applied sciences

MDPI

*Article*

# Finsformer: A Novel Approach to Detecting Financial Attacks Using Transformer and Cluster-Attention

**Hao An †, Ruotong Ma †, Yuhan Yan, Tailai Chen, Yuchen Zhao, Pan Li, Jifeng Li, Xinyue Wang, Dongchen Fan and Chunli Lv ***

China Agricultural University, Beijing 100083, China
* Correspondence: lvcl@cau.edu.cn
† These authors contributed equally to this work.

**Abstract:** This paper aims to address the increasingly severe security threats in financial systems by proposing a novel financial attack detection model, Finsformer. This model integrates the advanced Transformer architecture with the innovative cluster-attention mechanism, dedicated to enhancing the accuracy of financial attack behavior detection to counter complex and varied attack strategies. A key innovation of the Finsformer model lies in its effective capture of key information and patterns within financial transaction data. Comparative experiments with traditional deep learning models such as RNN, LSTM, Transformer, and BERT have demonstrated that Finsformer excels in key metrics such as precision, recall, and accuracy, achieving scores of 0.97, 0.94, and 0.95, respectively. Moreover, ablation studies on different feature extractors further confirm the effectiveness of the Transformer feature extractor in processing complex financial data. Additionally, it was found that the model's performance heavily depends on the quality and scale of data and may face challenges in computational resources and efficiency in practical applications. Future research will focus on optimizing the Finsformer model, including enhancing computational efficiency, expanding application scenarios, and exploring its application on larger and more diversified datasets.

**Keywords:** financial attack detection; Transformer architecture; cluster-attention mechanism; deep learning in finance



**Citation:** An, H.; Ma, R.; Yan, Y.; Chen, T.; Zhao, Y.; Li, P.; Li, J.; Wang, X.; Fan, D.; Lv, C. Finsformer: A Novel Approach to Detecting Financial Attacks Using Transformer and Cluster-Attention. *Appl. Sci.* **2024**, *14*, 460. https://doi.org/10.3390/app14010460

Academic Editors: Shahadat Uddin, Tasadduq Imam and Sisira Colombage

Received: 4 December 2023
Revised: 31 December 2023
Accepted: 31 December 2023
Published: 4 January 2024



## 1. Introduction

With the rapid development of financial technology, financial systems have become one of the primary targets of cyberattacks [1–3]. The vast amount of sensitive transactions and customer information involved in financial institutions makes them particularly vulnerable to various cyberattacks, especially targeted financial fraud. These attacks can cause significant economic losses and undermine user trust and the stability of financial markets. Therefore, developing efficient and accurate attack detection systems is crucial for safeguarding financial security. Diaz-Verdejo Jesus et al. conducted experimental studies on the detection rates of three standard pre-configurations of SIDS in the context of URI web attacks and proposed an effective method to reduce false positives by disabling complete rule sets of signatures. However, WAF is only a subset of the detection capabilities of SIDS, and it is unclear whether their model results hold true in different types of attacks [4]. Saez-de-Camara Xabier et al. found that traditional IT security mechanisms, such as signature-based intrusion detection and defense systems, are difficult to integrate [5]. Abdulganiyu Oluwadamilare Harazeem et al. discovered that current intrusion detection systems (IDSs) identify unknown attacks, but their false positive rate is still high [6]. Yang et al. indicated that traditional IDSs are divided into analysis-based or signature-based; however, analysis-based IDSs face the significant challenge of manually labeling security-related data. To address this, they proposed a weakly supervised learning algorithm-based IDS model training scheme [7].

Traditional attack detection methods, such as rule-based and signature-based systems, although effective in specific scenarios, usually cannot adapt to rapidly changing attack patterns and complex financial data environments [8]. With the advancement of artificial intelligence technology, machine learning, especially deep learning, has shown great potential in the field of attack detection. Deep learning methods, due to their advantages in feature learning and pattern recognition, have been proven to be particularly effective in handling complex and high-dimensional data.

