametric masking using low-rank factorization of weights [37-40].

Visualization of the obfuscated *abc*, *cheeba*, *december_3*, *stasi*, *otario*, *dm*, *v-sign*, *tequila*, *flip*, *mimikatz* malware at different training epochs is shown in Figures 5-7.

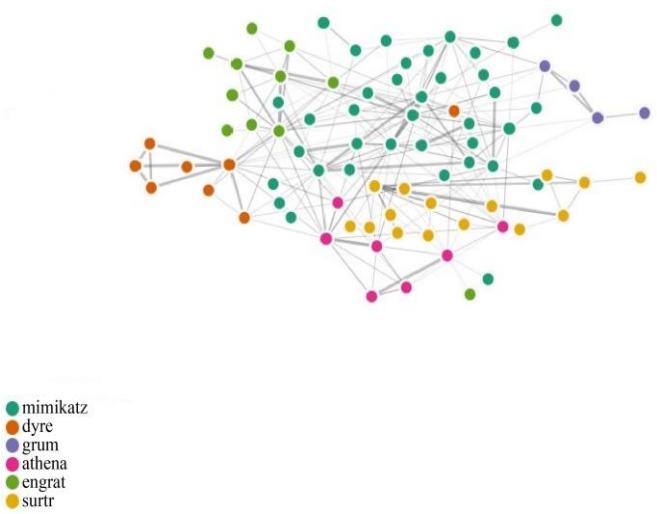


Fig. 5 Visualization of the obfuscated malware (I training epoch)

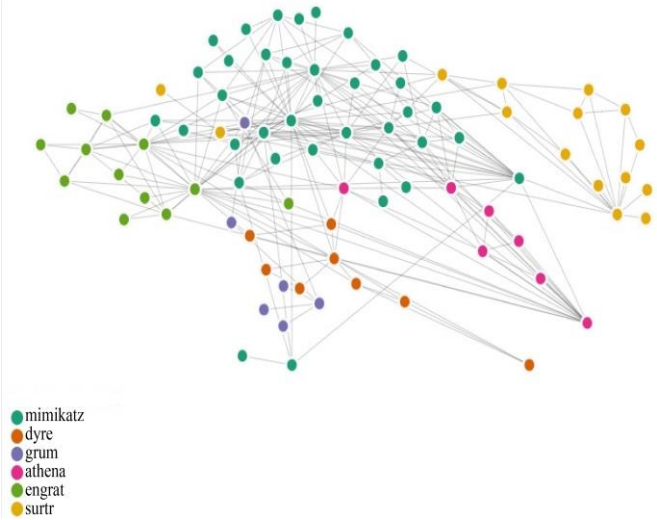


Fig. 6 Visualization of the obfuscated malware (II training epoch)

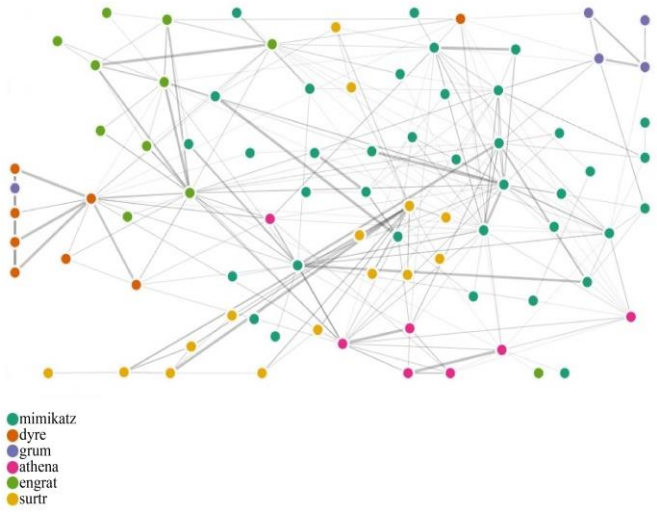


Fig. 7 Visualization of the obfuscated malware (III training epoch)

4.5.1. Attacks Carried Out

Attacks on interactions, packet permutation/drop, and fuzzing using deterministic fuzzers [41, 42].

4.5.2. Attacks on the Model

Model extraction (model stealing), input inversion/reconstruction.

4.6. Evaluation Criteria

- The classification task was evaluated using the metrics *accuracy*, $F_{1macro,}$ *and AUROC*. As the loss function, we used the averaged logarithmic loss (*Log Loss)*.

