DOI: https://doi.org/10.36910/6775-2524-0560-2026-63-06
UDC 004.8:004.056:004.7
Kudrenko Stanislava, Candidate of Technical Sciences, Associate Professor
https://orcid.org/0000-0002-0759-3908
Nimych Oleksii, PhD-student
https://orcid.org/0000-0003-1759-7088
Makieiev Ihor, PhD-student
https://orcid.org/0009-0009-8679-5652
Alkema Vitalii, PhD-student
https://orcid.org/0009-0000-0009-8237
State University «Kyiv Aviation Institute», Kyiv, Ukraine

A ROBUST CLIENT UPDATE AGGREGATION ALGORITHM FOR COUNTERACTING MODEL POISONING ATTACKS IN FEDERATED LEARNING

Kudrenko S., Nimych O., Makieiev I., Alkema V. A robust client update aggregation algorithm for counteracting model poisoning attacks in federated learning. The article considers the problem of counteracting model poisoning attacks in federated learning systems. It is shown that the client update aggregation stage is security-critical, since compromised clients may transmit intentionally distorted local updates and influence the quality of the global model. A robust client update aggregation algorithm is proposed. The algorithm combines the standard FedAvg weight, which reflects the size of the local client dataset, with an additional reliability coefficient of the corresponding update. To estimate reliability, the normalized Euclidean distance between a client update and the reference update is used. The reference update is defined as the component-wise median of the received client updates. This approach makes it possible to reduce the influence of suspicious or potentially poisoned updates without rigidly excluding them from the aggregation process. The paper presents a formal description of the algorithm, its workflow, an analysis of expected behavior for typical, heterogeneous, and suspicious client updates, and an illustrative numerical example of robust weight calculation. The proposed approach can be used to improve the resilience of machine learning and deep learning systems under cyber threats, especially in tasks related to data protection, information security, and the protection of information infrastructure in critical infrastructure environments.
Keywords: federated learning, deep learning, cybersecurity, information security, cyberattack, cyber threats, data protection, anomaly detection, critical infrastructure.

Кудренко С.О., Німич О.В., Макєєв І.Г., Алькема В.В. Алгоритм стійкої агрегації клієнтських оновлень для протидії атакам отруєння моделі (model poisoning) у федеративному навчанні. У статті розглянуто проблему протидії атакам типу *model poisoning* у системах федеративного навчання. Показано, що етап агрегації клієнтських оновлень є критично важливим з погляду кібербезпеки, оскільки скомпрометовані клієнти можуть передавати навмисно спотворені локальні оновлення та впливати на якість глобальної моделі. Запропоновано алгоритм стійкої агрегації клієнтських оновлень, який поєднує стандартну вагу FedAvg, що враховує розмір локального набору даних, із додатковим коефіцієнтом надійності оновлення. Для оцінювання надійності використовується нормована евклідова відстань між клієнтським оновленням і референтним оновленням, яке визначається як покомпонентна медіана множини отриманих оновлень. Такий підхід дає змогу зменшувати вплив підозрілих або потенційно отруєних оновлень без їх жорсткого виключення з процесу агрегації. Наведено формальний опис алгоритму, схему його роботи, аналіз очікуваної поведінки для типових, гетерогенних і підозрілих клієнтських оновлень, а також ілюстративний числовий приклад розрахунку робастних ваг. Запропонований підхід може бути використаний для підвищення стійкості систем машинного та глибокого навчання в умовах кіберзагроз, зокрема для захисту даних та інформаційної інфраструктури об'єктів критичної інфраструктури.
Ключові слова: федеративне навчання, глибоке навчання, кібербезпека, інформаційна безпека, кібератака, кіберзагрози, захист даних, детекція аномалій, критична інфраструктура.

Introduction

Federated learning has become an important approach for building distributed machine learning systems in environments where raw data cannot be centrally collected. This is especially relevant for critical infrastructure, industrial Internet of Things, edge and fog computing systems, medical information systems, transport networks, and other distributed environments with privacy, communication, and security constraints. In such systems, client nodes keep their local datasets and transmit only model updates to the central server.

However, the decentralized nature of federated learning introduces new cybersecurity risks. Since the global model is formed by aggregating updates received from distributed clients, the reliability of these updates directly affects the quality and security of the final model. If some clients are compromised, they may send intentionally distorted updates that influence the global training process. Such attacks are commonly referred to as model poisoning attacks.