Elsaeidy, Asmaa A. et al. developed a four-layer deep neural network to detect replay attacks in smart cities and applied it in real-world scenarios, showing that deep learning models can detect normal and attack behaviors with high accuracy [9]. Rashi Md. Mamunur et al. used adversarially retrained samples to reinforce IDS models, ultimately increasing accuracy to over 99%, but the model is prone to overfitting and needs consideration on how to reduce sensitivity to attack behaviors [9]. Waqar Muhammad et al. proposed a deep-learning-based malware detection model for Android-based IoT (AIoT) devices to prevent various malware attacks, achieving an accuracy of 99.87% [10]. Sandouka Soha B. et al. combined EfficientNet with generative adversarial networks (GANs) for fingerprint presentation attack detection (PAD) and validated the proposed method on the public LivDet2015 dataset, showing that the proposed method outperforms other CNN models [11]. Alshingiti Zainab et al. used LSTM, CNN, and LSTM-CNN deep learning methods to detect phishing websites, with the final results showing that CNN performed the best, reaching an accuracy of 99.2% [12]. Ozcan Alper et al. proposed a novel hybrid deep learning model, a hybrid DNN-LSTM model for detecting phishing (URL), and a further developed DNN-BiLSTM model, with research results indicating that the DNN-BiLSTM model's accuracy is higher than the DNN-LSTM model, achieving 98.79% and 99.21% accuracy on the provided datasets, although some noisy instances may affect model performance in actual use [13]. Afzal Sara et al. proposed a hybrid deep learning method named URLdeepDetect for detecting malicious URLs, ultimately achieving 98.3% accuracy [14]. Pastor Antonio et al. deployed a cryptomining scenario to train machine learning models for detecting malicious attacks on digital currency mining, achieving the capability to detect cryptomining attacks even in encrypted states [15]. Wang et al. proposed a deep-learning-based system for the development of decentralized financial (DeFi) attack detection, DeFiScanner, with experimental results showing a true positive rate of 91% [16]. Alkhatib I Khalid et al. proposed a deep-learning-based model for credit card fraud detection—a seven-layer neural network architecture—achieving an area under the ROC curve score of 99.1% [17]. Fursov Ivan et al. proposed a black-box attack scenario for financial institutions' transaction records, using adversarial training samples to test the robustness of models, aiding financial institutions in better utilizing deep learning models for transaction records [18]. Qasaimeh Malik et al. proposed a deep learning algorithm, DNN, to predict network attack patterns, reaching a prediction accuracy of 90.36%, beneficial for banks and other financial institutions to take preemptive security measures [19].

The research encompasses applications of deep learning techniques in detecting and defending against various cyberattacks, particularly in the context of smart cities [20], the internet of things [21], financial institutions [22], and other digital systems. These studies reflect the significant role and challenges faced by deep learning in the field of cybersecurity. In smart cities, challenges such as handling vast amounts of data and ensuring real-time responsiveness of models remain a significant issue [23], along with the need to address overfitting problems. This includes balancing the models' generalization capabilities with their sensitivity to attack behaviors [24]. In the internet of things, the challenge lies in adapting to new types of malware and reducing false positives, as well as enhancing the accuracy and adaptability of detection algorithms to cope with evolving attack methods [25]. The detection of phishing attacks is also a hot research topic [26], yet dealing with large-scale and dynamically changing network environments remains a significant challenge [27]. Additionally, security issues in the financial domain are receiving considerable attention,

with the difficulty residing in rapidly adapting to and predicting new attack patterns within financial environments [28]. These studies indicate that, despite the vast potential of deep learning in the realm of cybersecurity, it also faces numerous challenges, including data processing, model generalization, and adapting to new types of attacks [29].

Finsformer proposed in this paper is a novel financial-system-attack detection model based on the Transformer and cluster-attention mechanism. The Transformer model, having achieved significant success in fields such as natural language processing (NLP), is highly suited for processing financial data sequences due to its powerful sequence-modeling capabilities. Meanwhile, the introduction of the cluster-attention mechanism, a novel aspect of this paper, aims to enhance the model's ability to identify attack patterns by effectively clustering data.