- Robustness to attacks was evaluated with the metric:

$$Fidelity = E_{x\sim D_{test}}\left[cos\left(N_i'(x), \widehat{N}_i(x)\right)\right],$$

where: *Fidelity* is the score of the surrogate $\widehat{N}$, $E_{x\sim D_{test}}$ denotes the expectation over objects x drawn from the test dataset D_{test}, $N_i'(x)$ the feature vector at the input x for the original model N, and $\widehat{N}_i(x)$the feature vector at the input xxx for the modified (surrogate) model.

- For membership inference attacks used $MIA_{AUC} = AUC(\{s(x_{in})\}, \{s(x_{out})\})$ (MIA, Membership Inference), $s(x_{in})$-is the value of the score function for an object x_{in} from the training data, and $s(x_{out})$is the value of the score function for an object x_{out} from external (out-of-training) data.

- Some hyperparameter values: learning rate η = 1e-3 (0,001), batch size = 128, epochs = 5, obfuscation strength λ ∈ {0.1, 0.3, 0.5}, interaction latent dim d = 64, W init = random normal, softmax = 1.0, coefficients: α = 1.0, β = 0.5, γ = 0.2.

4.7. Final Evaluation Function

$$Q = \alpha_1 \cdot Accuracy \pm \alpha_2 \cdot LogLoss \pm$$
$$\pm\alpha_3 \cdot LinkScore \pm \alpha_4 \cdot ObfScore \pm \alpha_5 \cdot Fidelity \quad (2)$$

(*Q - final evaluation function*). The weights $\alpha_i > 0$ and the operation type (+ or –) are chosen based on the task's priorities (for example, balancing between accuracy and stealth). Attack scenarios were constructed following the approaches described in [43, 44].

5. Research Results

The research results are presented in Tables 1-4 and Figures 8-13. Visualizations of the values of parameters *accuracy*, *precision*, *recall*, and *specificity* for various attacks are presented in Figures 14-16. As shown in Table 1, the hybrid method provides the best balance between accuracy and robustness while maintaining high metric values. Additionally, the hybrid approach maintains higher precision and recall stability even under variable noise conditions.

Table 1. Classification accuracy comparison

Method	Accuracy	Precision	Recall	F1-score
Baseline Model	0.87	0.85	0.83	0.84
Obfuscation	0.85	0.83	0.82	0.82
Connection Strengthening	0.88	0.86	0.84	0.85
Hybrid Method	0.91	0.89	0.87	0.88

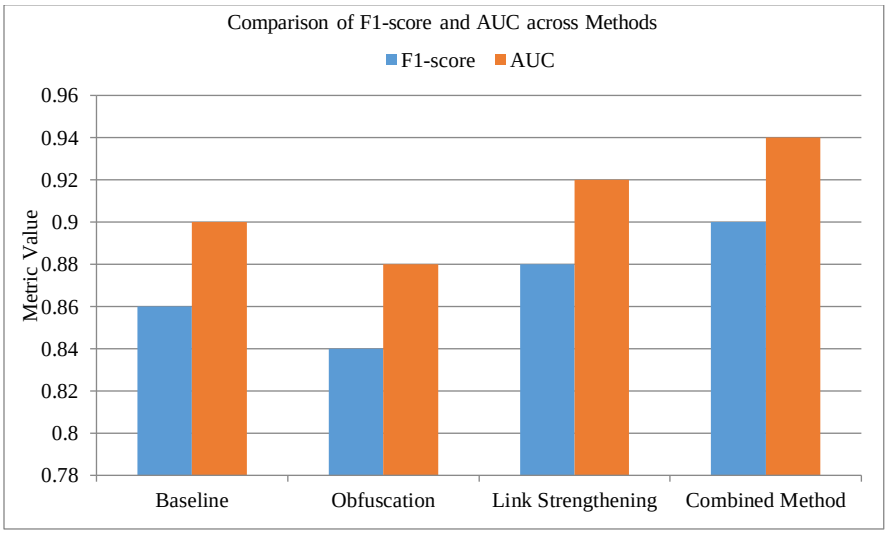


Fig. 8 Comparison results of classification accuracy for different methods

This indicates that the integration of obfuscation with interconnection strengthening improves both generalization and feature consistency across distributed nodes. Specifically, the hybrid configuration achieves an accuracy of 0.91, exceeding the baseline model by 4.6%**,** and improves the *F1-score* to 0.88, which is 7.3% higher than the obfuscation-only setup. Moreover, the precision and recall values remain above 0.87 even under moderate perturbations, demonstrating consistent behavior across all test scenarios.

