applied sciences

MDPI

Article

Integrating Merkle Trees with Transformer Networks for Secure Financial Computation

Xinyue Wang †, Weifan Lin †, Weiting Zhang †, Yiwen Huang, Zeyu Li, Qian Liu, Xinze Yang, Yifan Yao and Chunli Lv *

China Agricultural University, Beijing 100083, China; yxz0113@cau.edu.cn (X.Y.)
* Correspondence: lvcl@cau.edu.cn
† These authors contributed equally to this work.

Abstract: In this paper, the Merkle-Transformer model is introduced as an innovative approach designed for financial data processing, which combines the data integrity verification mechanism of Merkle trees with the data processing capabilities of the Transformer model. A series of experiments on key tasks, such as financial behavior detection and stock price prediction, were conducted to validate the effectiveness of the model. The results demonstrate that the Merkle-Transformer significantly outperforms existing deep learning models (such as RoBERTa and BERT) across performance metrics, including precision, recall, accuracy, and F1 score. In particular, in the task of stock price prediction, the performance is notable, with nearly all evaluation metrics scoring above 0.9. Moreover, the performance of the model across various hardware platforms, as well as the security performance of the proposed method, were investigated. The Merkle-Transformer exhibits exceptional performance and robust data security even in resource-constrained environments across diverse hardware configurations. This research offers a new perspective, underscoring the importance of considering data security in financial data processing and confirming the superiority of integrating data verification mechanisms in deep learning models for handling financial data. The core contribution of this work is the first proposition and empirical demonstration of a financial data analysis model that fuses data integrity verification with efficient data processing, providing a novel solution for the fintech domain. It is believed that the widespread adoption and application of the Merkle-Transformer model will greatly advance innovation in the financial industry and lay a solid foundation for future research on secure financial data processing.

Keywords: financial computation model; Merkle tree; secure data handling; deep learning; artificial intelligence

Citation: Wang, X.; Lin, W.; Zhang, W.; Huang, Y.; Li, Z.; Liu, Q.; Yang, X.; Yao, Y.; Lv, C. Integrating Merkle Trees with Transformer Networks for Secure Financial Computation. *Appl. Sci.* **2024**, *14*, 1386. https://doi.org/10.3390/app14041386

Academic Editor: Andrea Prati

Received: 2 January 2024
Revised: 30 January 2024
Accepted: 6 February 2024
Published: 8 February 2024

1. Introduction

In the digital era, the financial industry is experiencing rapid transformation, with an increasing number of financial services and transactions being conducted online. This shift has made financial data security an increasingly critical concern [1]. Financial transaction data not only hold significant economic value but also carry personal privacy and corporate secrets, making it crucial to ensure the security and integrity of such data. In order to enhance the guidance for investors interested in financial transactions or for analysts attempting deception, the establishment of a more transparent and trustworthy information system is first required. This can be achieved by ensuring that data and information in financial markets are made publicly available, accurate, and promptly accessible to both investors and analysts. Secondly, regulatory authorities need to intensify their supervision and enforcement efforts in financial markets to deter potential fraudulent activities. Furthermore, investors and analysts should receive high-quality financial education to comprehend the fundamental principles and risks of financial markets, enabling them to better grasp and address market fluctuations. Simultaneously, they also need to acquire

the knowledge of identifying potential fraudulent activities and speculative operations to safeguard their investments. Lastly, the development of technological and data analysis tools can assist investors and analysts in making more informed decisions. By employing advanced algorithms and big data analysis techniques, they can gain a better understanding of market trends, identify potential risks, and consequently, formulate investment strategies with greater confidence. Traditional security measures, such as encryption and access control, provide protection to a certain extent, but their effectiveness is often limited in the face of complex cyber-attacks and internal threats [2]. Moreover, with the rapid development of financial technology, new security challenges continue to emerge, necessitating the development of more advanced and reliable security technologies.

Wu et al. [3] proposed an enhanced blockchain-based secure sharing (EBSS) scheme for real estate financial certificates that protects shared vouchers and allows for anonymous authentication; however, it is costly. Zhao et al. [4] designed a new method for secure online data backup to address issues such as data loss. Li et al. [5] researched a multi-chain model based on financial transactions and its corresponding cross-chain transaction protocol, achieving interoperability between homogeneous or heterogeneous blockchains, though the actual blockchain environment is complex, affecting its application effectiveness. In response, Du et al. [6] proposed an anti-tampering data query model based on B+ trees and Merkle trees, addressing the issue of single-query blockchain data systems, with experimental results indicating that this method performs best.

Among the emerging technologies, Merkle trees (Merkle Tree) and the Transformer model have attracted widespread attention due to their unique characteristics. Merkle trees, a hash-based data structure, efficiently and securely verify data content and have been successfully applied in various distributed systems and cryptocurrencies. Their main advantage lies in the ability to verify data integrity, enabling the validation of a part of the data without revealing the entire content [7], which is crucial for protecting the privacy and security of financial data.

On the other hand, the Transformer model, as a revolutionary deep learning architecture, has demonstrated outstanding performance in natural language processing (NLP) and several other domains [8]. Its main advantage is its capability to process sequential data, particularly in understanding complex, long-distance dependencies [9]. In the financial sector, this ability can be employed to capture complex patterns in transaction data, thereby improving the accuracy of financial fraud detection and the capability to predict market changes.

Lian et al. [10] proposed a financial product recommendation network based on the Transformer, showing that it has a faster training speed and better results than RNN. Zhang et al. [11] used the Transformer for stock trend prediction, proposing a Transformer encoder-based attention network (TeaNet) architecture, addressing the time-dependency issue of financial data. Chen et al. [12] proposed an MTCformer method combining multi-channel TextCNN and Transformer for Ponzi scheme detection, utilizing the Transformer to extract semantic features, with experimental results showing that MTCformer achieved 97% accuracy, outperforming other methods (although it is time-consuming). Kim Gyeongmin et al. [13] proposed a Bitcoin trading model—Cryptocurrency BERT Incorporated Trading System (CBITS); research indicates that our model significantly outperforms other trading methods.

However, despite the significant advantages of Merkle trees and the Transformer model, research combining the two in the field of financial security remains relatively limited. This paper proposes an innovative combination—the Merkle-Transformer model—to explore the application potential of these two technologies in secure financial computing. The objective is to enhance the security and efficiency of financial data processing by combining the data verification capability of Merkle trees with the data processing power of the Transformer model.

Karagiannis Ioannis et al. [14] proposed a method based on blockchain for sharing financial information between financial institutions, but it cannot guarantee the integrity of

data during processing; Alegria Alberto Vidal et al. [15] presented a quantitative analysis method for calculating the risk of cyber-attacks in the data security domain of finance; however, complex environments may affect the performance of this method in practical applications. Ahmed Mohammad Rasheed et al. [16] proposed a blockchain-based payment system for the Indian financial industry, which can achieve considerable speed, but it has not been tested in real-time, and the performance of the system under different network scenarios cannot be guaranteed.

The design of the Merkle-Transformer model considers multiple aspects of financial data processing. First and foremost, the model ensures the security and integrity of data during processing, which is vital in domains such as financial transactions and stock price prediction. Secondly, by effectively integrating the structure of Merkle trees, the model provides an additional layer of security for financial data. This not only helps prevent data tampering but also makes the data verification process more efficient. Lastly, by combining the advantages of the Transformer model, our model exhibits excellent performance in processing large-scale financial data, especially in capturing complex patterns and long-term dependencies.

In this paper, the design and implementation of the Merkle-Transformer model will be elaborated, including the verifiable computation module based on Merkle trees and the secure verifiable Transformer model. Additionally, a new loss function—the Merkle loss function—specifically designed for the model will be introduced to optimize its performance and enhance security. In the results and discussion section, the application results of the model on financial transaction data (for financial attack detection) and stock price data (for stock price prediction) will be showcased. Through these results, the advantages of the Merkle-Transformer model in terms of performance, security, and cost will be verified, and its application prospects in the domain of secure financial computing will be explored. The methodology presented in this paper is aimed at enhancing fairness and stability in financial markets, while concurrently augmenting the decision-making capabilities and confidence of investors and analysts. This is achieved by devising the model to ensure the aforementioned objectives are met. In conclusion, the Merkle-Transformer model not only provides a new approach to the secure processing of financial data but also holds significant theoretical and practical implications. Through the research presented in this paper, a new perspective and solution for the development of financial technology security is anticipated to contribute.

