Sciences Methods and Technologies International Journal (SciMeTech)

ISSN: 3085-5284, journal homepage: www.scimetech.com

A Lightweight Blockchain-based Framework for Ensuring Training Data Integrity in Machine Learning Pipelines

Achraf AIT BENAALI[1], Anas ABOU EL KALAM[1] and Wissam ABBASS[1]

[1]Cadi Ayyad University, National School of Applied Sciences, Research Laboratory for Smart and Sustainable Technologies, Marrakesh, Morocco

How to cite:

Achraf AIT BENAALI, Anas ABOU EL KALAM and Wissam ABBASS, "A Lightweight Blockchain-based Framework for Ensuring Training Data Integrity in Machine Learning Pipelines", Sciences Methods and Technologies International Journal (SciMeTech), Vol 2, Issue 2, p 170-176

Keywords:

Machine Learning Security
Blockchain
Data Integrity
Data Poisoning
Dataset Verification
Cybersecurity

Abstract

Machine learning (ML) is based on a core concept in artificial intelligence (AI) which is data training that makes ML models understand patterns and relationships. Since ML systems heavily rely on the integrity of training data, it's essential to protects the datasets from tampering and data poisoning attacks, which can significantly degrade model performance. In this paper, we propose a lightweight and practical framework that leverages blockchain technology to ensure the integrity of training datasets. The proposed approach computes a cryptographic hash of the dataset and stores it on a blockchain-like structure, enabling verification prior to model training. We implement a prototype using Python and evaluate the system under normal and tampered data scenarios. Experimental results demonstrate that the proposed method effectively detects dataset modifications with negligible overhead. This work provides a simple yet effective solution for enhancing trust in ML pipelines.

I. Introduction

Machine learning has become core component of modern intelligent systems, with application ranging from healthcare to cybersecurity. The performance of ML models depends strongly on the quality and integrity of the training data so that they can produce a highly accurate output. These data are often stored in many different environments such as distributed or cloud environments. This makes the data vulnerable to poisoning attacks via unauthorized modifications, which can cause big problems in areas that rely on critical data. Data poisoning attacks represents a really significant threat to the performance of ML models because they aim to manipulate the training dataset to influence model behavior, often without being easily detectable. However, a minor modification in the dataset can lead to significant degradation in model accuracy and reliability. For this reason, it's crucial to protect this data from such danger which makes ensuring data integrity before training essential.

Blockchain technology as a decentralized, distributed, and immutable digital ledger that securely records transactions across a network of computers, ensure data integrity without intermediaries and offers a promising solution for secure data verification. In this paper, we propose a lightweight framework that integrates blockchain-based verification into the ML training pipeline. Unlike existing complex solutions, our approach is simple, practical, and easy to implement. The main contributions of this work are threefold. First, we propose a lightweight framework for verifying the integrity of training datasets using blockchain principles. Second, we present a simple and practical integration mechanism that allows this verification process to be seamlessly incorporated into existing machine learning pipelines. Finally, we provide experimental validation of the approach by evaluating its effectiveness on both tampered and untampered datasets, demonstrating its ability to detect modifications with minimal overhead.

II. Related Work

Recent studies have highlighted the vulnerability of machine learning systems to data poisoning attacks, where adversaries manipulate training data to degrade model performance. Since blockchain technology has been increasingly explored as a means to ensure data integrity and trust in distributed environments, it represents a great way to ensure security of ML training data. For example, (Kim, 2025) proposed a blockchain-based defense mechanism that enhances the resilience of AI models against data poisoning attacks [1]. Similarly, in the paper of (Venkatesan and Rahayu, 2024), the integration of machine learning techniques within blockchain systems has been investigated to improve security, particularly through hybrid consensus mechanisms capable of detecting anomalies and preventing cyber-attacks [2].