Therefore, the aggregation stage in federated learning should be considered not only as an optimization procedure but also as a security-critical operation. The development of robust aggregation algorithms capable of reducing the influence of suspicious or poisoned client updates is an important task for improving the resilience of federated learning systems.

Problem Statement

At the t-th communication round of federated learning, the central server receives a set of local updates from client nodes:

$$U^t = \{\Delta w_1^t, \Delta w_2^t, \dots, \Delta w_n^t\}, \qquad (1)$$

where Δw_i^t denotes the local update generated by the i-th client after training the current global model w_t on its local data.

Following the classical Federated Averaging scheme proposed by McMahan et al. [1], the next global model is formed as follows:

$$w_{t+1} = w_t + \sum_{i=1}^{n} \alpha_i \, \Delta w_i^t, \qquad (2)$$

where α_i is the standard aggregation weight assigned to the i-th client update. In classical FedAvg, this weight usually reflects the relative size of the local dataset of the corresponding client.

The main vulnerability of this scheme is that the server cannot directly verify the correctness of each local update, since the original client data remain decentralized. Therefore, the set U^t may contain not only legitimate updates but also anomalous or intentionally poisoned updates. If such updates are included in aggregation without additional analysis, they may shift the global model toward incorrect decisions, reduce its accuracy, or introduce hidden malicious behavior.

To reduce this risk, the standard FedAvg weight α_i should not be used alone. It should be supplemented with an adaptive reliability coefficient β_i^t, which reflects the estimated reliability of the corresponding client update in the current communication round. As a result, the final robust aggregation weight γ_i^t is introduced. This coefficient combines the dataset-size-based contribution of the client and the estimated reliability of its update:

$$w_{t+1} = w_t + \sum_{i=1}^{n} \gamma_i^t \, \Delta w_i^t. \qquad (3)$$

Thus, the scientific problem is to develop and justify an interpretable mechanism for transforming the deviation of client updates from the reference update into adaptive aggregation weights. Such weights should reduce the influence of suspicious or potentially poisoned updates while preserving the useful contribution of legitimate client updates:

$$\widehat{U}^t = \{\gamma_1^t \Delta w_1^t, \gamma_2^t \Delta w_2^t, \dots, \gamma_n^t \Delta w_n^t\}. \qquad (4)$$

For legitimate updates, the final weight γ_i^t should preserve their useful contribution to the global model. For suspicious or poisoned updates, this coefficient should reduce their influence. Therefore, the task is to develop a robust client update aggregation algorithm that evaluates the deviation of each update from the typical behavior of the majority of clients and uses this evaluation to form adaptive robust aggregation weights.

Analysis of Recent Research and Publications

The basic principles of federated learning were formulated by McMahan et al. [1], who proposed the Federated Averaging algorithm. In this approach, client nodes train a model locally and send only model updates to the central server, where these updates are aggregated into a global model. This work became the foundation for further research in decentralized and privacy-preserving machine learning.

A comprehensive overview of federated learning, its advantages, limitations, and open problems is presented by Kairouz et al. [2]. The authors emphasize that federated learning combines distributed optimization, communication efficiency, privacy protection, data heterogeneity, and security issues. These factors are especially important in systems where clients differ in data quality, computational resources, and reliability.

The problem of unreliable or malicious participants is closely related to Byzantine-robust distributed learning. Blanchard et al. [3] proposed Byzantine tolerant gradient descent and the Krum aggregation rule, which selects updates that are most consistent with the majority of other updates. This approach demonstrated that simple averaging is insufficient when some clients may behave maliciously.

Yin et al. [4] investigated Byzantine-robust distributed learning and considered robust statistical aggregation rules, including coordinate-wise median and trimmed mean. These methods reduce the influence of extreme or abnormal values during aggregation and are therefore important for protecting distributed learning from faulty or malicious updates.

The specific problem of model poisoning attacks in federated learning was studied by Fang et al. [5]. The authors showed that even Byzantine-robust aggregation methods may be vulnerable to carefully designed malicious local updates. Bagdasaryan et al. [6] investigated backdoor attacks in federated learning and demonstrated that a malicious client can introduce hidden behavior into the global model while preserving its normal performance on ordinary test data.