2. Related Works

2.1. Hash Algorithms

In the subsection on hash algorithms, it is highlighted that hash algorithms play a critical role in the field of financial security, as they provide a foundation for data integrity and security [17]. An introduction to hash verification algorithms is offered, including their characteristics and applications in detecting financial attacks. Hash algorithms are functions that transform an input of arbitrary length (usually a file) into a fixed-length string, referred to as a hash value [18]. The main characteristic of a hash algorithm is its unidirectional nature, meaning that it is computationally infeasible to reverse-engineer the original input from the hash value. Furthermore, hash algorithms should satisfy the following important properties:

1. **Determinism**: A hash algorithm must produce the same output for any given input. For example, for input x, the output of the hash function, H, should always be the fixed $H(x)$.
2. **Efficient computation**: Computing the hash value $H(x)$ should be quick for any input value x.
3. **Collision resistance**: It should be extremely difficult to find two different inputs (x and y) that yield the same hash value, i.e., $H(x) = H(y)$.

4. **Avalanche effect**: A small change in the input value should cause a significant change in the hash value. In other words, if x and x' differ slightly, then $H(x)$ and $H(x')$ should be significantly different.

This paper will focus on how these hash algorithms are applied to the secure processing of financial data, particularly in scenarios involving the integration with Merkle trees and the Transformer model. By combining the data integrity protection of hash algorithms with the efficient data processing capabilities of the Transformer model, the proposed Merkle-Transformer model aims to provide an innovative solution for the secure processing of financial data.

2.2. Merkle Tree

In this section, we describe how Merkle trees (which are hash-based data structures) are extensively utilized in various scenarios for security and data integrity verification [19], particularly in the fintech sector. Essentially, Merkle trees are a special type of binary tree where each non-leaf node is the hash of the hash values of its child nodes, as depicted in Figure 1.

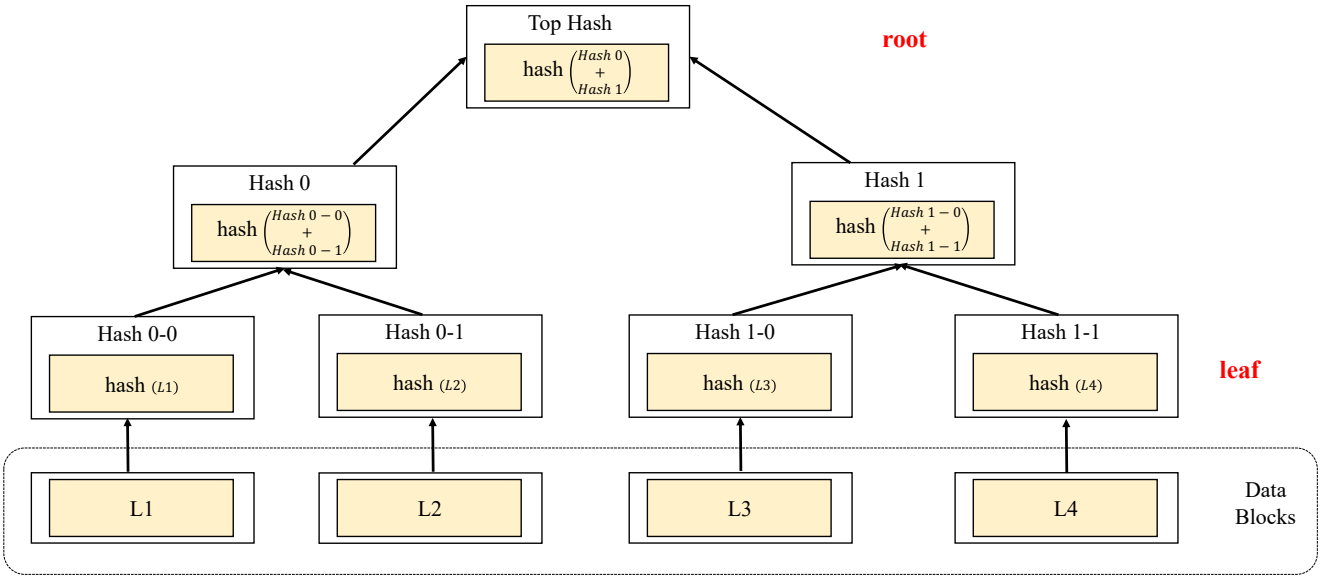

Figure 1. Illustration of a Merkle tree.

Within a Merkle tree, leaf nodes contain hash values of data blocks (such as transaction records or file segments), while each non-leaf node contains the combined hash of the hash values of its two child nodes. The top of the tree features a single node, known as the root node or Merkle root, representing a unique fingerprint of the entire dataset. The construction of a Merkle tree can be mathematically expressed as follows:

$$MerkleTree = Hash(Hash(A) \oplus Hash(B) \oplus Hash(C) \oplus \ldots) \quad (1)$$

Here, A, B, C, etc., represent data blocks, $\oplus$ represents concatenation, and the $Hash()$ function is utilized to generate hash values.

In the context of financial attack detection, Merkle trees can ensure the integrity and immutability of transaction records. Consider, for example, a blockchain system comprising multiple transaction records. Each block in the blockchain can utilize a Merkle tree to store the hash values of transactions [20]. To verify whether a specific transaction is included in a block, only a path from that transaction record to the Merkle root (i.e., a Merkle proof) is required.

This process can be represented by the following equation:

$$MerkleProof = Hash(Hash(Transaction) \oplus Hash(Sibling_1) \oplus \ldots \oplus Hash(Sibling_n)) \quad (2)$$

Here, *Transaction* is the transaction record being verified, and $Sibling_1, \ldots, Sibling_n$ are sibling nodes on the path from the transaction record to the root. In practical applications, Merkle trees offer an efficient and secure way to verify the integrity of large volumes of data. In the financial domain, this characteristic is especially significant. It not only ensures the security of transaction data but also effectively guards against financial fraud and tampering activities [21]. For instance, in distributed ledger technologies (such as blockchain), Merkle trees are used to ensure the integrity of transaction records in each block. Through this method, participants can verify the validity of individual transactions, even without downloading the entire blockchain. A key advantage of Merkle trees is their "avalanche effect": any change in a data block within the tree leads to a change in the entire tree's root hash value. This feature allows for any unauthorized modifications to the data to be rapidly detected, thereby providing a robust mechanism for data integrity protection.

In the Merkle-Transformer model, these characteristics of Merkle trees are leveraged to enhance the security of financial data processing. By combining Merkle trees with the Transformer model, a secure and efficient method for processing complex financial data is provided, particularly when dealing with large-scale and high-frequency transaction data.

2.3. Financial Computation Model Based on Transformer

In the subsection on Transformer-based financial computation models, the Transformer's exemplary performance in various domains, particularly in finance, is emphasized since its introduction [22]. This model, proposed by Vaswani et al. in 2017 [23], revolutionized sequence data processing with its "Self-Attention" mechanism. Compared to traditional recurrent neural networks (RNNs) [24] and long short-term memory networks (LSTMs) [25], the Transformer addresses long-distance dependencies more effectively, a critical aspect in financial time-series data analysis. The core of the Transformer is its self-attention mechanism, allowing the model to consider other elements while processing each element in a sequence, thereby capturing complex patterns in data more effectively. The computation of self-attention can be represented as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3)$$

Here, Q, K, and V represent the query, key, and value, respectively, with d_k denoting the dimension of the key. This mechanism enables the model to allocate different attention weights to each element in a sequence, as shown in Figure 2.

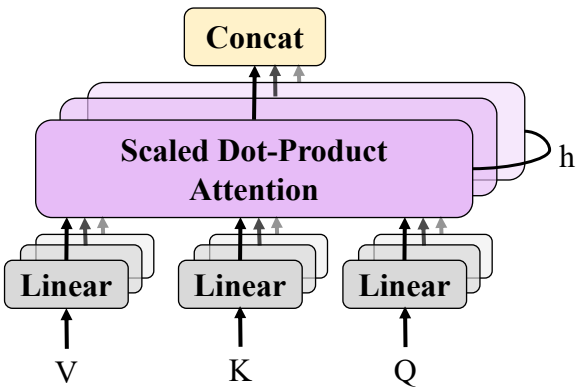

Figure 2. Illustration of the multi-head attention mechanism.

The Transformer model also includes two main parts: an encoder and a decoder. The encoder consists of multiple identical layers, each with a self-attention mechanism and a simple feed-forward neural network. The decoder comprises multiple identical layers, but each decoder layer includes an additional attention mechanism focusing on the encoder's output. In financial computation models, the Transformer is employed for various tasks, such as stock price prediction, risk assessment, and market trend analysis [26]. In these applications, the Transformer effectively captures complex patterns and long-term dependencies in financial time-series data. For instance, in stock price prediction models, the Transformer can be represented by the following formula [27]:

$$P_t = f(\text{Transformer}(S_{t-n}, \ldots, S_{t-1})) \tag{4}$$

where P_t denotes the stock price prediction at time t, and $S_{t-n}, \ldots, S_{t-1}$ represent the stock price series at previous n time points, with f being the function linking the Transformer output to the prediction result.