The Transformer model was chosen for this paper because of its several advantages suitable for processing financial data. Firstly, the self-attention mechanism of the Transformer can capture long-distance dependencies in data, which is particularly important when dealing with financial time-series data. Secondly, compared to traditional deep learning models like convolutional neural networks (CNNs) and recurrent neural networks (RNNs), the Transformer offers higher efficiency and flexibility in processing sequence data. Additionally, the parallel processing capability of the Transformer makes it more efficient in handling large-scale data. The introduction of the cluster-attention mechanism is based on the particularity of financial data, which often contains diverse transaction patterns and user behaviors that traditional attention mechanisms may not fully capture. Cluster-attention, by clustering similar data points, can more effectively model and identify different attack patterns, thereby increasing the accuracy and robustness of detection.

In this paper, we first introduce methods based on clustering, traditional machine learning, and deep learning for detecting attack behaviors. These not only provide a theoretical foundation for our model but also set the context for comparative analysis. We then detail the composition of the Finsformer model, including the collection and annotation of datasets, the design of the model architecture, and the detailed planning of experimental design. In the results and discussion section, we showcase the performance of Finsformer on real financial datasets and analyze the role of the cluster-attention mechanism and Transformer feature extractor through ablation studies. In summary, the objective of this paper is to propose and validate a novel financial-system-attack detection model, Finsformer, combining the latest deep learning technology and innovative attention mechanisms, in the hope of contributing to increased accuracy and efficiency in attack detection. Through this work, we aim to provide robust technical support for the financial security field and more effective tools for financial institutions to counteract cyberattacks.

## 2. Related Work

### *2.1. Cluster-Based Attack Detection Methods*

In the field of network security, clustering algorithms are widely utilized for the identification of anomalous behaviors, especially in attack detection. The core idea of these methods involves grouping data points based on similarity, where anomalous data typically do not conform to any pattern of normal behavior, thus enabling effective identification through cluster analysis.

#### 2.1.1. K-Means Clustering

K-means clustering, a widely used clustering algorithm [30], aims to group similar data points together. In the context of financial attack detection, k-means clustering is applied to identify anomalous transaction patterns, such as distinguishing between normal and fraudulent transactions. The fundamental concept of k-means clustering involves minimizing the sum of distances between each point and the centroid of its cluster [31], as shown in Figure 1. First, k initial "means" (in this case k = 3) are randomly generated within the data domain (shown in color). Second, k clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram

generated by the means. Third, the centroid of each of the k clusters becomes the new mean. Finally, steps 2 and 3 are repeated until convergence has been reached.

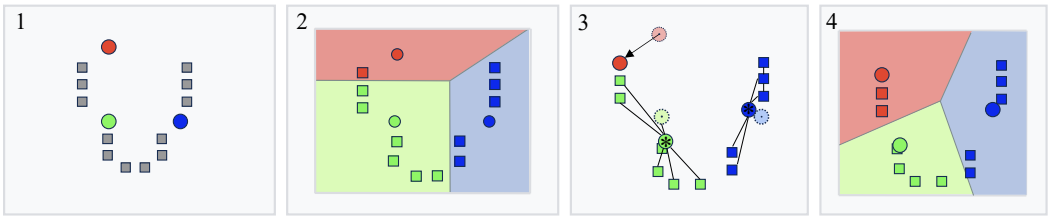


**Figure 1.** Visualization of k-means clustering algorithm.

Mathematically, this is represented by the following optimization problem [30]:

$$\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} |\mathbf{x} - \boldsymbol{\mu}_i|^2 \tag{1}$$

Here, $k$ is the number of clusters, $\mathbf{S} = S_1, S_2, \ldots, S_k$ is the set of clusters, $\mathbf{x}$ represents data points, and $\boldsymbol{\mu}_i$ is the center of cluster $S_i$. The k-means clustering algorithm involves randomly selecting $k$ cluster centers, assigning each data point to the nearest center, and then updating the cluster centers to the mean of the points in each cluster. This process is repeated until convergence criteria are satisfied [32].

#### 2.1.2. Density-Based Clustering

Density-based clustering, such as DBSCAN (density-based spatial clustering of applications with noise), are algorithms capable of identifying clusters of arbitrary shapes [33], as shown in Figure 2.