The combination of blockchain, federated learning and Internet of Things have become really popular lately. Researchers have suggested using blockchain to make Internet of Things systems safer. As mentioned by (Usama et al., 2026), this is done by making sure that only the right people can access the system that data is shared in a trusted way and that bad people cannot use intelligence to attack the system [3]. Federated learning is also helpful because it lets people work together on a project without having to share all their data. However, it is still possible for someone to mess with the data on purpose. With all these new ideas such as the one of (Wankhede and Patel, 2025), the existing ways of doing things are often too complicated and use up too many resources [4]. On the other hand, this project is about creating a simple way to make sure that the data used for training is the same prepared data and remains unchanged.

III. Proposed Method

a. System Overview

The proposed system ensures that the dataset used for the training of an ML model has not been modified by verifying its integrity through a blockchain-based mechanism. This is done via computing a cryptographic hash code of the training dataset and store it in a blockchain ledger that guaranties that the hash code remains unchangeable. When it's time to use the dataset for the training of the ML model, the recomputing of its hash is essential using the same cryptographic hash function. Lastly, the comparison between this last hash code and the previous one tells us whether the data are twisted or not. As a result, using this system makes the training data highly protected from tampering and poisoning attacks.

b. System Architecture

The system consists of five essential components that are dataset, hashing module, blockchain storage (simulated), verification module and ML model. The process begins from the hashing module that calculates a hash code and store it in the blockchain ledger. When we have the unique hash code, now it's very hard to change it because of the high data security in the blockchain technology. Next, there is a verification module that ensure that the dataset remains unchangeable via the comparison between the existing hash code inside the blockchain and the new one calculated in the time just before starting to train the ML model. If the hash code is the same then there is no problem, the dataset is the same and it is unchangeable. If not, then there are no doubts, the data has been tampered and that would cause a big problem and lack of accuracy of the ML model.

In this system, there are two main phases. First, the storing phase is the process of computing of the hash code via the hashing module and storing it inside the ledger. Second, the verification phase is where the check of the computed hash at the time just before the training of the ML model. The figure 1 present the full system architecture:

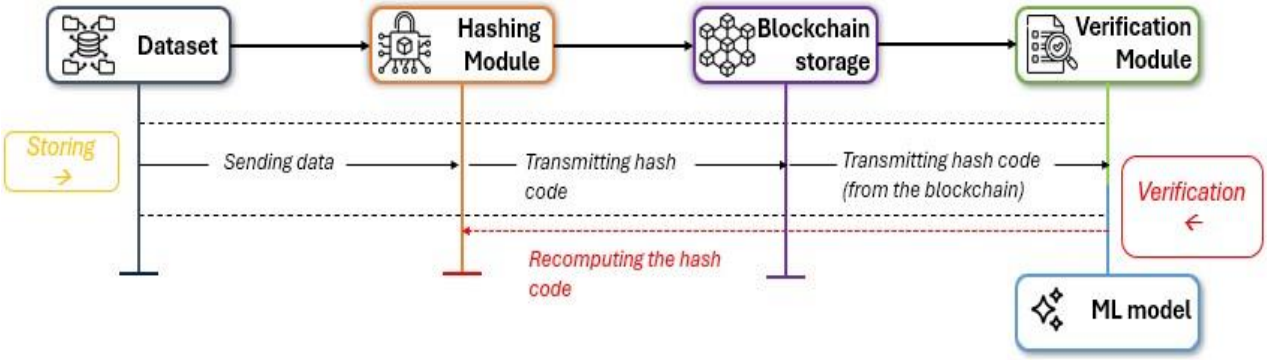

***Figure 1**: System Architecture*

c. Workflow

In order to ensure the integrity and authenticity of the exchanged data, we propose a simple yet effective verification workflow based on cryptographic hashing and blockchain storage. The process is divided into two main phases as mentioned before. At first, the registration phase is the process when the original data is processed and transformed into a secure hash value using the SHA-256 algorithm, which is then stored on the blockchain as a trusted reference. Secondly, the verification is the phase where any incoming data is subjected to the same hashing process, and the resulting hash is compared with the one previously recorded. The comparison result allows the system to determine whether the data has remained unchanged or has been altered. The output of this phase is then one of two possibilities. First, a positive signal to start the training of the ML model and that happens when the hash codes are the same. Second, an alert indicating a potential integrity violation in case the second computed hash code isn't the same as the first one. The following figure shows the full described workflow:

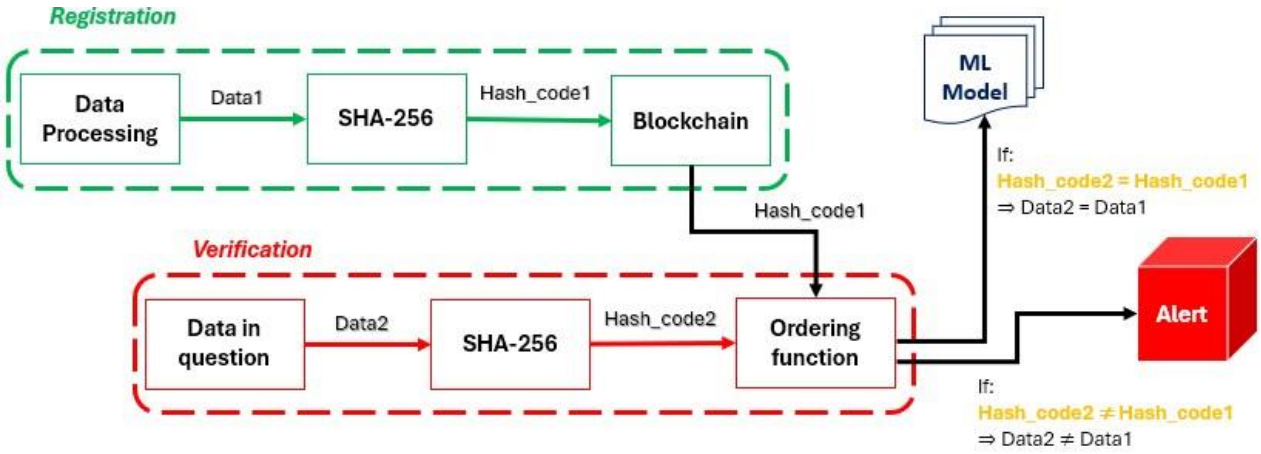

Figure 2: *Data security workflow*

d. Algorithm

In order to maintain data integrity in this project, the combination of cryptographic hashing and secure storage of hashes in a Blockchain is employed. Firstly, once data processing is completed, it is necessary to hash the original dataset using the SHA-256 algorithm and store the resulting hash on the blockchain. Secondly, prior to the commencement of the training stage, it is required to rehash the intended training dataset using the same algorithm and compare the results with the hash value present on the blockchain. Finally, if the second computed hash is identical to the one stored it means that both the datasets are the same and the data is therefore legitimate. Otherwise, this means that there was some alteration in the data, and it must not be used. Through this procedure, which offers an effective way to ensure data integrity in the training process, machine learning models can safely perform their duties. This is illustrated in the third figure.