Another significant application of the Transformer in finance is fraud detection [28]. In this scenario, the model must identify anomalous patterns from a vast array of transaction data. With its self-attention mechanism, the Transformer can efficiently identify these complex fraud patterns. A Transformer model for fraud detection can be represented as follows:

$$F_t = \text{Transformer}(\text{Transaction}_1, \ldots, \text{Transaction}_t) \tag{5}$$

where F_t represents the fraud detection outcome at time t, and $\text{Transaction}_1, \ldots, \text{Transaction}_t$ denote the transaction sequence up to time t. In conclusion, the Transformer model plays a crucial role in financial computation models due to its efficient handling of long-distance dependencies and capturing of complex patterns. A deeper understanding of the Transformer's structure and principles enables the design of more accurate and efficient financial computation models, achieving breakthroughs in areas like stock price prediction, risk assessment, and fraud detection. In the Merkle-Transformer model, these advantages of the Transformer will be utilized, combined with the security of Merkle trees, to provide a secure and efficient solution for financial data processing.

3. Materials and Method

3.1. Dataset Collection

3.1.1. Financial Transaction Datasets

In the subsection on financial transaction data, it is noted that financial transaction datasets typically consist of a vast array of transaction records, potentially including information such as transaction time, parties involved, amount, and type. These datasets are often employed for financial attack detection to identify anomalous transaction behaviors, such as fraud or money laundering. When selecting datasets, attention is paid to the diversity, authenticity, and completeness of the data. Prominent public financial transaction datasets include the Credit Card Fraud Detection dataset provided by Kaggle [29]. The reason for utilizing these datasets is that they offer a wealth of real-world cases, assisting in effectively testing and optimizing the performance of the Merkle-Transformer model in financial attack detection. By analyzing real transaction data, more accurate simulation and identification of financial fraud and other abnormal behaviors can be achieved.

3.1.2. Stock Price Datasets

In the subsection on stock price data, stock price datasets, including historical trading data of the stock market such as opening and closing prices, highest and lowest prices, and trading volume, are discussed. These datasets are typically used for training and testing stock price prediction models, to forecast future stock market trends. The stock price data for this study are obtained from various public financial market databases, including Yahoo Finance, Google Finance, and others [29]. Additionally, professional financial data service providers like Bloomberg and Reuters offer high-quality stock price data. The reason for

using these datasets is that they provide detailed market transaction records, reflecting the collective behavior and sentiments of market participants. By analyzing the data, the model can learn complex patterns affecting stock price fluctuations, thereby enhancing prediction accuracy. After acquiring stock price data, data cleaning and preprocessing are conducted to ensure data quality and consistency. Specific to stock price prediction, various technical indicators such as moving averages, Relative Strength Index (RSI), and Bollinger Bands are calculated as input features for the model. Moreover, external factors like macroeconomic indicators and market news, which also influence stock price volatility, are considered.

3.1.3. Generated Datasets

In the subsection on generated datasets, the purpose of generating datasets is to improve the generalization ability and robustness of the model. In the real world, financial data are often imbalanced, particularly in scenarios of financial attack detection where normal transactions far outnumber fraudulent ones. To address this issue, data generation techniques are used to balance the datasets. A primary method employed is the synthetic minority over-sampling technique (SMOTE) [30]. SMOTE generates new samples by interpolating among minority class samples, thereby balancing class distribution. This method enhances model performance in handling imbalanced data. The basic idea of SMOTE is to find the k nearest neighbors for each minority class sample and then generate new samples through random interpolation between these samples. Assuming a minority class sample x_i with its nearest neighbor set $\{x_{i1}, x_{i2}, \ldots, x_{ik}\}$, a new sample x_{new} is generated as follows:

$$x_{new} = x_i + \lambda \cdot (x_{ij} - x_i) \tag{6}$$

Here, x_{ij} is a nearest neighbor of x_i, and λ is a randomly chosen parameter within [0, 1]. Through these methods, not only high-quality datasets are obtained, but the model's robustness and generalization ability are also enhanced through data generation techniques. This is crucial for the development and testing of the Merkle-Transformer model.

3.2. Dataset Preprocessing and Augmentation

In the subsection on dataset preprocessing and augmentation, it is outlined that preprocessing and augmenting the dataset are key steps in experimental preparation. This work ensures the quality and diversity of the dataset, crucial for subsequent model training and evaluation. A detailed explanation of data preprocessing and augmentation methods is provided, including annotation methods, preprocessing techniques, and the application and mathematical principles of data augmentation techniques.

3.2.1. Dataset Annotation

In the context of financial data processing, dataset annotation is an important step, particularly in applications such as financial attack detection and stock price prediction. For financial transaction data, the focus is primarily on labeling transactions as normal or anomalous (fraudulent). Stock price data, on the other hand, needs to be labeled according to price trends within a specific time window, such as rising, falling, or stable.

1. **Annotation of financial transaction data**: Based on historical records and background information of transactions, each transaction is labeled as "normal" or "fraudulent".
2. **Annotation of stock price data**: For stock price data, annotation is typically based on historical trends of price changes. For example, if the stock price in a given time window rises beyond a certain threshold, it might be labeled as "rising"; conversely, if the price falls below the threshold, it is labeled as "falling".

3.2.2. Dataset Preprocessing

Data preprocessing is a key step to ensure data quality and model performance. Initially, data cleaning involves removing or correcting erroneous and inconsistent records in the dataset. This is to ensure the accuracy of model training and evaluation, preventing

incorrect data from leading the model to learn incorrect patterns. Following this, normalization or standardization processes involve transforming all features to the same scale for more effective learning and generalization by the model. Common methods include min-max normalization and Z-score standardization. The mathematical expression for Z-score standardization is as follows:

$$x' = \frac{x - \mu}{\sigma} \tag{7}$$

Here, x represents the original feature value, μ and σ represent the mean and standard deviations of the feature values, and x' is the standardized feature value.

Handling missing values is also a crucial aspect of data preprocessing. For missing data, various methods can be employed, such as interpolation, filling with mean values, or simply deleting related records. Finally, feature engineering involves selecting and constructing features according to the needs of the model. In financial data analysis, especially for stock price prediction, calculating moving averages or other technical indicators as input features for the model might be necessary.

3.2.3. Dataset Augmentation

Data augmentation is an important method to enhance model generalization ability, especially in situations with limited data or imbalanced class distribution. Common methods of data augmentation in financial data processing include the following:

1. **Synthetic minority over-sampling technique (SMOTE)**: As previously mentioned, SMOTE is a popular data augmentation method used to address class imbalance issues. It generates new samples by interpolating between minority-class samples.
2. **Time series data augmentation**: For stock price data, time series-specific data augmentation methods such as time window sliding and time series segmentation can be used to increase the number and diversity of data samples.
3. **Feature expansion**: In some cases, creating new features by adding noise, transforming, or combining existing features can enhance the diversity of the dataset.

Through the above data preprocessing and augmentation steps, the quality and diversity of the dataset can be ensured, providing a solid foundation for the training and evaluation of the Merkle-Transformer model.

3.3. Merkle-Transformer

3.3.1. Overall

In this paper, we propose an innovative method that combines the advantages of Merkle trees and Transformer models to enhance the security and efficiency of financial computations. The core concept of the Merkle-Transformer model is to ensure the security and integrity of data through Merkle trees while leveraging the efficient computational capability of the Transformer model. This combination not only improves the accuracy of data processing but also ensures that security is not compromised in the pursuit of efficiency.

Figure 3 illustrates a schematic of the working principle of the Merkle-Transformer model. The figure elucidates how the outputs of the intermediate layers of a deep learning model are combined with the Merkle tree structure to ensure the security and verifiability of the model's processing. Each layer (L1 to L8) are first considered as leaf nodes in the Merkle tree. Subsequently, adjacent outputs are paired and combined through a hash function Hash, progressively building the Merkle tree until forming the top hash, representing a unique fingerprint of the entire model's output.