Further research on protection against local model poisoning attacks is presented by Lu et al. [7], who considered methods for improving the robustness of Byzantine-robust federated learning. Alkhunaizi et al. [8] proposed an approach for suppressing poisoning attacks based on analyzing distances between local client updates. This idea is especially relevant for the present study, since it supports the use of update deviation as a basis for adaptive weighting.

Previous studies related to anomaly detection, malicious node behavior, and compromised distributed environments also form an important background for this work. In particular, works [9–11] address traffic anomaly detection, protection against routing attacks, and prediction of node compromise in edge and fog environments. These studies are relevant because a compromised node in federated learning can affect not only communication processes but also the global machine learning model through poisoned updates.

Unresolved Aspects of the Problem

Despite significant progress in federated learning security, the problem of robust aggregation under model poisoning attacks remains insufficiently solved. The standard FedAvg approach is efficient and simple, but it assumes that client updates are generally reliable. If poisoned updates are included in aggregation, they may directly affect the global model.

Byzantine-robust methods such as Krum, coordinate-wise median, and trimmed mean improve resistance to abnormal updates, but they also have limitations. Krum is based on selecting an update that is closest to the majority of other updates, but it may be vulnerable when malicious updates are carefully adapted to appear similar to legitimate ones. Median and trimmed mean reduce the influence of extreme values, but they do not always account for the overall effect of a suspicious update on the global learning process.

Another unresolved issue is the balance between protection and preservation of useful information. Complete exclusion of suspicious updates may be too rigid, especially in heterogeneous federated learning environments where legitimate client updates can naturally differ from each other. Therefore, it is necessary to use a more flexible approach that does not simply remove suspicious updates but reduces their influence according to the degree of deviation from typical client behavior.

Thus, there is a need for a simple and interpretable robust aggregation algorithm that forms adaptive weights for client updates. Such an algorithm should reduce the influence of potentially poisoned updates while preserving useful information from legitimate clients and without requiring access to local client data.

Purpose of the Article

The purpose of the article is to develop a robust client update aggregation algorithm for counteracting model poisoning attacks in federated learning.

To achieve this purpose, the study analyzes the aggregation stage as a security-critical procedure, characterizes the threat of poisoned client updates, justifies the need for adaptive weighting during aggregation, and proposes a method for calculating reliability coefficients β_i^t to reduce the influence of suspicious updates on the global model.

Proposed Robust Client Update Aggregation Algorithm

Based on the problem formulation given in (1)–(4), the proposed algorithm aims to transform the initial set of client updates U^t into a weighted set $\widehat{U}^t$, in which the influence of suspicious or potentially poisoned updates is reduced before aggregation. Unlike standard FedAvg, the proposed approach does not rely only on the dataset-size-based client weight. Instead, it combines the classical FedAvg weight with an additional reliability coefficient that reflects the deviation of the client update from the typical update behavior in the current communication round.

In classical FedAvg, the aggregation weight of the i-th client is determined by the relative size of its local dataset:

$$\alpha_i = \frac{n_i}{N}, \qquad (5)$$

where n_i is the number of training samples stored by the i-th client, and

$$N = \sum_{i=1}^{n} n_i \qquad (6)$$

is the total number of training samples across all participating clients.

However, under model poisoning attacks, the size of the local dataset does not guarantee the reliability of the corresponding update. A compromised client may have a large value of α_i, but still transmit a poisoned update. Therefore, the proposed algorithm introduces an additional reliability coefficient:

$$\beta_i^t \in (0,1], \qquad (7)$$

which characterizes the reliability of the i-th client update at the t-th communication round.

To calculate this coefficient, the server first determines the reference update for the set U^t defined in (1). The reference update is calculated as the component-wise median of all received client updates:

$$\Delta w_{ref}^t = \text{median}(U^t). \qquad (8)$$

The median is used because it is more resistant to extreme values than the arithmetic mean. Therefore, Δw_{ref}^t can be interpreted as an approximation of the typical update of the majority of clients.