Such numerical stability suggests that the hybrid framework provides a more uniform error distribution and effectively mitigates local overfitting effects within the distributed learning process. The chart illustrates the *accuracy*, *precision*, *recall*, and *F1-score* values for the baseline model, obfuscation, connection strengthening, and the hybrid method. It can be seen that the hybrid method demonstrates the best overall balance among these metrics.

Table 2. Robustness against interaction attacks

Attack Type	Baseline Model	Obfuscation	Connection Strengthening	Hybrid Method
Packet Reordering	0.72	0.74	0.82	0.86
Channel Noise	0.69	0.71	0.81	0.85
Fuzzing	0.65	0.70	0.80	0.84

As observed from Table 2, the hybrid method consistently achieves the highest robustness scores across all types of interaction-level attacks. Specifically, its resistance to packet reordering reaches 0.86, outperforming the connection-strengthening-only method by 4% and the baseline by 19.4%. Under channel noise, the hybrid approach maintains a robustness of 0.85, indicating effective preservation of inter-network data integrity.

Even during fuzzing attacks, which typically cause the greatest instability, the method maintains 0.84, demonstrating its ability to ensure stable feature transmission and communication reliability under adverse network conditions. The results of packet reordering, channel noise, and fuzzing attacks are shown. The hybrid method demonstrates higher robustness compared to isolated approaches.

Obfuscation by itself reduces the effectiveness of these attacks. The obtained results confirm that the hybrid method provides the highest resistance to model-level attacks.

The reduction of fidelity and similarity metrics demonstrates that the obfuscation mechanism effectively conceals internal parameters while the strengthened links prevent reconstruction of the model's latent space.

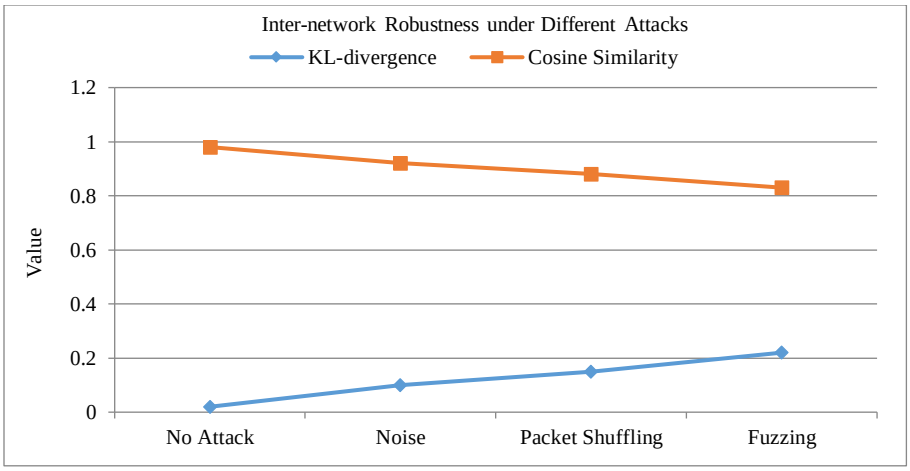


Fig. 9 Results of robustness against interaction attacks (packet reordering, noise, fuzzing)

Table 3. Robustness against model attack

Attack	Metric	Baseline	Obfuscation	Connection Strengthening	Hybrid Method
Model Stealing	Fidelity Score ↓	0.91	0.74	0.88	0.69
Model Inversion	Feature Similarity ↓	0.83	0.66	0.81	0.61
Member-ship Inference	Attacker AUC ↓	0.86	0.72	0.84	0.68