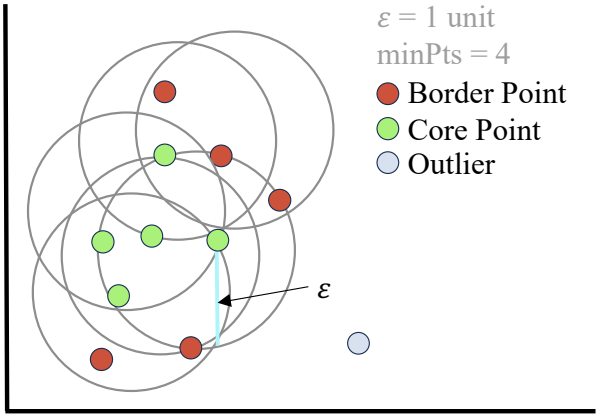


**Figure 2.** Visualization of DBSCAN clustering algorithm.

In financial attack detection, DBSCAN effectively identifies anomalous transaction behaviors in dense areas, unaffected by noise data. The core idea of DBSCAN is to define clusters as sufficiently dense regions [34]. Key parameters of the algorithm include the neighborhood radius $\epsilon$ and the minimum number of points $MinPts$. For each point, if its $\epsilon$-neighborhood contains at least $MinPts$ points, then the point is considered a core point. Clusters are formed based on the concepts of core points and reachability. The mathematical description of DBSCAN is summarized as follows [33]:

$$\text{If } |N_\epsilon(p)| \geq MinPts, \text{ then } p \text{ is a core point} \tag{2}$$

Here, $N_\epsilon(p)$ represents the set of points within the $\epsilon$-neighborhood of point $p$.

#### 2.1.3. Hierarchical Clustering

Hierarchical clustering is a method for creating nested clusters without the need to pre-specify the number of clusters [35]. In financial attack detection, hierarchical clustering is used to explore the inherent structure of data and identify potential anomalous transaction patterns. Hierarchical clustering is divided into two approaches: agglomerative and divisive. Agglomerative hierarchical clustering starts with each data point as a separate cluster and gradually merges the closest clusters; divisive hierarchical clustering, on the other hand, starts with all data as one cluster and progressively splits them into smaller clusters. The cluster distance during hierarchical clustering can be calculated using various metrics, such as single linkage (nearest neighbor), complete linkage (furthest neighbor), or average linkage. For example, the merging condition for agglomerative hierarchical clustering using single linkage can be expressed as

$$d(S_i, S_j) = \min(|\mathbf{x} - \mathbf{y}|) \quad \forall \mathbf{x} \in S_i, \mathbf{y} \in S_j \tag{3}$$

where $d(S_i, S_j)$ is the distance between clusters $S_i$ and $S_j$.

### *2.2. Attack Detection Methods Based on Deep Learning Models*

In the field of financial attack detection, deep learning technologies have gained widespread attention for their excellent feature extraction capabilities and strong pattern recognition performance [1]. Convolutional neural networks (CNNs) [36,37], recurrent neural networks (RNNs) [38], and autoencoders [39] are three core deep learning models that have shown remarkable abilities in handling complex financial data. The following sections detail the structural features of these models and their applications in financial attack detection scenarios.

#### 2.2.1. Recurrent Neural Networks

A recurrent neural network (RNN) is another deep learning model, particularly suited for processing sequential data [38,40]. In financial attack detection, RNNs are capable of handling the temporal dependencies of transaction data, identifying potential anomalous transaction patterns. A characteristic of RNNs is the introduction of loops in the model, allowing the network to retain information from previous moments. However, traditional RNNs are prone to gradient vanishing or exploding problems; thus, in practical applications, variants such as long short-term memory (LSTM) networks [41] or gated recurrent units (GRUs) [42] are commonly used. The basic formula of an RNN is expressed as [41]

$$h_t = \sigma(W_{hx}x_t + W_{hh}h_{t-1} + b_h) \tag{4}$$

Here, $h_t$ represents the hidden state at time $t$, $x_t$ is the input, $W_{hx}$ and $W_{hh}$ are weight matrices, $b_h$ is the bias term, and $\sigma$ is the activation function.