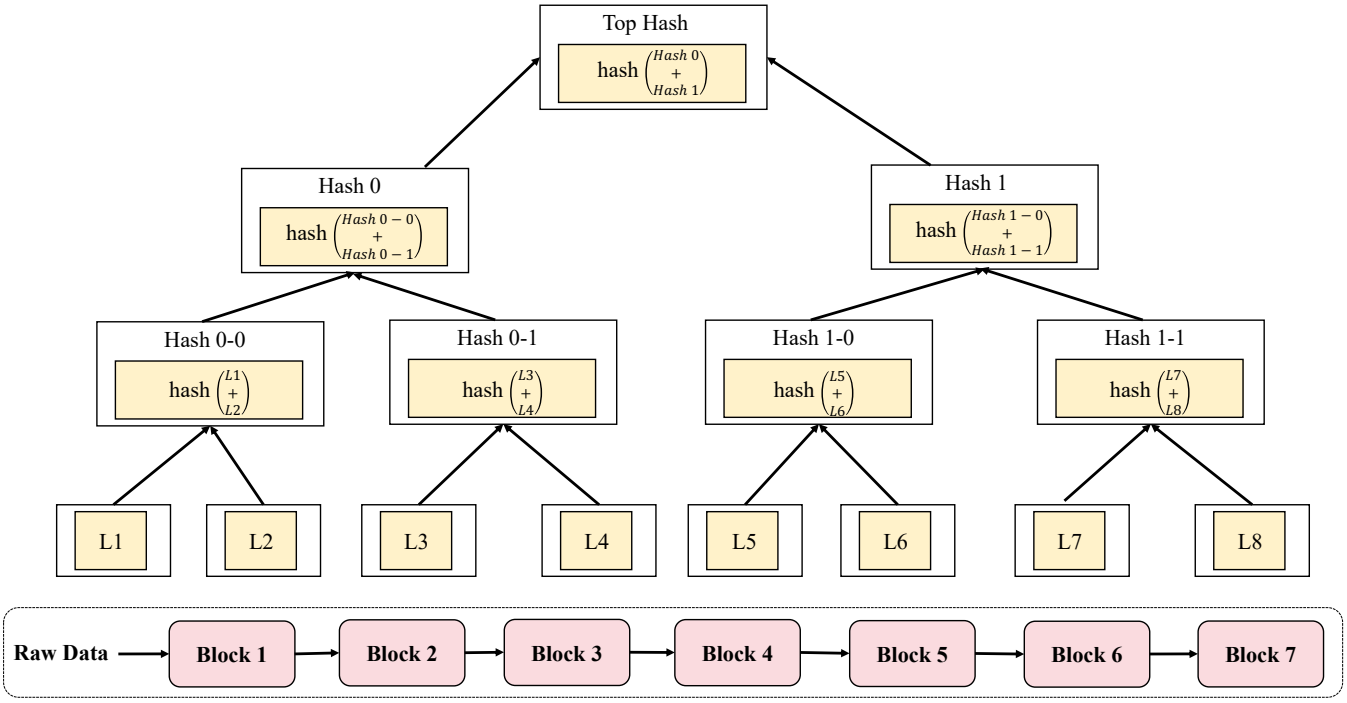

Figure 3. Overview of the Merkle-Transformer.

Specifically, the raw data passes through a series of attention modules, each generating a feature representation (L1 to L8). This paper introduces a shallow convolutional neural network at the model's entrance for initial feature extraction, and the extracted features are then fed into the Transformer model. These feature representations are further combined through the hash function to form intermediate nodes (Hash 0-0, Hash 0-1, etc.), culminating in the top hash. At the end of the model, after passing through the flattened and dense layers, the output is obtained. Throughout this process, every step of the model's output is linked to the Merkle tree structure via the hash function, implying that any minor change in output will cause a variation in the root hash, thereby verifying the integrity and unaltered state of data throughout the transmission and processing. This design enables the Merkle-Transformer model not only to process complex financial data but also to ensure security during data processing, as any unauthorized data modification will be detected. This is particularly important for financial data, often involving sensitive information and critical decision-making. Additionally, the Merkle-Transformer model provides a verifiable computation process, allowing every step of the model's operation to be traceable and verifiable, greatly increasing the transparency and trustworthiness of financial computation models.

In summary, the Merkle-Transformer model integrates deep learning and Merkle trees to provide a secure and efficient new framework for financial computations. This combination effectively utilizes the strengths of both technologies—the powerful data processing ability of deep learning and the unique security advantage of Merkle trees—offering a novel solution for financial data processing and analysis.

3.3.2. Verifiable Computation Module Based on Merkle Tree

The verifiable computation module based on Merkle trees is a core component of the model, playing a vital role in safeguarding data security and integrity throughout the computation process. The design of this module merges the data verification mechanism of Merkle trees with the efficient computation capability of deep learning, aiming to provide a secure and efficient framework for financial data processing. The design details of this module follow the basic principles of Merkle trees, where data blocks are hashed to form a tree-like structure, with each non-leaf node being the hash of the hash values of its child nodes. In our model,

this concept is extended to the outputs of the intermediate layers of deep learning models. Specifically, for each intermediate layer of the model, the hash value of its output is calculated, and these hash values are organized in the manner of a Merkle tree.

Considering that each layer in deep learning can be viewed as a data transformation, the outputs of each layer are treated as data blocks. For instance, if the model's first layer is a two-dimensional convolutional layer with input x and output h_1, then the hash value $H(h_1)$ of h_1 can be calculated. Similarly, the hash value $H(h_2)$ for the output h_2 of the second layer is computed, and so on. These hash values are then paired and hashed again to form the parent nodes of the Merkle tree, continuing until the root hash is formed. Mathematically, this process can be expressed as follows:

$$h_{\text{parent}} = H(h_{\text{left}} || h_{\text{right}}) \tag{8}$$

where h_{parent} is the hash value of the parent node, h_{left} and h_{right} are the hash values of the child nodes, H is the hash function, and $||$ denotes concatenation. In practical applications, our deep learning model may include various types of layers, such as convolutional layers (Conv2D), pooling layers, and dense layers. These layers are stacked in sequence to form a network capable of extracting and processing complex features. For example, a typical stacking structure might involve the input passing through two convolutional layers, followed by a pooling layer, several more convolutional layers, and finally, a dense layer to produce the output. The output of each layer is used not only for the input of the next layer but also for constructing the Merkle tree. The output dimensions of each convolutional or dense layer depend on the layer's configuration and the dimensions of the input. For example, the output dimension of a convolutional layer might be a function of the number of filters, stride, and padding. The advantage of this design is that it combines the efficient computational capability of deep learning models with the data integrity verification mechanism provided by Merkle trees. This combination not only enhances the model's performance in processing financial data but also increases the security of the data processing. Any unauthorized alteration to the data or model output will be reflected in the root hash of the Merkle tree and can, therefore, be quickly detected. Additionally, due to the structure of the Merkle tree, which makes the verification process highly efficient, only a small portion of nodes need to be verified to ensure the integrity of the entire dataset, significantly reducing computational and storage overhead. In financial computation tasks, this design is particularly important as it not only ensures the reliability and auditability of data processing but also allows for rapid response to potential security threats. Whether in scenarios of financial transaction detection or stock price prediction, ensuring the security and integrity of data is of utmost importance.

3.3.3. MatrixSplit Verification Method

In this subsection, it is explained that while traditional Merkle trees are highly efficient in processing single data streams, a decline in verification efficiency occurs when dealing with matrices or multi-dimensional data using conventional methods. To address this issue, a novel matrix splitting method is introduced and applied within an improved Merkle tree structure.

Figure 4 displays the matrix splitting verification method, a novel approach to data structure processing aimed at enhancing the efficiency of Merkle trees in verifying matrix data. In this method, the matrix initially undergoes processes of consecutive and non-overlapping slicing. Consecutive slicing refers to continuously dividing the original matrix by rows or columns to form multiple smaller submatrices. Non-overlapping slicing involves splitting the original matrix into multiple smaller, non-overlapping submatrix blocks. This slicing approach reduces the amount of data needed to be processed in a single verification while maintaining data integrity. Mathematically, matrix splitting can be expressed as follows:

$$M_{ij} = Slice(M, i, j) \tag{9}$$

where M is the original matrix, M_{ij} is the resulting submatrix from slicing, $Slice$ is the splitting function, and i and j represent the starting indices of rows and columns for slicing, respectively. During non-overlapping slicing, a specific slicing strategy is used to ensure that each submatrix block can be independently verified, crucial for the parallelization and efficiency of computation. The choice of this slicing strategy depends on the size of the submatrices and the structural characteristics of the original matrix. After slicing, each submatrix block is assigned a unique hash value, then combined using the improved Merkle tree structure. Unlike traditional Merkle trees, the improved Merkle tree allows nodes within the tree to represent submatrix blocks rather than just single data points. For a submatrix block M_{ij}, its hash value is represented as follows:

$$h_{M_{ij}} = H(M_{ij}) \tag{10}$$

where H is the hash function. In constructing the tree, hash values of adjacent submatrix blocks are combined to compute their parent hash value, continuing until the root hash value of the tree is calculated. The advantage of this design is that when verifying the integrity of matrix data, it is not necessary to verify every element of the entire matrix, but only the hash values of the submatrix blocks and the root hash value. This significantly reduces the computational load in the verification process and enhances the parallelization of verification. In particular, when dealing with large-scale financial data, this method significantly improves verification efficiency.