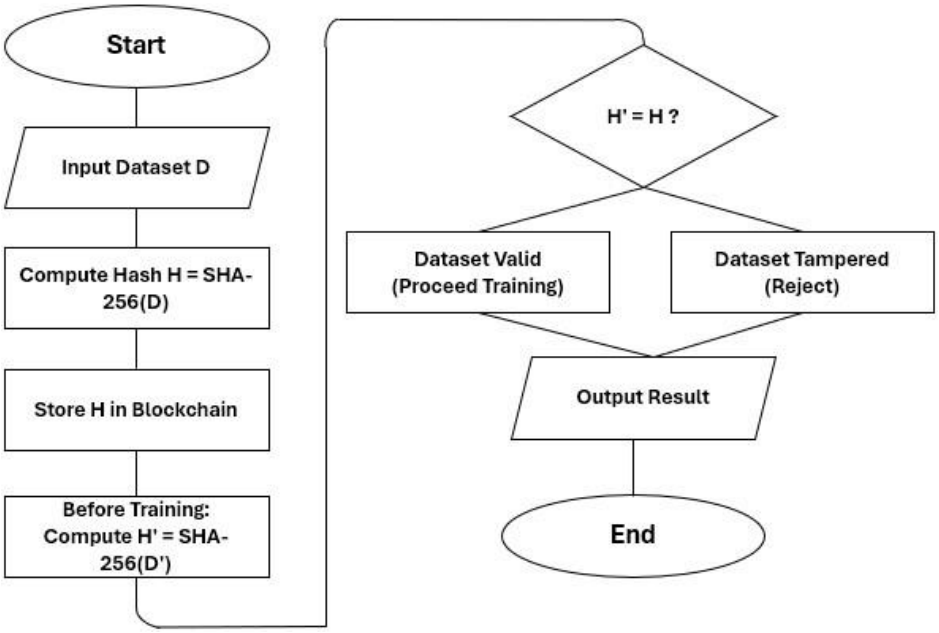

Figure 3: *Algorithm for Data Integrity Check*

e. Proposed Cryptographic Solution

A cryptographic hash (sometimes called 'digest') is a kind of 'signature' for a text or a data file. SHA-256 is the adopted hash function in the Bitcoin system because of its high security. it generates an almost-unique 256-bit (32-byte) signature for a file (CSV, JSON, images, NumPy arrays, Parquet, etc.). It cannot be restituted back to the original data which makes it suitable for this matter. This function is used to map an input dataset of arbitrary size to a fixed-length output of 256 bits. Formally, the hash function is defined as:

$H: \{0,1\}^* \rightarrow \{0,1\}^{256}$

$$\{D \mapsto \text{SHA-256}(D)$$

where $D \in \{0,1\}^*$ represents the dataset encoded as a binary string, and $H \in \{0,1\}^{256}$ is the corresponding hash value. SHA-256 satisfies collision resistance, defined as:

$Pr[\exists D_1 \neq D_2 : H(D_1) = H(D_2)] \approx 2^{-256}$

This implies that finding two distinct datasets producing the same hash is computationally infeasible. However, the hash depends on the exact byte representation. So, there is a challenge that's is not the format itself but the consistency, which means that two datasets with the same content but different formatting (e.g., column order, spacing, encoding) will produce different hashes. Also, if the dataset is loaded into memory and then re-saved, even small changes (like float precision or metadata) can change the hash which means for complex datasets (e.g., collections of files, folders, or streaming data), we need a deterministic way to serialize them before hashing and that leads us to canonical hashing. This type of hashing gives data normalization (e.g., fixed ordering, encoding, structure) before hashing and that is more robust across environments. The fourth figure below show clearly these challenges in a chorological way.