To reduce dependency on the absolute scale of model updates at different stages of training, the deviation of each client update from the reference update is calculated using the normalized Euclidean distance:

$$d_i^t = \frac{\| \Delta w_i^t - \Delta w_{ref}^t \|_2}{\| \Delta w_{ref}^t \|_2 + \varepsilon}, \qquad (9)$$

where $\|\cdot\|_2$ denotes the Euclidean norm, and ε is a small positive constant used to avoid division by zero when the reference update norm approaches zero.

The value d_i^t shows how strongly the update Δw_i^t differs from the typical update behavior of the majority of clients. A small value of d_i^t corresponds to a more reliable update, while a large value indicates a potentially anomalous or poisoned update.

The reliability coefficient is then defined as a robust penalty function of the normalized deviation:

$$\beta_i^t = \frac{1}{1 + d_i^t}. \qquad (10)$$

The form of the reliability coefficient in (10) is selected as a simple and interpretable robust penalty function. It is not claimed to be globally optimal for all possible poisoning strategies. Instead, it is used because it satisfies the key requirements of robust aggregation in the considered setting. First, the coefficient is bounded in the interval $0 < \rho_i^t \leq 1$, so it cannot increase the influence of any client update above its original FedAvg-based contribution. Second, it is a monotonically decreasing function of the normalized deviation d_i^t, which means that updates with larger deviation from the reference update receive lower reliability. Third, the function implements soft down-weighting rather than hard exclusion, which is important in non-IID federated learning environments where legitimate client updates may naturally differ from each other.

From a practical point of view, the use of (10) makes it possible to control the influence of suspicious updates without introducing a rigid rejection threshold. When $d_i^t = 0$, the reliability coefficient reaches its maximum value, and the corresponding update is not penalized. As d_i^t increases, the reliability coefficient decreases smoothly, which limits the influence of strongly deviating updates. Therefore, the proposed coefficient provides a stable transformation of the deviation score into an aggregation weight correction.

If stronger or weaker penalization is required in a practical implementation, the same function can be extended by introducing a sensitivity parameter $\lambda > 0$, for example in the form $\rho_i^t = 1/(1 + \lambda d_i^t)$. A

smaller value of λprovides softer penalization and is more suitable for highly heterogeneous non-IID data, whereas a larger value of λsuppresses strongly deviating updates more aggressively. In the present study, $\lambda = 1$is used as a neutral baseline value in order to keep the algorithm lightweight and avoid introducing an additional hyperparameter.

If the client update is close to the reference update, the normalized deviation is small and $\rho_i^t$is close to 1. In this case, the update almost preserves its original FedAvg-based contribution. If the update significantly deviates from the reference update, ρ_i^tdecreases and the influence of this update on the global model is reduced. Thus, the proposed coefficient acts as a smooth reliability-based correction of the standard aggregation weight.

The final robust aggregation weight γ_i^t, introduced conceptually in (3)–(4), combines the standard FedAvg weight $\alpha_i$and the reliability coefficient β_i^t:

$$\gamma_i^t = \frac{\alpha_i \beta_i^t}{\sum_{j=1}^{n} \alpha_j \, \beta_j^t}. \qquad (11)$$

This normalization is also important for stability, because it prevents the total contribution of all client updates from changing after reliability correction. As a result, the proposed method redistributes the aggregation weights among clients according to their estimated reliability, but preserves the general weighted-averaging structure of FedAvg

$$\sum_{i=1}^{n} \gamma_i^t = 1. \qquad (12)$$

As a result, the robust aggregation rule has the following form:

$$w_{t+1} = w_t + \sum_{i=1}^{n} \gamma_i^t \, \Delta w_i^t. \qquad (13)$$

Thus, the proposed algorithm does not replace the standard FedAvg weighting mechanism.