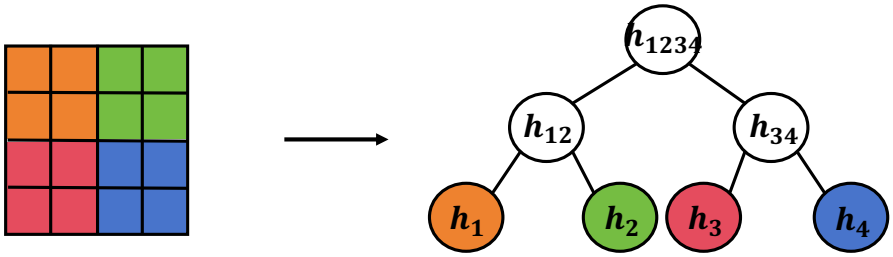

Figure 4. Illustration of the MatrixSplit method. Initially, the matrix is divided into multiple continuous submatrices using a continuous slicing approach, with each submatrix containing a portion of consecutive rows or columns from the original matrix. Subsequently, the disjoint slicing technique is applied to further partition the matrix into non-overlapping smaller matrix blocks, each composed of a specific region from the original matrix. These matrix blocks are assigned individual hash values, serving as unique identifiers for leaf nodes in the modified Merkle tree. By constructing the modified Merkle tree, effective validation of individual matrix blocks can be performed without the need to validate the entire integrity of the matrix. This approach is particularly valuable in large-scale data processing scenarios, significantly improving validation efficiency while supporting parallel processing and incremental updates.

In the financial computation tasks of this paper, the application of the matrix splitting verification method offers distinct advantages. Since financial data are often multi-dimensional, such as stock price data matrices encompassing time, prices, volumes, and other dimensions, directly applying traditional Merkle trees leads to efficiency issues. By adopting the matrix splitting verification method, the integrity and correctness of data can be verified more quickly, ensuring the reliability and security of the financial computation model. Additionally, this method supports incremental updates and rapid re-computation; when matrix data change, only the hash values of the affected submatrix blocks and their related path need to be recalculated, without processing the entire dataset.

3.3.4. Verifiable Secure Transformer Inference Model

Apart from the verifiable computation module based on the Merkle tree, another core component is the verifiable secure Transformer model. This module aims to efficiently process financial data while providing a mechanism to verify the data processing procedure,

ensuring data security and integrity throughout the computation process. The verifiable secure Transformer model, in its design, incorporates the fundamental architecture of the original Transformer model, consisting of multiple self-attention layers and feed-forward networks, with added verifiability. Each layer of the model generates an intermediate output, which is used not only for computing the input of the next layer but also for generating the Merkle tree, facilitating comprehensive data integrity verification.

This self-attention mechanism allows the model to consider the influence of all other inputs while processing each input, thereby capturing long-distance data dependencies. Following the self-attention layer is the feed-forward network, which further processes the output of the self-attention layer. Feed-forward networks typically consist of two linear transformations with an activation function, such as ReLU [31] or GELU [32], in between, and are mathematically expressed as follows:

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{11}$$

Here, W_1, W_2, and b_1, b_2 represent the weights and bias terms of the network layer, respectively. To achieve verifiability, an additional step is added to the output of each layer: the computation of its hash value, which is then integrated as a node in the Merkle tree. This means that at every layer of the model, outputs are produced not only for computation in the subsequent layer but also for validating data integrity. In the financial computation tasks addressed in this paper, the verifiable secure Transformer model offers numerous advantages. Firstly, it improves the efficiency and accuracy of financial data processing, as the Transformer model is capable of effectively handling long sequence dependencies. Secondly, the model enhances the security of data processing, as any unauthorized alteration to the data will be reflected in the root hash value of the Merkle tree and can be rapidly detected. Finally, this design supports the scalability and flexibility of the model, as it allows for the addition of new data processing layers or adjustments to existing ones while maintaining the validity of existing data verifications.

3.3.5. Merkle Loss Function

In this paper, a novel loss function, the Merkle loss function, is introduced. This function is a significant supplement and expansion of traditional loss functions, designed to integrate the data verification mechanism of Merkle trees with the optimization process of deep learning models. The Merkle loss function is conceptualized to consider not only prediction errors at every step of model training but also the integrity and security of data. Traditional loss functions, such as mean squared error (MSE) or cross-entropy, primarily focus on the model's predictive performance, i.e., the discrepancy between predicted and actual values. The Merkle loss function, however, introduces an additional term that measures the consistency between the model output and the root hash value of the Merkle tree. This new loss function is represented as follows:

$$L_{\text{Merkle}} = L_{\text{pred}} + \lambda \times L_{\text{merkle}} \tag{12}$$

where L_{pred} is the traditional prediction loss term, such as MSE or cross-entropy, L_{merkle} is the loss term related to the root hash value of the Merkle tree, and λ is a hyperparameter balancing the importance of the two. The above approach is designed to conform to the general expression of the loss function. In practice, by setting λ to a very large number (e.g., 10,000), it allows the model to immediately stop training upon detecting data integrity issues, similar to a penalty term in optimization algorithms.

3.4. Experiment Design

The design of the experiment is a crucial component in demonstrating the effectiveness of the Merkle-Transformer model. The experiment aims to showcase the advantages of our model in processing financial data and verify its capability to ensure data security. To ensure the accuracy and reliability of the experimental results, a standard dataset division

strategy is adopted. Specifically, the dataset is divided into training, validation, and test sets. The training set is used for the learning process of the model, the validation set for adjusting model parameters and hyperparameters, and the test set for the final evaluation of model performance. Typically, the dataset is divided into a 70%, 15%, and 15% ratio, although the actual proportions may vary based on the size and characteristics of the dataset. This division ensures the model's generalization on unseen data and avoids overfitting.

In the experiments, multiple baseline models were selected for comparison, including Transformer [23], BERT [33], BERT-wwm [34], RoBERTa [35], and SpanBERT [36]. These models have been widely applied in the field of natural language processing and have demonstrated excellent performance across various tasks. Their selection aims to showcase the performance improvement of the Merkle-Transformer model relative to the current state-of-the-art technologies. The Transformer model, as the pioneer of self-attention mechanisms, serves as the foundation for our comparison. The BERT model, known for its powerful contextual understanding capabilities, has established new benchmarks in multiple NLP tasks. BERT-wwm, RoBERTa, and SpanBERT represent further advancements based on BERT, enhancing model performance through different pre-training strategies and architectural optimizations. The choice of these models as baselines is motivated by their representation of the current level of advancement in natural language processing technology and their exceptional ability to handle sequential data. By comparing the Merkle-Transformer model with these baselines, a fair assessment of our model's performance can be conducted, especially in the specific domain of financial data processing.

The Adam optimizer [37] was chosen for optimization. Adam, an adaptive learning rate optimization algorithm, combines the advantages of momentum and RMSprop optimization methods. It adjusts the learning rate for each parameter based on its update history, making it particularly suitable for handling large-scale and complex datasets. Additionally, the Adam optimizer has been proven effective in various deep learning tasks, thus being the preferred choice. Hyperparameter settings are based on an understanding of the problem and preliminary experimental results, including learning rate, batch size, and the number of training epochs. The learning rate starts high and is gradually reduced during training to refine the adjustment of model weights. The batch size depends on the dataset size, model complexity, and hardware resource limitations. The number of training epochs needs to be sufficient to ensure the model can adequately learn patterns in the data but not so numerous as to cause overfitting.

Through such an experimental design, a comprehensive evaluation of the Merkle-Transformer model's performance in secure financial computation tasks can be conducted, allowing for a fair comparison with current advanced models. By comparing experimental results, it is expected to demonstrate that the Merkle-Transformer not only provides excellent predictive performance in processing financial data but also enhances the security of the data processing process.