#### 2.2.2. Autoencoder

Autoencoders are a type of neural network used for unsupervised learning [43], which extract features by learning a compressed representation of input data. In financial attack detection, autoencoders can be utilized to learn the characteristics of normal transaction data, thereby identifying transactions that deviate from normal patterns. An autoencoder consists of two parts: an encoder and a decoder. The encoder maps input data to a hidden layer representation, while the decoder maps this representation back to the original data space. By minimizing the difference between the input data and the reconstructed data, the autoencoder can learn effective features of the data [39]. The basic formula of an autoencoder is given as [43]

$$\hat{x} = g(f(x)) \tag{5}$$

where $x$ is the input data, $f$ represents the encoding function, $g$ denotes the decoding function, and $\hat{x}$ is the reconstructed data.

## 3. Materials and Methods

### 3.1. Dataset Collection

In this study, multiple data sources were selected to construct a comprehensive and representative dataset for financial attack detection. The dataset comprises publicly available financial transaction records, synthetic data, and anonymized data from financial institutions.

1. Public financial transaction records: Data from publicly available sources, such as stock and credit card transactions, typically include information on transaction time, amount, and parties involved. The advantages of public data lie in their transparency and accessibility, contributing to the study's general applicability and reproducibility.
2. Synthetic data: Considering the sensitivity and difficulty in obtaining real financial data, synthetic data serve as an essential supplement. Algorithms are utilized to generate synthetic data with realistic characteristics, such as using Monte Carlo simulations for transaction patterns. This approach aids in simulating complex attack scenarios while preserving privacy.
3. Anonymized data from financial institutions: In collaboration with financial institutions, a portion of real financial transaction data were obtained. These data were anonymized before sharing to protect customer privacy. The authenticity and complexity of these data are crucial in enhancing the practicality and accuracy of the model.

The rationale for selecting these datasets for experimentation is based on several considerations: diversity and representativeness, combining public, synthetic, and real anonymized data to ensure the dataset covers a wide range of scenarios and patterns, thus enhancing the model's generalization ability. Authenticity and reliability, as real data provide a credible benchmark for assessing the model's performance in practical applications. Privacy protection, as the use of anonymized and synthetic data allows for research without disclosing sensitive information.

### 3.2. Dataset Annotation

Dataset annotation is a crucial step in ensuring the effectiveness of model training. For financial attack detection datasets, the annotation process involves categorizing transaction records as "normal" or "attack".

1. Annotation criteria: A series of criteria based on transaction characteristics, such as transaction frequency, amount, and historical behavior of the parties involved, were established. For instance, frequent large transactions might be flagged as suspicious attacks.
2. Expert review: The annotation process involved the participation of experts in the financial field. They conducted preliminary annotations based on their experience and industry knowledge, especially for complex or ambiguous cases.
3. Algorithmic assistance: To enhance efficiency, simple machine learning algorithms were used for pre-annotation, followed by manual expert review. This method combines the efficiency of algorithms with the accuracy of human expert judgment.
4. Iterative optimization: The annotation process is iterative. After initial training on pre-annotated data, the model's predictions are used to guide further manual annotations, forming a feedback loop.

A key mathematical principle in the annotation process is Bayes' theorem, which can be used to calculate the probability of a transaction being an attack given certain specific transaction features. Bayes' theorem [44] is expressed as

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \tag{6}$$

where $P(A|B)$ is the probability of event $A$ occurring under condition $B$, $P(B|A)$ is the probability of condition $B$ given that event $A$ has occurred, and $P(A)$ and $P(B)$ are the

marginal probabilities of event $A$ and condition $B$, respectively. Applying this principle allows for a more accurate assessment of the correlation between various transaction features and potential attack behaviors, thereby improving the accuracy of annotations and the performance of the model. Overall, the data collection and annotation process in this study aims to ensure the representativeness, authenticity, and privacy protection of the dataset while enhancing annotation efficiency and accuracy through the combination of expert knowledge and algorithmic analysis. This process is vital for developing an efficient and accurate financial attack detection model.