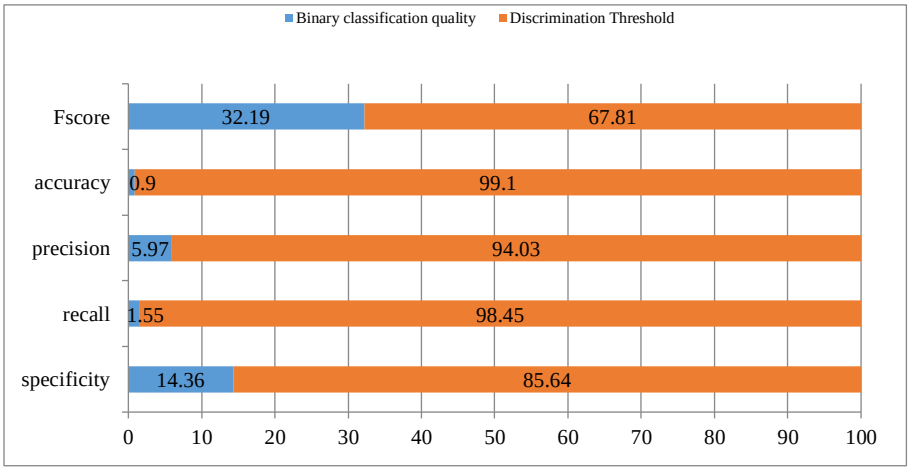


Fig. 10 Visualization of the values of the parameters accuracy, precision, recall, specificity, and F-score (I training epoch)

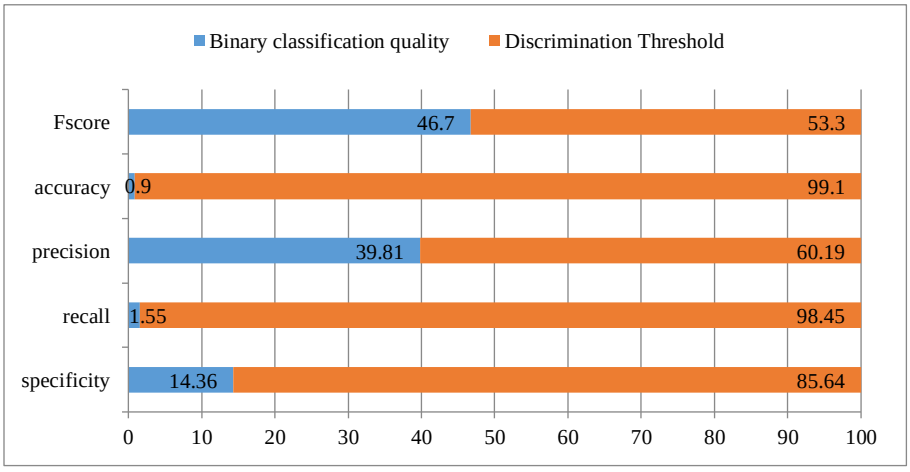


Fig. 11 Visualization of the values of the parameters accuracy, precision, recall, specificity, and F-score (II training epoch)

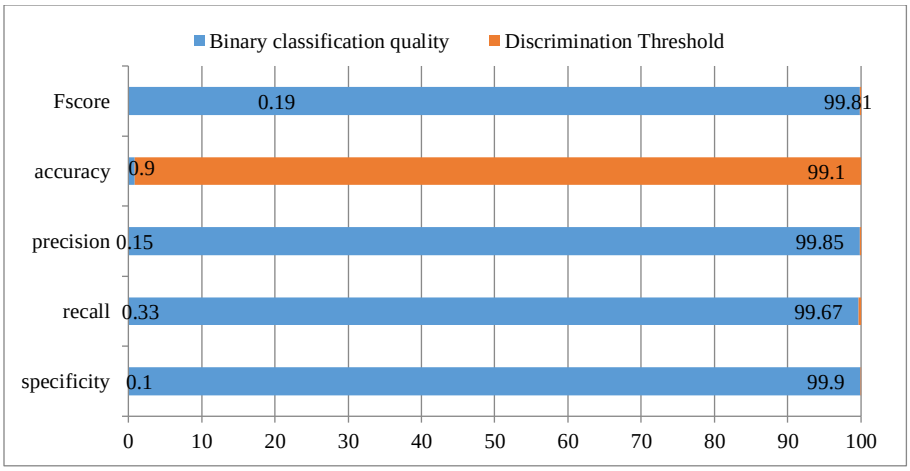


Fig. 12 Visualization of the values of the parameters accuracy, precision, recall, specificity, and F-score (IV training epoch, overfitting)

The model achieves its optimal F-score value at the third training epoch (Figures 10-11). Determining the F-score allows them to set initial values for input parameters when configuring the network infrastructure, minimizing precomputations. At training epochs IV and higher, the model overfitted.