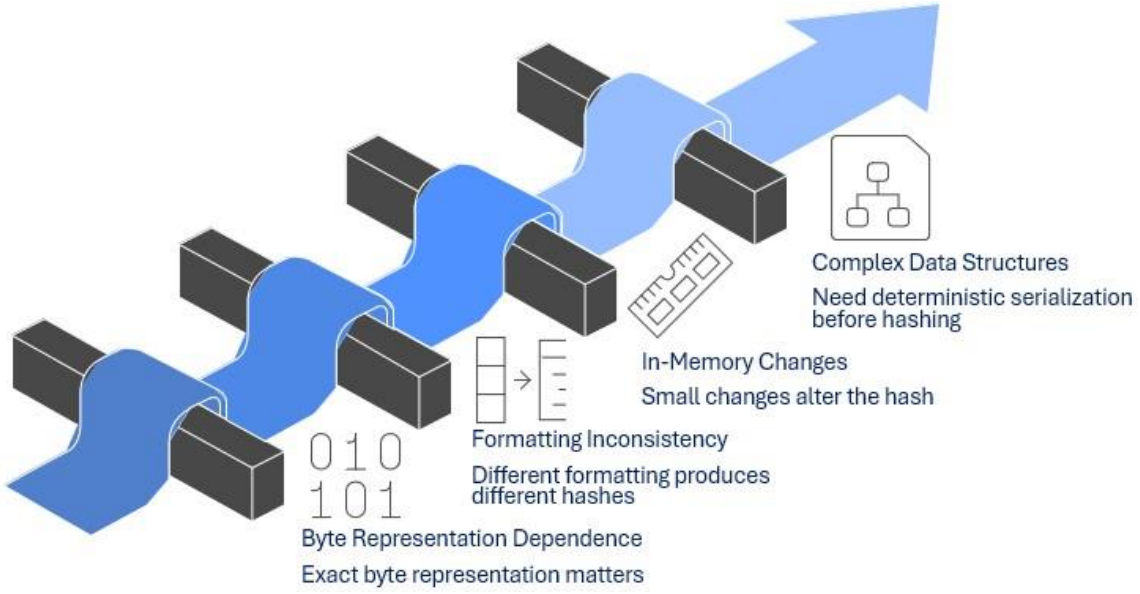

Figure 4: *Dataset Hashing Challenges*

IV. Implementation

a. Tools and Environment

The system under consideration has been implemented using the Python programming language because of its versatility and the high level of support provided by the language for data manipulation and analysis as well as machine learning tasks. The experimental framework uses several popular scientific modules such as NumPy and Pandas that provide powerful tools for working with datasets. In order to construct the machine learning algorithm, the Scikit-learn module was chosen because of its simplicity and reliability, allowing one to create an effective simple linear regressor in order to view the model's accuracy after the data manipulation. Moreover, the Python hashlib module was used to calculate SHA-256 hashes of all records in the dataset.

b. Dataset and machine learning model

This research makes use of a dataset that has information regarding the car specification and CO_2 emissions measurement. This automotive dataset is available publicly on Kaggle [5]. The attributes included in the dataset include the size of the engine, number of cylinders, and other metrics such as city fuel efficiency, highway fuel efficiency, and combined fuel efficiency (in liters/100km). The target attribute, CO_2 emission, is given in terms of grams/km. Data cleaning was performed before carrying out any analysis on the data by excluding all cases with missing attributes and selecting important attributes to model the data. In this study, the attribute used as the predictor is combined fuel consumption because it correlates highly with CO_2 emissions. In the study, the machine learning model used is simple linear regression. The process of splitting the data into the training set and testing set was done in an 80/20 ratio. A tampered version of the dataset is created to simulate the integrity attack of data poisoning.

c. Blockchain simulation

Blockchain technology for integrity verification is achieved using a Python simulation process. First, the dataset is encoded into serialized form and hashed using the SHA-256 algorithm from the available Python libraries. In order to simulate blockchain technology in a simple form, the blockchain is modeled using a list with several blocks. Each of these blocks contains the hash of the dataset and other details including the index of the block and then comes the verification process which involves calculating the dataset hash and comparing it with the hash recorded in the blockchain. If there is any difference between these two values, this indicates that the dataset has been tampered with. The fifth figure illustrates the proposed blockchain simulation via python.