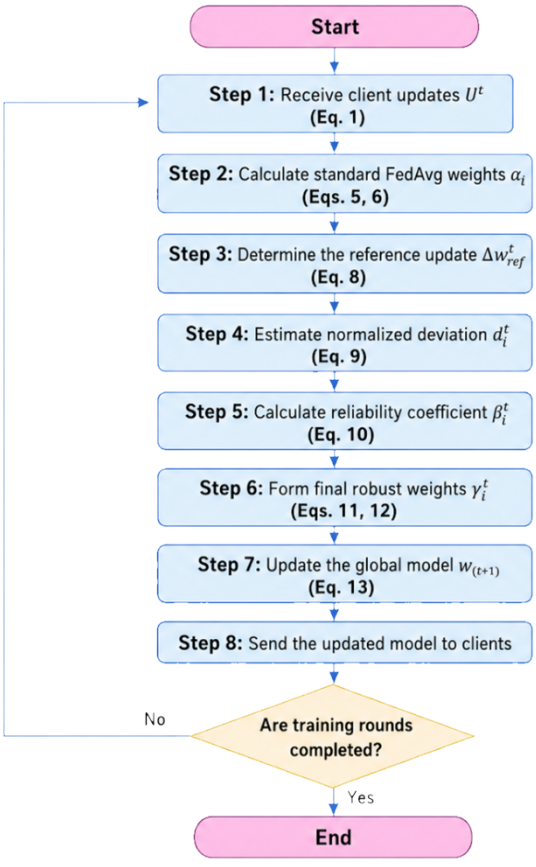

Fig. 1. General workflow of the proposed robust client update aggregation algorithm

It preserves the basic weight α_i, which reflects the amount of local data, and modifies it by the reliability coefficient β_i^t, which reflects the abnormality of the corresponding update. This makes it possible to reduce the influence of poisoned updates while preserving the useful contribution of legitimate clients.

The general workflow of the proposed robust aggregation algorithm is shown in Fig. 1. The diagram provides a high-level representation of the server-side aggregation procedure during one communication round. In contrast to Algorithm 1, which summarizes the main computational steps,

Figure illustrates the overall logic of the process, including the receipt of client updates, reliability-based weight correction, global model update, and transition to the next communication round.

The detailed sequence of operations corresponding to Fig. 1 is described below. The algorithm is executed by the central server at each communication round of federated learning and uses the mathematical relations introduced in (1)–(13). Its main idea is to preserve the classical FedAvg logic through the dataset-size-based weight α_i, while additionally reducing the influence of suspicious updates through the reliability coefficient β_i^t. To avoid scale dependency, the deviation of each update is evaluated using the normalized Euclidean distance introduced in (9).

Algorithm 1. Robust client update aggregation under model poisoning attacks

Input: current global model w_t; set of client updates U^t; local dataset sizes of participating clients; small positive constant ε.

Output: updated global model w_{t+1}.

Step 1. Calculation of FedAvg weights. The server calculates the standard dataset-size-based FedAvg weights $\alpha_i$according to (5)–(6).

Step 2. Determination of the reference update. The reference update $\Delta w_{ref}^t$is calculated according to (8) as the component-wise median of all received client updates.

Step 3. Estimation of normalized deviation. For each client update, the normalized deviation $d_i^t$from the reference update is calculated according to (9). Normalization by $\| \Delta w_{ref}^t \|_2 + \varepsilon$reduces dependence on the absolute scale of model updates.

Step 4. Calculation of the reliability coefficient. For each client update, the reliability coefficient $\beta_i^t$is calculated according to (10). Smaller deviations correspond to higher reliability, whereas strongly deviating updates receive lower reliability values.

Step 5. Formation of robust aggregation weights. The standard FedAvg weight $\alpha_i$is combined with the reliability coefficient β_i^t, and the result is normalized according to (11)–(12). The obtained coefficient $\gamma_i^t$is used as the final robust aggregation weight.

Step 6. Global model update. The global model is updated according to the robust aggregation rule given in (13). As a result, updates with higher estimated reliability preserve a larger contribution, while suspicious or potentially poisoned updates have reduced influence.

The proposed algorithm applies soft down-weighting rather than direct exclusion of suspicious updates. This is important for heterogeneous federated learning environments, where legitimate client updates may naturally differ from each other. The use of normalized deviation makes the reliability estimation less dependent on the absolute scale of model updates at different training stages.

Expected Algorithmic Behavior under Different Client Regimes

The expected behavior of the proposed robust aggregation mechanism can be analyzed using the mathematical properties of (9)–(11). Since the final aggregation weight γ_i^tdepends on both the standard FedAvg weight $\alpha_i$and the reliability coefficient β_i^t, the influence of each client update is adjusted according to its deviation from the reference update of the current communication round. This makes it possible to distinguish three typical client behavior regimes.

Regime 1. Typical client updates $\left(d_i^t \to 0 \right)$.