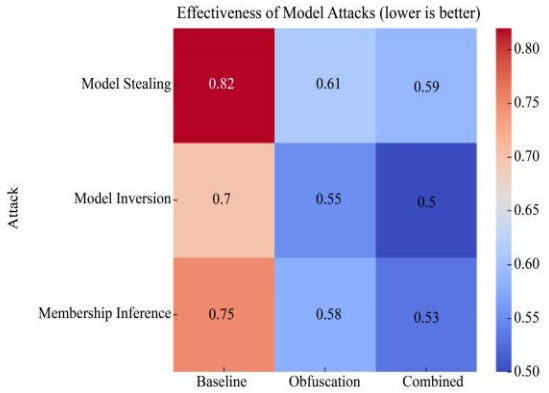


Fig. 13 Shows metrics for model stealing, model inversion and membership inference attacks

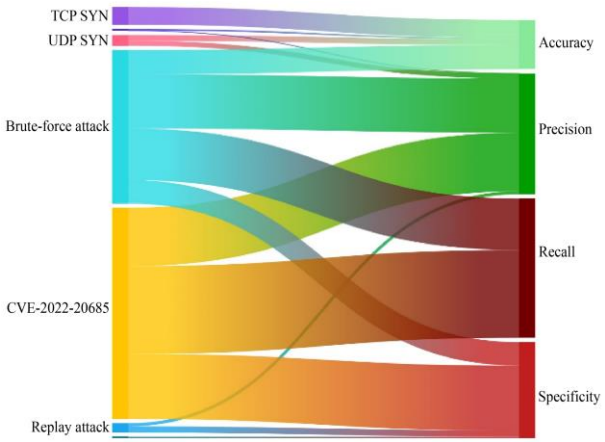


Fig. 14 Visualization of the values of the parameters accuracy, precision, recall, and specificity(I training epoch)

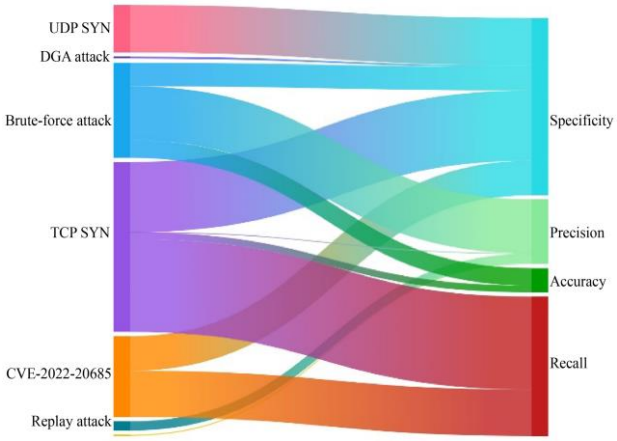


Fig. 15 Visualization of the values of the parameters accuracy, precision, recall, and specificity (II training epoch)

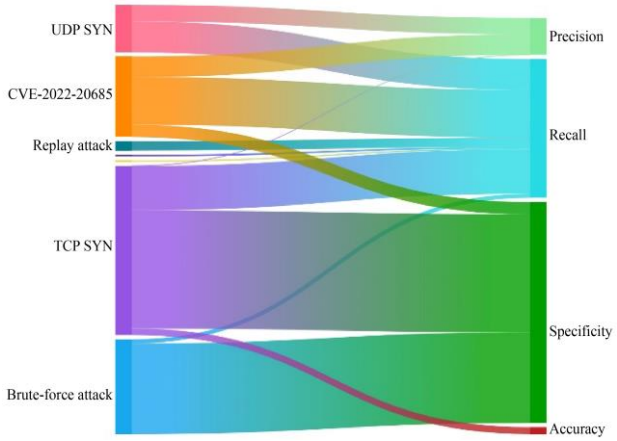


Fig. 16 Visualization of the values of the parameters accuracy, precision, recall, and specificity (III training epoch)

When scaling network infrastructures (both topologically and by supported services), changing the accuracy, precision, recall, and specificity parameters makes it possible to create statistically independent values for training samples and, consequently, increase the reliability of the entire model.