Figure 5: *Python code for the blockchain simulation*

V. Experimental Results

a. Experimental Setup

The test environment is set up in order to measure the performance of the suggested framework of integrity verification in two different conditions. First, when the original data set is used without any alterations, which corresponds to a normal and trustworthy training environment and when the dataset is deliberately changed by adding some minor changes to the dataset. For the first case the hash code is stored as detailed before and for the second one the SHA-256 hash of the dataset is computed and compared with the reference hash stored in the blockchain. Thus, the system can conclude whether the data set was manipulated before entering the training process.

b. Results

This section presents the experimental results obtained from evaluating the proposed framework under both clean and tampered dataset conditions. The objective is to assess the impact of data integrity on model performance and validate the effectiveness of the blockchain-based verification mechanism.

i. Impact of Data Tampering on Model Performance

The first training and testing of the linear regression model took place utilizing the initial dataset that was not subjected to any tampering. A linear correlation between the fuel consumption (Comb) value and the CO2 emissions can be easily observed, implying the successful performance of the model. In order to assess the stability of the model, tampering was simulated with the help of adding Gaussian noise to the dataset.

The results below in the sixth and the seventh figures that show difference in terms of ML model accuracy indicate how effectively the performance of the regression model has deteriorated as a result of data tampering. In the clean dataset scenario, there exists a strong linear relationship between the amount of fuel burned and CO_2 emitted, which allows for the creation of an effective regression model, capable of fitting the data perfectly with low errors. When noise is added to the dataset to create a data poisoning situation, however, the nature of the data changes, losing its linear nature and becoming much more dispersed around the regression line. This means that not only has the relationship between the independent variable (fuel consumed) and the dependent variable (CO_2 emitted) become weaker, but also that it will be harder for the regression model to identify any patterns while training on the poisoned data set. As a result, the effectiveness of the model is greatly reduced.

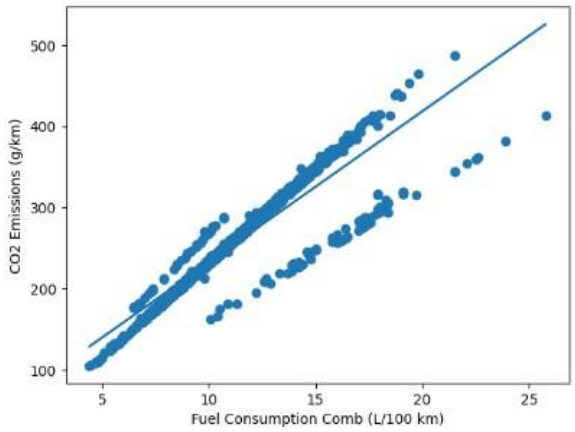

Figure 6: *Result visualization of clean data*

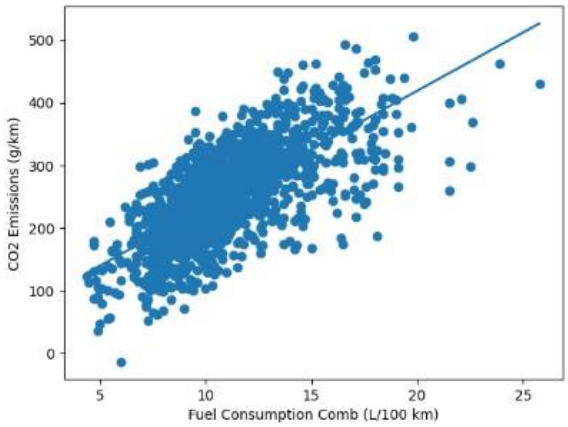

Figure 7: *Result visualization of tampered data*

ii. Performance Comparison

To quantitatively evaluate the impact of tampering, we compare the R^2 score of the model trained on clean data versus tampered data. The coefficient of determination is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where:

- y_i **denotes the actual value,**
- $\hat{y}_i$ **denotes the predicted value,**
- $\bar{y}$ **is the mean of the observed values,**
- n **is the number of observations.**

The results show a substantial drop in performance after data tampering. The reduction in R^2 score confirms that even small modifications in the dataset can lead to significant degradation in model accuracy. In this case, data poisoning in CO_2 emission datasets leads not only to degraded predictive performance but also to incorrect representation of the underlying environmental relationship, which may result in misleading environmental assessments and faulty policy decisions if deployed in real-world systems. This highlights the vulnerability of machine learning systems to data integrity attacks. The following figure shows this difference clearly:

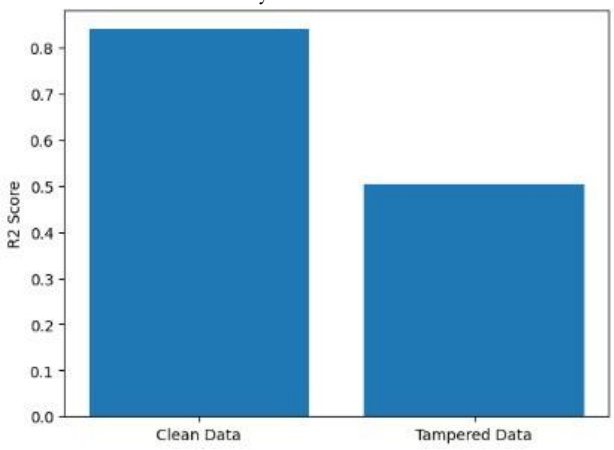

Figure 8: *Model performance comparison (R2 Score)*

VI. Discussion

It can be seen from the above results that the proposed architecture successfully detects any change in the dataset before the training process takes place due to the fact that even minimal differences in the data yield completely different hash values. This highlights the importance of ensuring the integrity of the data in order to keep the reliability and performance of ML models intact. Another important benefit of the presented solution is that it is lightweight, thus being easily deployable in various settings.

Future research may consider overcoming these challenges through incorporation into a concrete blockchain platform along with the extension of the proposed method for enabling continuous data validation in changing conditions. Additionally, coupling the developed model with anomaly detection or federated learning security schemes might offer a more robust countermeasure against sophisticated threats like adaptive data poisoning.

VII. Conclusion

In this paper, a novel framework to ensure the integrity of machine learning training datasets based on the concepts of cryptographic hashing and blockchains was designed and implemented. The use of the SHA-256 hash algorithm allowed creating a unique hash value for each dataset and storing it for further comparison and verification prior to training. Therefore, the implementation provided an efficient tool for detecting and preventing any modifications to the dataset.

The findings of the conducted experiments prove the significance of data integrity for the effective operation of MLbased systems. Even the slightest modifications of the dataset cause significant changes in model performance. However, the proposed framework provides the means to detect such modifications effectively while being simple and easy to implement, both of which make the solution appropriate for practical use.

Nevertheless, the present solution works only for integrity verification at a particular stage and cannot offer any realtime security during the data gathering or transferring process. Furthermore, the usage of the blockchain simulator does not allow for considering all the restrictions associated with the real-life implementation of this concept. In future research, we plan to study how to incorporate real blockchain implementations into our system and apply this framework to dynamic scenarios, possibly together with federated learning. Thus, this paper suggests an efficient way of enhancing the security of AI systems using data integrity verification.

Acknowledgments

This work is supported by the National Center for Scientific and Technical Research CNRST, Rabat, Morocco.

References

[1] Kim, S.-K. (2025). Enhanced blockchain-based data poisoning defense mechanism. *Applied Sciences, 15*, 4069.

[2] Venkatesan, K., & Rahayu, S. B. (2024). Blockchain security enhancement: An approach towards hybrid consensus algorithms and machine learning techniques. *Scientific Reports, 14*, 1149.

[3] Usama, M., Aziz, A., Alasbali, N., Alturki, N., Hanif, M., & Rehman, M. U. (2026). Blockchain-enabled identity management for IoT: A multi-layered defense against adversarial AI. *Scientific Reports, 16*, 4371.

[4] Wankhede, S. B., & Patel, D. (2025). Federated learning and blockchain approach for securing IoT data. *Discover Internet of Things, 5*, 116.

[5] Pinuto, P. (n.d.). *Ecodrive: CO2 emissions efficiency analysis* [Dataset]. Kaggle. https://www.kaggle.com/code/pinuto/ecodrive-co2-emissions-efficiency-analysis/input