If a client update is close to the reference update, its normalized deviation d_i^tremains small. According to (10), the reliability coefficient β_i^tapproaches 1. In this case, the final robust aggregation weight γ_i^tremains close to the standard FedAvg weight α_i. Therefore, the useful contribution of a legitimate client is preserved, and the proposed algorithm behaves similarly to classical FedAvg for updates that correspond to the typical behavior of the majority of clients.

Regime 2. Heterogeneous but legitimate client updates $\left(d_i^t\right.$is moderate).

In non-IID federated learning environments, legitimate clients may naturally generate updates that differ from the reference update because their local data distributions are not identical. In this case, the normalized deviation d_i^ttakes a moderate value. The reliability coefficient $\beta_i^t = 1/(1 + d_i^t)$decreases

smoothly, which leads to a proportional reduction of the final aggregation weight γ_i^t. Thus, the update is not removed from the aggregation process, but its influence is bounded. This behavior is important because it allows the algorithm to preserve useful information from heterogeneous clients while reducing the risk of excessive influence from strongly divergent updates.

Regime 3. Suspicious or poisoned client updates $\left(d_i^t \text{is large}\right)$.

If a compromised client submits a poisoned update, for example through additive noise, sign-flipping, or another distortion strategy, the update may significantly deviate from the reference update. In this case, the normalized deviation d_i^tincreases, and the reliability coefficient β_i^tdecreases according to (10). As a result, the final aggregation weight γ_i^tassigned to this update is reduced. Therefore, the proposed algorithm limits the ability of suspicious updates to shift the global model w_{t+1}in an undesirable direction.

The described behavior shows that the proposed algorithm implements a soft weighting strategy rather than rigid filtering. Typical updates preserve their influence, moderately divergent updates are partially down-weighted, and strongly anomalous updates receive a significantly lower contribution to the global aggregation. This is consistent with the requirements of federated learning, where the server should improve robustness against model poisoning attacks without direct access to local client data.

The proposed mechanism assumes that the majority of participating clients in a communication round behave correctly. If the majority of updates are poisoned or if malicious clients generate updates that closely imitate the reference behavior, additional detection mechanisms, such as cosine similarity, historical reputation, or validation-based checks, may be required. However, within the considered setting, the proposed normalized deviation-based weighting provides a simple and interpretable way to reduce the influence of anomalous client updates.

Therefore, the proposed reliability coefficient should be interpreted as a robust heuristic weighting mechanism rather than as a universally optimal estimator. Its stability follows from its boundedness, monotonicity, normalization within the final aggregation weights, and reliance on the median-based reference update, which is less sensitive to extreme values than the arithmetic mean

The expected behavior of the proposed weighting mechanism is summarized in Table 1. It shows how the normalized deviation d_i^taffects the reliability coefficient $\beta_i^t$and, consequently, the final robust aggregation weight γ_i^t.

Table 1. Expected behavior of the proposed robust weighting mechanism

Client Update Characteristic	Deviation d_i^t	Reliability Coefficient β_i^t	Impact on Final Weight γ_i^t	Expected Effect
Close to the reference update	Low $\left(d_i^t \to 0\right)$	Low $\left(d_i^t \to 0\right)$	$\gamma_i^t \approx \alpha_i$	Preservation of the useful contribution of a legitimate client
Moderately divergent update	Moderate	Moderate	Slightly reduced	Balancing robustness with data heterogeneity in non-IID environments
Strongly divergent or suspicious update	High $\left(d_i^t \gg 1\right)$	Low $\left(\beta_i^t \to 0\right)$	Significantly reduced	Limitation of the influence of potentially poisoned updates

As shown in Table 1, the proposed algorithm does not apply rigid exclusion of client updates. Instead, it provides gradual reduction of their influence depending on the normalized deviation from the reference update. This allows the aggregation procedure to preserve useful contributions from legitimate and heterogeneous clients while limiting the impact of suspicious or potentially poisoned updates.

To clarify the position of the proposed algorithm among existing aggregation approaches, it is useful to compare it with several commonly used robust aggregation methods in federated learning. Standard FedAvg provides efficient aggregation but does not include a mechanism for reducing the influence of poisoned updates. Median, trimmed mean, and Krum-based approaches improve robustness, but they may either ignore the dataset-size contribution of clients or apply relatively rigid filtering rules. In contrast, the

proposed algorithm preserves the FedAvg weighting logic and supplements it with an adaptive reliability coefficient.