The map shows the combined effectiveness of the three main attacks on the model in baseline configurations. Lower values indicate decreased attacker success, with the hybrid method consistently yielding the lowest results for model theft (0.59), inversion (0.50), and membership inference (0.53). Based on the obtained data, statistical tests were conducted on both the false network infrastructure itself and ML IDS tests. As shown in Table 4, the hybrid model achieves the highest integral score, reflecting a strong balance between accuracy, robustness, and stealth. This balance confirms the efficiency of combining complementary defense mechanisms within a unified optimization process.

Table 4. Integral Evaluation

Method	Accuracy	Attack Robustness	Stealth	Integral Score
Baseline Model	0.87	0.70	0.40	0.65
Obfuscation	0.85	0.72	0.82	0.78
Connection Strengthening	0.88	0.81	0.50	0.80
Hybrid Method	0.91	0.86	0.79	0.88

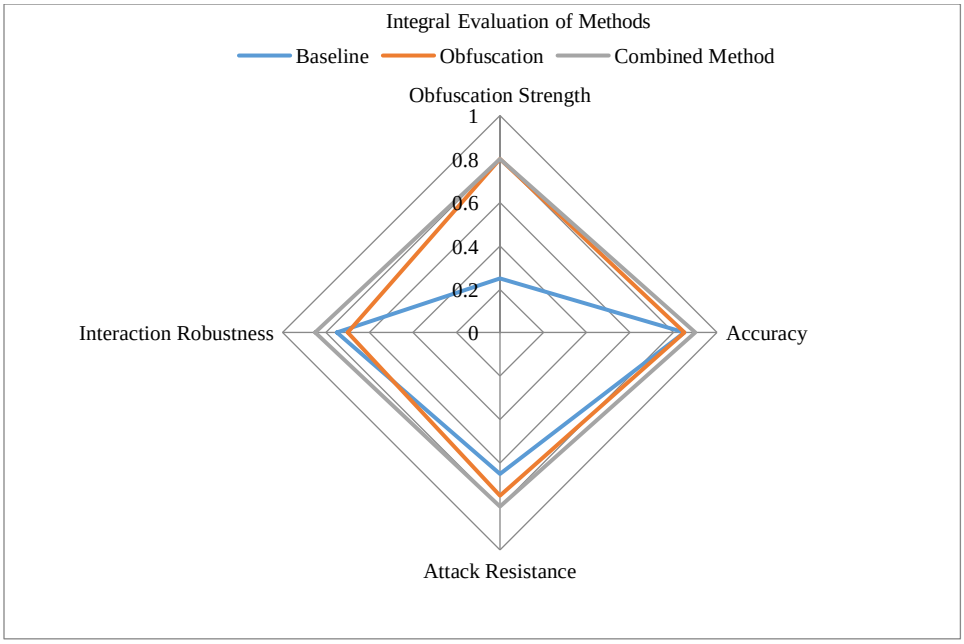


Fig. 17 Integral evaluation of accuracy, robustness, and stealth for different methods

6. Conclusion

This research proposes a method that combines two complementary approaches to protecting distributed systems based on neural networks: strengthening the connections between individual models and obfuscating their structure and parameters. The hybrid approach improves the accuracy of the global learning task while reducing the risk of successful model or channel attacks. Based on the obtained results, it can be concluded that the proposed method increases the resilience of inter-network interaction under various attack scenarios such as noise, packet reordering, and fuzzing.

The hybrid strategy demonstrates a superior balance between accuracy (0.91), robustness (0.86), and stealth (0.79) compared to isolated methods, confirming its effectiveness as an integrated security mechanism. However, these improvements are accompanied by higher computational costs during training and slightly increased inference time, which represents a reasonable trade-off between security and efficiency. Moreover, exploring adaptive weighting of obfuscation and strengthening components could further improve dynamic resilience against evolving attack strategies. The introduction of gradient synchronization mechanisms across distributed nodes may enhance convergence stability and minimize inter-network divergence under adversarial conditions. It is important to note the potential risks of dual-use obfuscation technologies, for example, the possibility of their use to conceal malicious models. Therefore, research into such methods should be accompanied by the development of ethical certification and monitoring mechanisms for their use. Finally, integrating hardware-assisted encryption for secure communication channels will reduce the latency overhead introduced by software-based cryptographic layers. Collectively, these enhancements will enable more scalable, efficient deployment of the proposed hybrid model in practical distributed machine learning systems, ensuring robust protection against both interaction-level and model-level attacks.