3.5. Evaluation Index

In our research, a series of quantitative evaluation metrics were employed to comprehensively assess model performance, reflecting various aspects of model functionality, particularly in prediction accuracy and data integrity protection. Precision is used to measure the proportion of true positives among all samples predicted as positive by the model. High precision directly relates to the credibility of the model in practical applications. The formula for precision is as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

where TP (true positive) represents the number of samples correctly predicted as positive, and FP (false positive) denotes the number of samples incorrectly predicted as positive.

Recall measures the proportion of actual positive samples identified by the model, reflecting the model's ability to capture important events. Recall is particularly crucial in

the field of financial security, as missing any real fraudulent event could lead to significant economic losses. The formula for recall is as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

where FN (false negative) refers to the number of actual positive samples incorrectly predicted as negative.

Accuracy is an indicator of the overall correctness of the model's predictions, reflecting the proportion of correctly predicted samples in the total sample set. This metric provides an intuitive overview of model performance and is calculated as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

where TN (true negative) represents the number of samples correctly predicted as negative.

For regression tasks like stock price prediction, the coefficient of determination R^2 is used to assess the quality of the model's predictions. R^2 measures the degree of correlation between the model's predictions and actual values, with values closer to 1, indicating stronger predictive ability. The formula for R^2 is as follows:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2} \tag{16}$$

where y_i represents the actual value of the ith sample, $\hat{y}_i$ is the model's prediction for the ith sample, and $\bar{y}$ is the average of all actual values.

These evaluation metrics comprehensively assess the model's performance from multiple dimensions. Precision and recall are particularly applicable to classification tasks, such as financial fraud detection, assisting in evaluating the model's ability to identify genuine risk events. Accuracy provides an overall performance evaluation, aiding in understanding the model's performance across the entire dataset. The coefficient of determination R^2, as a regression task metric, reflects the degree of fit between the model's predictions and actual outcomes, which is crucial for financial time series analysis like stock price prediction.

4. Results and Discussion

In Section 4.1, we cover the performance of the model in financial behavior detection and stock price prediction tasks, the verification of its security, the analysis of system overhead, as well as the limitations and prospects of the model. Initially, in the performance analysis section, an in-depth examination of the Merkle-Transformer model's performance in financial behavior detection tasks is conducted. By comparing it with various baseline models, its precision, recall, and F1 score in identifying fraudulent transactions and money laundering activities are assessed. The experimental results demonstrate that the Merkle-Transformer, considering data integrity and security while processing financial sequential data, exhibits commendable performance in detecting financial fraud, particularly excelling in reducing false positives compared to traditional models.

In Section 4.2, we discuss the current limitations of the Merkle-Transformer model, such as scalability issues when processing extremely large datasets, and ways to further enhance the model's security and reduce overhead. Furthermore, future research directions are proposed, including optimizing the model structure, improving security mechanisms, and exploring potential applications in other financial tasks.

4.1. Performance Analysis

The objective of this experiment is to validate and showcase the performance of the Merkle-Transformer model in processing financial data, especially in comparison with current mainstream deep learning models. Through this experiment, the superiority of the

Merkle-Transformer in key performance metrics such as precision, recall, accuracy, and F1 score is intended to be demonstrated, thereby proving its potential application in the field of financial security. The results in Table 1 indicate that the Merkle-Transformer outperforms all other models in all evaluation metrics, followed by RoBERTa, BERT, Transformer, BERT-wwm, and SpanBERT. These results reflect the performance differences of various models in financial behavior detection tasks, where the Merkle-Transformer's superior performance is attributed to its unique design and algorithms.

Table 1. Financial computation performance.

Model	Precision	Recall	Accuracy	F1 Score
Merkle-Transformer	0.95	0.97	0.96	0.96
RoBERTa	0.90	0.92	0.91	0.91
BERT	0.87	0.89	0.88	0.88
Transformer	0.85	0.87	0.86	0.86
BERT-wwm	0.83	0.85	0.84	0.84
SpanBERT	0.81	0.84	0.83	0.82

Firstly, the Merkle-Transformer model integrates the data integrity verification mechanism of Merkle trees with the efficient data processing capabilities of the Transformer model. The Merkle tree structure provides a mechanism for verifying data integrity, ensuring that data remains unaltered or undamaged during processing, which is especially important in financial data processing. The Transformer model, known for its efficient self-attention mechanism, excels in handling complex sequential data. This combination not only improves the accuracy of data processing but also enhances the robustness of the model when facing challenges unique to financial data. Models like RoBERTa, BERT, and SpanBERT, although performing well in natural language processing tasks, are limited in their performance in financial data processing due to their original design focus. These models primarily concentrate on semantic understanding of text data, not on the specific characteristics of financial data. Additionally, they lack a mechanism for verifying data integrity, an important consideration in financial data processing. The Transformer model, as the originator of the self-attention mechanism, has certain advantages in processing sequential data but its application in the financial domain is limited by its generality. Without customized processing for the specific characteristics of financial data, it may not achieve optimal performance in financial behavior detection tasks. The BERT-wwm model, despite its advantages in processing Chinese text, is limited in this specific task of financial behavior detection due to its understanding of financial-specific contexts.

The superior performance of the Merkle-Transformer model can be attributed to its integration of the data integrity verification mechanism of Merkle trees with the efficient data processing capabilities of the Transformer model. In contrast, models like RoBERTa, BERT, Transformer, and BERT-wwm, despite their exemplary performance in natural language processing, showed slightly inferior results in this specific task of stock price prediction. This is mainly because these models, while capable of processing sequential data, lack customized processing abilities for financial characteristics. For instance, they might be insufficient in capturing complex market trends and subtle financial signals, resulting in lesser predictive performance compared to the Merkle-Transformer, which is specifically designed for financial data. The SpanBERT model performed the worst across all metrics, possibly due to its focus on the span representation of text during pre-training, rather than the time dependencies in sequential data, which may not be optimal for dealing with highly time-sensitive data like stock prices.

4.2. Security and Performance Verification

In this section, the security features of the Merkle-Transformer model and its performance across various hardware platforms are extensively evaluated. Additionally, in combination with the Merkle-Transformer verification method, the impact of various run-

time configurations on the model's performance is further explored. These experiments are designed to not only validate the efficiency and accuracy of the model in performing tasks such as financial behavior detection and stock price prediction but also to ensure the verifiability and security of the model's operations.

As illustrated in Table 2, tests across multiple hardware platforms reveal that even on lower-end devices like Raspberry Pi and NVIDIA Jetson Nano, the Merkle-Transformer model maintains stable performance while ensuring integrity verification throughout the computation process. Its performance on the GPU and Apple M1 is particularly noteworthy, indicating the model's effective utilization of the high-performance computational resources of modern hardware platforms. The data also show that multi-threading execution significantly improves inference throughput while keeping the CPU load at a lower level, crucial for applications requiring deployment in resource-constrained environments.

Table 2. Performance comparison between local inferences and Merkle-Transformer running hardware platforms.

Hardware	Concurrency	CPU Load	Latency	Throughput
RPi	Baseline	Mid	8.3 s	157
RPi	2× Concurrency	High	3.1 s	618
RTX 3070	Baseline	Low	3.0 s	482
RTX 3070	2× Concurrency	Mid	4.1 s	1107
Apple M1	Baseline	Low	3.9 s	531
Apple M1	2× Concurrency	Mid	6.2 s	1209
Jetson Nano	Baseline	Low	21.4 s	125

In Table 3 and Figure 5, the Merkle-Transformer shows significant advantages over other verification methods like Slalom [38] in terms of communication and computation overhead, storage requirements, and potential error detection. This result confirms the efficiency of our verification method in ensuring the integrity of machine learning inferences.

Table 3. Qualitative comparisons between verification methods for checking inference integrity.

Model	Communication	Computation	Concurrency	FNE
Slalom [38]	High	High	Low	Low
MLP	Low	High	Low	High
ResNet	Mid	High	Low	High
Merkle-Transformer	Low	Mid	High	Low

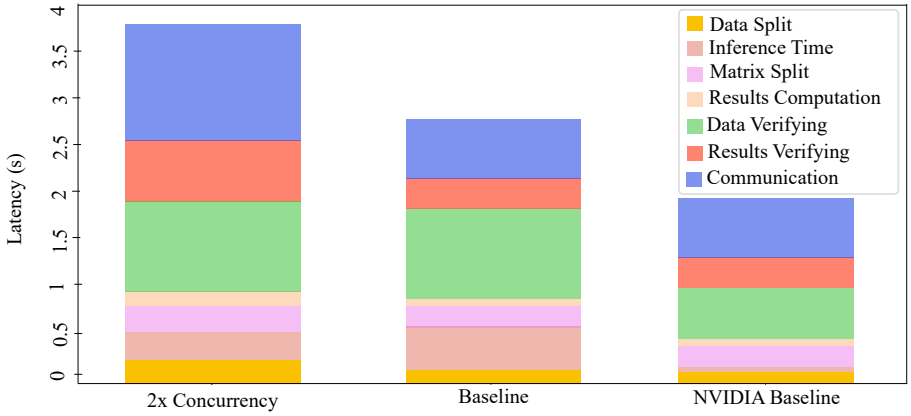

Figure 5. Latency detail of Merkle-Transformer on different hardware platforms.