Table 2. Comparison of aggregation methods in federated learning

Aggregation method	Main principle	Limitation under model poisoning	Advantage of the proposed algorithm
FedAvg	Weighted averaging based on local dataset size	A compromised client with a large dataset may strongly influence the global model	The dataset-size weight is additionally corrected by the reliability coefficient
Coordinate-wise median	Uses median values of update components	May ignore the relative importance of clients with different dataset sizes	The proposed method preserves FedAvg weighting while reducing suspicious influence
Trimmed mean	Removes extreme update values before averaging	Rigid filtering may remove useful heterogeneous updates	The proposed method reduces influence gradually instead of direct exclusion
Krum	Selects the update closest to the majority of updates	May be vulnerable to adaptive poisoning strategies	The proposed method uses soft weighting of all updates based on normalized deviation
Proposed algorithm	Combines FedAvg weight $\alpha_i$with reliability coefficient β_i^t	Assumes that the majority of clients are not compromised	Provides simple, interpretable, and scale-normalized robust aggregation

As shown in Table 2, the proposed algorithm combines the advantages of FedAvg and robust aggregation methods. It preserves the dataset-size-based weighting principle of FedAvg, but additionally introduces a reliability-based correction of each client update. Therefore, suspicious updates are not excluded directly, but their influence on the global model is reduced according to their normalized deviation from the reference update.

The proposed algorithm should be considered as a lightweight and interpretable robust aggregation mechanism. Its main limitation is the assumption that the majority of client updates in a communication round represent legitimate behavior. More complex coordinated attacks may require additional verification mechanisms, such as historical client reputation, validation-based checks, or cosine-similarity-based direction control**.**

Numerical Example of Robust Weight Calculation

To illustrate the operation of the proposed aggregation mechanism, consider a simplified communication round with three illustrative synthetic client updates. Two updates are assumed to be close to each other and represent typical client behavior, while the third update is intentionally distorted and imitates a poisoned update.

Let the client updates be defined as follows:

$$\Delta w_1^t = (0.10,\ 0.20),$$
$$\Delta w_2^t = (0.12,\ 0.18),$$
$$\Delta w_3^t = (2.00,\ -2.00).$$

For simplicity, assume that all clients have equal local dataset sizes. Therefore, their standard FedAvg weights are

$$\alpha_1 = \alpha_2 = \alpha_3 = \frac{1}{3}.$$

According to (8), the reference update is calculated as the component-wise median of the received updates:

$$\Delta w_{ref}^t = median(U^t) = (0.12,\ 0.18).$$

Then, according to (9), the normalized deviation of each update from the reference update is calculated using the L_2norm. The reliability coefficient is then determined by (10), and the final robust aggregation weight is obtained according to (11). The results are presented in Table 3.

Table 3. Example of robust aggregation weight calculation

Client	Client update Δw_i^t	FedAvg weight α_i	Normalized deviation d_i^t	Reliability coefficient β_i^t	Final weight γ_i^t
C_1	(0.10, 0.20)	0.333	0.131	0.884	0.452
C_2	(0.12, 0.18)	0.333	0.000	1.000	0.512
C_3	(2.00, -2.00)	0.333	13.311	0.070	0.036

As shown in Table 3, in the standard FedAvg scheme all three clients would have the same contribution to the global model because their dataset-size-based weights are equal. However, the third update strongly deviates from the reference update. As a result, its reliability coefficient decreases to 0.070, and its final aggregation weight γ_3^tdecreases from $0.333$to 0.036..

This example demonstrates the main effect of the proposed algorithm: a suspicious update is not directly removed from the aggregation process, but its influence on the global model is significantly reduced. At the same time, the updates that are close to the reference behavior preserve most of their contribution. Therefore, the proposed robust weighting mechanism provides a soft and interpretable way to limit the impact of potentially poisoned client updates.

Conclusions.

This paper proposes a robust client update aggregation algorithm for counteracting model poisoning attacks in federated learning. The developed approach focuses on the aggregation stage, where malicious or compromised clients may influence the global model by transmitting distorted local updates.