Acknowledgments

The author acknowledges the administration of the National Polytechnic University of Armenia for providing the technical facilities necessary for this research.

Special thanks are extended to Mrs. Tsaghik Hovhannisyan, Scientific Secretary of the University, and Mr. Arman Hovhannisyan, lecturer at the Department of Information Security and Software Development.

References

[1] Rezak Aziz et al., "Exploring Homomorphic Encryption and Differential Privacy Techniques towards Secure Federated Learning Paradigm," *Future Internet*, vol. 15 no. 9, pp. 1-25, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Yugeng Liu et al., "ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models," *Proceedings of the 31st USENIX Security Symposium*, pp. 4525-4542, 2022. [Google Scholar] [Publisher Link]

[3] Daryna Oliynyk, Rudolf Mayer, and Andreas Rauber, "I Know What You Trained Last Summer: A Survey on Stealing Machine Learning Models and Defences," *ACM Computing Surveys*, vol. 55 no. 14s, pp. 1-41, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[4] Han Cao et al, "Dslassocov: A Federated Machine Learning Approach Incorporating Covariate Control," *arXiv Preprint*, pp. 1-35, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[5] Alexander Wood, Kayvan Najarian, and Delaram Kahrobaei, "Homomorphic Encryption for Machine Learning in Medicine and Bioinformatics," *ACM Computing Survey*, vol. 53, no. 4, pp. 1-35, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[6] Jun Niu et al., "A Survey on Membership Inference Attacks and Defenses in Machine Learning," *Journal of Information and Intelligence*, vol. 2, no. 5 pp. 404-454, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[7] Zecheng He, Tianwei Zhang, and Ruby B. Lee, "Model Inversion Attacks against Collaborative Inference," *Proceedings of the 35th Annual Computer Security Applications Conference (ACSAC)*, pp. 148-162, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[8] Soumia Zohra El Mestari, Gabriele Lenzini, and Huseyin Demirci, "Preserving Data Privacy in Machine Learning Systems," *Computers & Security*, vol. 137, pp. 1-22, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] Yulian Sun et al., "Obfuscation for Deep Neural Networks against Model Extraction: Attack Taxonomy and Defense Optimization," *Applied Cryptography and Network Security*, pp. 391-414, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[10] Shu Sun et al., "Investigation of Prediction Accuracy, Sensitivity, and Parameter Stability of Large-Scale Propagation Path Loss Models for 5G Wireless Communications," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 2843-2860, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[11] Runze Zhang et al., "FedCVG: A Two-Stage Robust Federated Learning Optimization Algorithm," *Scientific Reports*, vol. 15, no. 1, pp. 1-13, 2025. [Google Scholar] [Publisher Link]

[12] Jingdong Jiang, Yue Zheng, and Chip-Hong Chang, "PUF-Based Edge DNN Model IP Protection with Self-obfuscation and Publicly Verifiable Ownership," *IEEE International Symposium on Circuits and Systems (ISCAS)*, London, United Kingdom, pp. 1-5, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[13] Rodrigo Castillo Camargo et al., "DEFENDIFY: Defense Amplified with Transfer Learning for Obfuscated Malware Framework," *Cybersecurity*, vol. 8, no. 1, pp. 1-23, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[14] Gwonsang Ryu, and Daeseon Choi, "A Hybrid Adversarial Training for Deep Learning Model and Denoising Network Resistant to Adversarial Examples," *Applied Intelligence*, vol. 53, no. 8, pp. 9174-9187, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Mingyi Zhou et al., "ModelObfuscator: Obfuscating Model Information to Protect Deployed ML-Based Systems," *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 1005-1017, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[16] Kacem Khaled et al., "Efficient Defense Against Model Stealing Attacks on Convolutional Neural Networks," *2023 International Conference on Machine Learning and Applications (ICMLA)*, Jacksonville, FL, USA, pp. 45-52, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[17] Kaixiang Zhao et al., "A Survey on Model Extraction Attacks and Defenses for Large Language Models," *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 6227-6236, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[18] Marco Romanelli, Konstantinos Chatzikokolakis, and Catuscia Palamidessi, "Optimal Obfuscation Mechanisms via Machine Learning," *arxiv Preprint*, pp. 1-16, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[19] Rickard Brännvall et al., "Technical Report for the Forgotten-by-Design Project: Targeted Obfuscation for Machine Learning," *arXiv Preprint*, pp. 1-26, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[20] Jingtao Li et al., "NeurObfuscator: A Full-stack Obfuscation Tool to Mitigate Neural Architecture Stealing," *2021 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, Tysons Corner, VA, USA, pp. 1-11, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[21] Mahya Morid Ahmadi et al., "DNN-Alias: Deep Neural Network Protection against Side-Channel Attacks via Layer Balancing," *arXiv Preprint*, pp. 1-9, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[22] William Stallings, *Cryptography and Network Security, Principe's and Practice*, Prentice Hall, 2010. [Publisher Link]