Furthermore, the impact of different runtime configurations of MatrixSplit on performance is explored. As indicated in Table 4, by optimizing communication and computation,

latency can be significantly reduced without sacrificing verification stringency. For example, when low partition of the network instead of the entire work, only a slight performance decrease is observed. This flexibility suggests that our model and verification method can be adjusted according to specific application needs to achieve the best balance between security and performance.

Table 4. Performance impact of various Merkle-Transformer runtime configurations.

Experimental Setup	Communication	Latency	Computational Source
Baseline	5.32 MB	3.13 s	Mid
No Cache	5.32 MB	4.17 s	High
No Compression	8.97 MB	4.33 s	Mid
Low Partition	21.78 MB	5.81 s	High
Low Level	15.69 MB	5.64 s	Low

4.3. Limitations and Future Works

This section aims to candidly assess the current shortcomings of the model and explore potential future research directions. Firstly, although the Merkle-Transformer has exhibited excellent performance in financial behavior detection and stock price prediction, its effectiveness in processing extremely large datasets has not been thoroughly tested. In practical applications, financial institutions may need to handle enormous volumes of data, posing higher demands on the model's processing capability and verification efficiency. Therefore, the model's performance and scalability in handling big data are critical areas for future focus. Secondly, despite the model's exemplary performance in security verification, new security threats may arise in practical deployment. For instance, adversarial attacks are an active research topic in the field of machine learning, and attackers might develop novel methods to evade the security mechanisms of the Merkle-Transformer. Consequently, continuous research and improvement of the model's security features to counter evolving threats are necessary. Furthermore, the performance of the Merkle-Transformer is also dependent on hardware platform configurations, as our experiments have shown. While the model maintains good performance across multiple hardware platforms, optimizing the model to reduce computational and storage overheads on resource-constrained devices, such as edge computing scenarios, is another key point for future research. This includes model compression, quantization, and other optimization processes.

Future work should also include testing the generality of the model and verifying the potential application of the Merkle-Transformer in other types of financial tasks, such as credit scoring and market risk management. Additionally, exploring the model's performance in multi-task learning and cross-domain learning is a promising area, which would further enhance the model's practicality and flexibility. From a technical research perspective, the mathematical foundation and theoretical framework of the Merkle-Transformer also warrant further exploration. For instance, mathematical optimization of the combination of Merkle trees and Transformers, algorithmic improvements, and mathematical proofs of data integrity verification mechanisms are potential directions for in-depth future research. This would not only help improve the model's performance but also deepen our understanding of the model's working mechanism.

Furthermore, implementing Merkle trees in various types of deep neural networks to ensure data integrity and security is a challenging yet highly promising task. Let us first consider the case of convolutional neural networks (CNNs) [31,39]. In CNNs, we can compute the hash values of features at each layer, from convolutional layers to pooling layers and then to fully connected layers, to construct corresponding Merkle tree nodes. This helps verify the integrity of features at each layer and detects data tampering or loss when necessary. Another common deep neural network architecture is the residual network (ResNet) [40]. In ResNet, we can calculate hash values within each residual block and add them to the respective Merkle tree. This ensures the consistency of features within each block while maintaining the smooth flow of information throughout the network. Lastly,

for multi-layer perceptrons (MLPs) [41], we can compute hash values before each hidden layer and the output layer to build corresponding Merkle tree nodes. This aids in verifying the data integrity at each layer, especially when highly trustworthy predictions are needed. By implementing Merkle trees in different types of DNN structures, we can enhance the protection of data integrity by the model. However, it is crucial to carefully balance computational costs and performance overhead to ensure practicality and feasibility.

In summary, while the Merkle-Transformer model has shown significant potential in processing financial data, it also faces many challenges and limitations. We hope that future research will address these limitations, continuously optimizing and refining the model to better serve the needs of the financial industry, and contribute new ideas and frameworks to the research of machine learning in the domain of financial security.

5. Conclusions

In this study, a novel model for financial data processing is presented, which provides a secure and efficient solution for financial data analysis by integrating the data integrity verification mechanism of Merkle trees with the high processing capability of the Transformer model. The research conducted is significant not only for enhancing the accuracy and security of financial data processing but also for propelling development and innovation in the field of fintech.

The performance of the Merkle-Transformer model on tasks such as financial behavior detection and stock price prediction was validated through a series of experiments. It was found that the Merkle-Transformer outperforms other baseline models, including RoBERTa, BERT, Transformer, BERT-wwm, and SpanBERT, across key performance metrics such as precision, recall, accuracy, and the F1 score. Notably, in the task of stock price prediction, the Merkle-Transformer achieved or approached scores above 0.9 across all evaluation metrics, demonstrating its robust predictive power and acute capture of market dynamics. These results affirm the efficacy and superior performance of the Merkle-Transformer in financial applications. Furthermore, we investigated the performance of the Merkle-Transformer, combined with the Merkle-Transformer verification method across various hardware platforms, including experiments on Raspberry Pi, GPU, Apple, and NVIDIA Jetson. The results show that the Merkle-Transformer maintains good performance even on resource-constrained devices while ensuring the integrity and security of the data processing through the Merkle-Transformer method.

The outcomes of this study have profound implications for investors and analysts. On the one hand, novel financial data processing models can enhance data accuracy and security, thereby instilling confidence in investors and enabling them to make more assured investment decisions. On the other hand, the introduction of this model can deter unethical conduct by analysts attempting fraud, thereby upholding the integrity and fairness of financial markets. Firstly, the approach presented in this paper employs the data integrity verification mechanism of Merkle trees to securely validate financial data. Secondly, the method outlined in this paper leverages Transformer models for the swift processing of large volumes of financial data, enabling real-time analysis and decision-making, which is critical for both investors and analysts. Efficient data processing not only accelerates decision-making but also facilitates the timely capture of market dynamics, thereby enhancing the competitive edge of investors and analysts. Furthermore, experimental results in this paper demonstrate that the Merkle-Transformer excels in stock price prediction tasks, with all evaluation metrics reaching or approaching scores of 0.9 or higher, indicating its robust predictive capabilities and keen ability to capture market dynamics. Additionally, this model exhibits broad applicability, running smoothly on both high-end servers and lower-end devices. The core contribution of this article lies in the proposition of a new financial data processing model that not only considers the efficiency of data processing but also introduces a mechanism for data security verification. By incorporating Merkle trees, the accuracy of data processing has been enhanced, and the model's resistance to data tampering has been strengthened, which is crucial for handling

sensitive financial information. Additionally, the model is adaptable to various hardware configurations, ensuring its applicability and stability in different environments. The research and development of the Merkle-Transformer model offer a fresh perspective and methodology for financial data processing. In today's era where data security is increasingly valued, it is believed that the findings of this study will provide direct technical support and assistance to the financial industry and offer new research directions and foundations for researchers in the field of fintech. Looking ahead, there is an anticipation that the Merkle-Transformer will see wider application in financial data processing scenarios and will continue to be optimized and refined to achieve greater security and efficiency.