The main feature of the proposed algorithm is the combination of the standard FedAvg weight α_i, which reflects the relative size of the client's local dataset, with the reliability coefficient β_i^t, which reflects the degree of deviation of the client update from the reference update in the current communication round. As a result, the final robust aggregation weight γ_i^ttakes into account both the amount of local data and the estimated reliability of the corresponding update.

The reliability coefficient is justified as a bounded and monotonically decreasing penalty function of the normalized deviation, which provides stable soft down-weighting of suspicious updates without their direct exclusion from aggregation.

To estimate the reliability of client updates, the algorithm uses the component-wise median as the reference update and calculates the normalized Euclidean distance between each client update and this reference value. The use of normalized deviation reduces dependency on the absolute scale of model updates, which is important at different stages of training, when update magnitudes may significantly change.

The proposed algorithm does not require access to local client datasets and therefore preserves the basic principles of federated learning. Suspicious updates are not directly removed from the aggregation process; instead, their influence is adaptively reduced according to the reliability coefficient. This makes the algorithm suitable for heterogeneous federated learning environments, where legitimate client updates may naturally differ from each other.

The expected behavior of the algorithm shows that typical client updates preserve their contribution to the global model, moderately divergent updates are softly down-weighted, and strongly anomalous or potentially poisoned updates receive significantly lower aggregation weights. The illustrative numerical example confirms that a strongly deviating update can have its final aggregation weight substantially reduced compared with standard FedAvg.

Thus, the proposed algorithm provides a simple, interpretable, and lightweight mechanism for increasing the robustness of federated learning against model poisoning attacks. Further research should focus on experimental validation of the proposed approach using benchmark federated learning datasets, comparison with FedAvg, Krum, coordinate-wise median, and trimmed mean aggregation methods, and extension of the reliability assessment mechanism by incorporating cosine similarity, historical client behavior, or validation-based checks.

References

1. McMahan, H. B., Moore, E., Ramage, D., Hampson, S., & Agüera y Arcas, B. (2017). Communication-efficient learning of deep networks from decentralized data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 54, 1273–1282.
2. Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning, 14*(1–2), 1–210. https://doi.org/10.1561/2200000083
3. Blanchard, P., El Mhamdi, E. M., Guerraoui, R., & Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in Neural Information Processing Systems, 30*, 119–129.
4. Yin, D., Chen, Y., Ramchandran, K., & Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. *Proceedings of the 35th International Conference on Machine Learning*, 80, 5650–5659.
5. Fang, M., Cao, X., Jia, J., & Gong, N. Z. (2020). Local model poisoning attacks to Byzantine-robust federated learning. *Proceedings of the 29th USENIX Security Symposium*, 1605–1622. https://doi.org/10.5555/3489212.3489304
6. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., & Shmatikov, V. (2020). How to backdoor federated learning. *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 108, 2938–2948.
7. Lu, S., Li, R., Chen, X., & Ma, Y. (2022). Defense against local model poisoning attacks to Byzantine-robust federated learning. *Frontiers of Computer Science, 16*, 166337. https://doi.org/10.1007/s11704-021-1067-4
8. Alkhunaizi, N., Kamzolov, D., Takáč, M., & Nandakumar, K. (2022). Suppressing poisoning attacks on federated learning for medical imaging. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022* (Lecture Notes in Computer Science, pp. 673–683). Springer. https://doi.org/10.1007/978-3-031-16452-1_64
9. Stolyar, A., Makieiev, I., & Kudrenko, S. (2024). ReliefF functional selection for traffic anomalies detection. *Problems of Informatization and Management, 4*(80), 64–70. https://doi.org/10.18372/2073-4751.80.19779
10. Stolyar, A., & Kudrenko, S. (2025). Security mechanisms for AODV and E-AODV protocols against black hole and gray tunnel attacks. *Problems of Informatization and Management, 1*(81), 119–125. https://doi.org/10.18372/2073-4751.81.20137
11. Kudrenko, S., Nimych, O., & Makieiev, I. (2025). Method for predicting node compromise in edge and fog environments for critical infrastructure. *Ukrainian Information Security Research Journal, 27*(2), 87–93. https://doi.org/10.18372/2410-7840.27.21183

Історія статті:
Отримано: 01.04.2026 Доопрацьовано: 22.05.2026 Прийнято до друку: 23.05.2026 Опубліковано: 29.05.2026