[23] Danny Dolev, and Andrew Chi-Chih Yao, "On the Security of Public Key Protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198-208, 1983. [CrossRef] [Google Scholar] [Publisher Link]

[24] S. Kullback, and R.A. Leibler, "On Information and Sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79-86, 1951. [Google Scholar] [Publisher Link]

[25] Jin-Young Kim, and Sung-Bae Cho, "Obfuscated Malware Detection using Deep Generative Model based on Global/Local Features," *Computers & Security*, vol. 112, pp. 1-20, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[26] Alexander Branitsky, "*Detection of Anomalous Network Connections based on the Hybridization of Computational Intelligence Methods*," Ph.D., Thesis, Saint-Petersburg Institute of Informatics and Automation of the Russian Academy of Sciences, Russian Federation, Saint-Petersburg, 2018. [Publisher Link]

[27] Microsoft Official Website. Windows Operating System Download Page, 2025. [Online]. Available: https://www.microsoft.com/en-us/evalcenter

[28] Mohammed Al-Ambusaidi et al., "RETRACTED ARTICLE: ML-IDS: An Efficient ML-Enabled Intrusion Detection System for Securing IoT Networks and Applications," *Soft Computing*, vol. 28, no. 2, pp. 1765-1784, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[29] The Canadian Institute for Cybersecurity Datasets (CIC-IDS) Website, 2025. [Online]. Available: https://www.unb.ca/cic/datasets/index.html

[30] The Malware Bazaar Website. [Online]. Available: https://bazaar.abuse.ch/

[31] The Malware Database Website. [Online]. Available: http://vxvault.net/ViriList.php

[32] The Malware Download Webpage, 2025. [Online]. Available: https://github.com/vxunderground

[33] Chelsea Finn, Pieter Abbeel, and Sergey Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," *Proceedings of the 34th International Conference on Machine Learning (PMLR)*, vol. 70 pp. 1126-1135, 2017. [Google Scholar] [Publisher Link]

[34] Zechun Liu et al., "ParetoQ: Scaling Laws in Extremely Low-bit LLM Quantization," *arXiv Preprint*, pp. 1-19, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[35] Juan Terven et al., "A Comprehensive Survey of Loss Functions and Metrics in Deep Learning," *Artificial Intelligence Review*, vol. 58, no. 7, pp. 1-172, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[36] Christos Louizos, Max Welling, and Diederik P. Kingma, "Learning Sparse Neural Networks through L_0 Regularization," *arXiv Preprint*, pp. 1-13, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[37] Amey Agrawal et al., "Etalon: Holistic Performance Evaluation Framework for LLM Inference Systems," *arXiv Preprint*, pp. 1-12, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[38] Scott M Lundberg, and Su-In Lee, "A Unified Approach to Interpreting Model Predictions," *Advances in Neural Information Processing Systems*, CA, USA, vol. 30, 2017. [Google Scholar] [Publisher Link]

[39] Daniel Nichols et al., "Performance-Aligned LLMs for Generating Fast Code," *arXiv Preprint*, pp. 1-12, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[40] Boyang Zhang et al., "Lossless Model Compression via Joint Low-Rank Factorization Optimization," *arXiv Preprint*, pp. 1-10, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[41] The Bootfazz Website, 2025. [Online]. Available: https://boofuzz.readthedocs.io/en/stable/#

[42] The Radamsa Project Webpage, 2025. [Online]. Available: https://gitlab.com/akihe/radamsa

[43] Milin Zhang et al., "Adversarial Attacks to Latent Representations of Distributed Neural Networks in Split Computing," *Computer Networks*, vol. 273, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[44] Yuheng Zhang et al., "The Secret Revealer: Generative Model-Inversion Attacks against Deep Neural Networks," *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 253-261, 2020. [Google Scholar] [Publisher Link]