Author Contributions: Conceptualization, X.W., X.Y. and C.L.; Methodology, X.W., W.L. and W.Z.; Software, X.W., W.L., W.Z. and X.Y.; Validation, W.L.; Formal analysis, Q.L.; Investigation, Y.H.; Resources, Y.H. and Z.L.; Data curation, W.Z., Y.H., Z.L. and Q.L.; Writing—original draft, X.W., W.L., W.Z., Y.H., Z.L., Q.L., X.Y., Y.Y. and C.L.; Writing—review & editing, Y.Y. and C.L.; Visualization, Z.L., Q.L. and X.Y.; Supervision, Y.Y.; Project administration, C.L.; Funding acquisition, C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 61202479.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Yin, Y.; Wang, L. Research on personal financial data storage medium system based on fractional order calculus encryption algorithm. *Chaos Solitons Fractals* **2020**, *131*, 109459. [CrossRef]
2. Zhou, Y. Regional Financial Data Processing Based on Distributed Decoding Technology. *Secur. Commun. Netw.* **2022**, *2022*, 1043140. [CrossRef]
3. Wu, Y.; Tie, G.; Yu, Y.; Li, J.; Song, J. EBSS: A secure blockchain-based sharing scheme for real estate financial credentials. *World Wide-Web-Internet Web Inf. Syst.* **2023**, *26*, 1599–1624. [CrossRef]
4. Zhao, J. Efficiency of corporate debt financing based on machine learning and convolutional neural network. *Microprocess. Microsyst.* **2021**, *83*, 103998. [CrossRef]
5. Li, C.; Zhang, G.; Mao, X.; Zhang, J.; Xing, C. Multi-Chain Model and Cross-Chain Communication Protocol for Financial Transactions. In Proceedings of the 2022 IEEE 22nd International Conference on Software Quality, Reliability, and Security Companion (QRS-C), Guangzhou, China, 5–9 December 2022; pp. 547–551. [CrossRef]
6. Du, P.; Liu, Y.; Li, Y.; Yin, H. EthMB plus: A Tamper-Proof Data Query Model Based on B plus Tree and Merkle Tree. In *Communications in Computer and Information Science, Proceedings of the 5th CCF China Blockchain Conference (CBCC), Wuxi, China, 23–25 December 2022*; Sun, Y., Cai, L., Wang, W., Song, X., Lu, Z., Eds.; Springer: Singapore, 2022; Volume 1736, pp. 49–59. [CrossRef]
7. Bailey, B.; Sankagiri, S. Merkle Trees Optimized for Stateless Clients in Bitcoin. In *Lecture Notes in Computer Science, Proceedings of the Conference on Financial Cryptography and Data Security (FC), Virtual Event, 1–5 March 2021*; Bernhard, M., Bracciali, A., Gudgeon, L., Haines, T., KlagesMundt, A., Matsuo, S., Perez, D., Sala, M., Werner, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 12676, pp. 451–466. [CrossRef]
8. Acheampong, F.A.; Nunoo-Mensah, H.; Chen, W. Transformer models for text-based emotion detection: A review of BERT-based approaches. *Artif. Intell. Rev.* **2021**, *54*, 5789–5829. [CrossRef]
9. Seyyar, Y.E.; Yavuz, A.G.; Unver, H.M. An Attack Detection Framework Based on BERT and Deep Learning. *IEEE Access* **2022**, *10*, 68633–68644. [CrossRef]
10. Lian, M.; Li, J. Financial product recommendation system based on transformer. In Proceedings of the 4th IEEE Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Virtual Event, 12–14 June 2020; Xu, B., Mou, K., Eds.; IEEE: Piscataway, NJ, USA, 2020; pp. 2547–2551. [CrossRef]
11. Zhang, Q.; Qin, C.; Zhang, Y.; Bao, F.; Zhang, C.; Liu, P. Transformer-based attention network for stock movement prediction. *Expert Syst. Appl.* **2022**, *202*, 117239. [CrossRef]
12. Chen, Y.; Dai, H.; Yu, X.; Hu, W.; Xie, Z.; Tan, C. Improving Ponzi Scheme Contract Detection Using Multi-Channel TextCNN and Transformer. *Sensors* **2021**, *21*, 6417. [CrossRef]
13. Kim, G.; Kim, M.; Kim, B.; Lim, H. CBITS: Crypto BERT Incorporated Trading System. *IEEE Access* **2023**, *11*, 6912–6921. [CrossRef]

14. Karagiannis, I.; Mavrogiannis, K.; Soldatos, J.; Drakoulis, D.; Troiano, E.; Polyviou, A. Blockchain Based Sharing of Security Information for Critical Infrastructures of the Finance Sector. In *Lecture Notes in Computer Science, Proceedings of the 24th European Symposium on Research in Computer Security (ESORICS), Luxembourg, 23–27 September 2019*; Fournaris, A., Athanatos, M., Lampropoulos, K., Ioannidis, S., Hatzivasilis, G., Damiani, E., Abie, H., Ranise, S., Verderame, L., Siena, A., et al., Eds.; Springer: Cham, Switzerland, 2020; Volume 11981, pp. 226–241. [CrossRef]
15. Alegria, A.V.; Loayza, J.L.M.; Montoya, A.N.; Armas-Aguirre, J. Method of Quantitative Analysis of Cybersecurity Risks Focused on Data Security in Financial Institutions. In Proceedings of the 2022 17th Iberian Conference on Information Systems and Technologies (CISTI), Madrid, Spain, 22–25 June 2022; Rocha, A., Bordel, B., Penalvo, F., Goncalves, R., Eds.; IEEE: Piscataway, NJ, USA, 2022.
16. Ahmed, M.R.; Meenakshi, K.; Obaidat, M.S.; Amin, R.; Vijayakumar, P. Blockchain Based Architecture and Solution for Secure Digital Payment System. In Proceedings of the ICC 2021-IEEE International Conference on Communications (ICC 2021), Virtual Event, 14–23 June 2021. [CrossRef]
17. Sheeba, T.B.; Hemanth, S.; Devaraj, V.; Arularasan, A.; Gopianand, M. Digital Hash Data Encryption for IoT Financial Transactions using Blockchain Security in the Cloud. *Int. J. Recent Innov. Trends Comput. Commun.* **2023**, *11*, 129–134. [CrossRef]
18. Alfrhan, A.; Moulahi, T.; Alabdulatif, A. Comparative study on hash functions for lightweight blockchain in Internet of Things (IoT). *Blockchain Res. Appl.* **2021**, *2*, 100036. [CrossRef]
19. Yu, M.; Sahraei, S.; Li, S.; Avestimehr, S.; Kannan, S.; Viswanath, P. Coded merkle tree: Solving data availability attacks in blockchains. In Proceedings of the International Conference on Financial Cryptography and Data Security, Kota Kinabalu, Malaysia, 10–14 February 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 114–134.
20. Zhu, H.; Guo, Y.; Zhang, L. An improved convolution Merkle tree-based blockchain electronic medical record secure storage scheme. *J. Inf. Secur. Appl.* **2021**, *61*, 102952. [CrossRef]
21. Haque, A.; Rahman, M. Blockchain technology: Methodology, application and security issues. *arXiv* **2020**, arXiv:2012.13366.
22. Ding, Q.; Wu, S.; Sun, H.; Guo, J.; Guo, J. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In Proceedings of the IJCAI, Virtual Event, 7–15 January 2020; pp. 4640–4646.
23. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1761–1768.
24. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
25. Moghar, A.; Hamiche, M. Stock market prediction using LSTM recurrent neural network. *Procedia Comput. Sci.* **2020**, *170*, 1168–1173. [CrossRef]
26. Yang, L.; Li, J.; Dong, R.; Zhang, Y.; Smyth, B. NumHTML: Numeric-Oriented Hierarchical Transformer Model for Multi-task Financial Forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 22 February–1 March 2022; Volume 36, pp. 11604–11612.
27. Li, C.; Qian, G. Stock Price Prediction Using a Frequency Decomposition Based GRU Transformer Neural Network. *Appl. Sci.* **2022**, *13*, 222. [CrossRef]
28. Wang, H.; Zheng, J.; Carvajal-Roca, I.E.; Chen, L.; Bai, M. Financial Fraud Detection Based on Deep Learning: Towards Large-Scale Pre-training Transformer Models. In Proceedings of the China Conference on Knowledge Graph and Semantic Computing, Shenyang, China, 24–27 August 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 163–177.
29. Huo, H.; Guo, J.; Yang, X.; Lu, X.; Wu, X.; Li, Z.; Li, M.; Ren, J. An Accelerated Method for Protecting Data Privacy in Financial Scenarios Based on Linear Operation. *Appl. Sci.* **2023**, *13*, 1764. [CrossRef]
30. Fernández, A.; Garcia, S.; Herrera, F.; Chawla, N.V. SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *J. Artif. Intell. Res.* **2018**, *61*, 863–905. [CrossRef]
31. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
32. Hendrycks, D.; Gimpel, K. Gaussian error linear units (gelus). *arXiv* **2016**, arXiv:1606.08415.
33. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
34. Cui, Y.; Che, W.; Liu, T.; Qin, B.; Yang, Z. Pre-training with whole word masking for chinese bert. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 3504–3514. [CrossRef]
35. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
36. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]
37. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
38. Tramer, F.; Boneh, D. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. *arXiv* **2018**, arXiv:1806.03287.
39. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.

40. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 16 June–1 July 2016.
41. Sreedharan, M.; Khedr, A.M.; El Bannany, M. A multi-layer perceptron approach to financial distress prediction with genetic algorithm. *Autom. Control Comput. Sci.* **2020**, *54*, 475–482. [CrossRef]