### *3.3. Proposed Method*

#### 3.3.1. Finsformer Overview

The Finsformer model is a deep learning model specifically designed for financial attack detection, combining the Transformer architecture with an innovative cluster-attention mechanism to enhance the accuracy and efficiency of detecting anomalous transaction behaviors in financial systems. The following sections provide a detailed overview of the Finsformer model's overall construction, its characteristics, and operational mechanism. Based on the Transformer architecture, widely applied for its outstanding performance in processing sequential data, the original Transformer model comprises multiple encoder and decoder layers, each containing a self-attention mechanism and a feed-forward neural network. In the Finsformer model, this structure has been partially modified and optimized to suit the characteristics of financial data. The operational mechanism of the Finsformer model involves several steps:

1. Input processing: The raw financial transaction data undergo preprocessing, including feature extraction, data cleansing, and normalization. The data, in the form of time series, include transaction amounts, timestamps, and account information. To effectively process these data, feature extraction and normalization are first carried out, transforming the raw data into a format that the model can handle.
2. Clustering and attention mechanism application: The cluster-attention mechanism is applied to the preprocessed data. Through cluster analysis, the model identifies key patterns in the data and focuses attention on these patterns. Differing from the traditional self-attention mechanism, the cluster-attention mechanism first clusters input data based on similarity, then applies the attention mechanism to these clusters. This approach enables the model to focus more on key patterns in the data, thereby enhancing the accuracy of detecting attack behaviors.
3. Feature extraction: After processing with the cluster-attention mechanism, the data are passed to the encoder and decoder layers. These layers further extract and process features, preparing for the final classification task.
4. Classification and detection: Finally, the model classifies the transactions based on the extracted features, determining whether each transaction is normal or an attack behavior.

By introducing the cluster-attention mechanism, the Finsformer model operates more efficiently in processing large volumes of financial transaction data, especially in identifying complex attack patterns. Compared to traditional models, Finsformer can more accurately detect abnormal patterns in financial data, which is crucial for detecting complex financial attack behaviors. The Finsformer model can be adjusted to suit different financial scenarios, exhibiting good flexibility and scalability.

#### 3.3.2. Transformer-Based Attack Behavior Detection Framework

In the Finsformer model, an improved Transformer architecture is adopted, as depicted in Figure 3. Here, the traditional multi-head attention mechanism has been innovatively replaced with a cluster-attention mechanism to more effectively handle the task of detecting

attack behaviors in financial systems. The following sections detail the design features of the network, its input–output characteristics, and the intricacies of the intermediate blocks, along with a mathematical explanation of the advantages of this design.

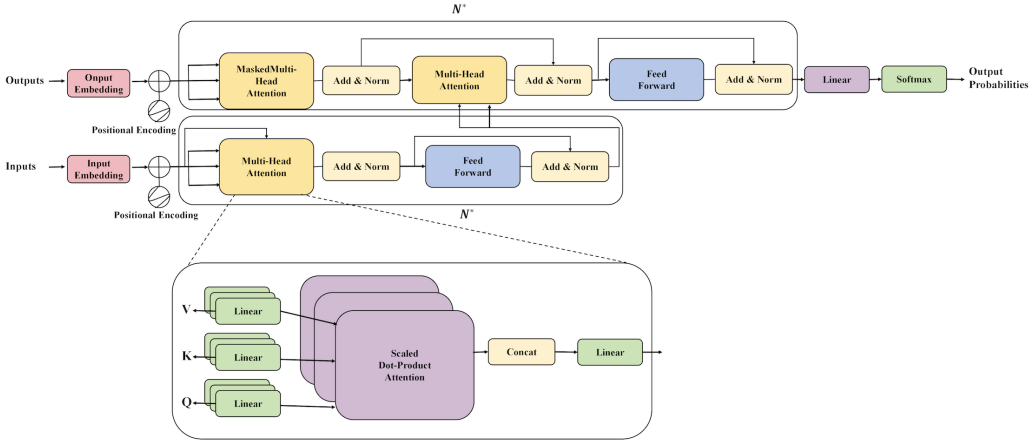


**Figure 3.** Illustration of the Finsformer.

Model design. The Finsformer model incorporates a cluster-attention mechanism in place of the conventional multi-head attention mechanism. This mechanism initially clusters the input data based on feature similarity, then calculates attention weights based on these clusters. This approach more accurately captures complex patterns and relationships in financial transaction data. The mathematical expression for the cluster-attention mechanism is

$$\text{ClusterAttention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}} + M\right)V \tag{7}$$

where $M$ is a mask matrix derived from data clustering, and $d_k$ is the dimensions of the key vectors.

Additionally, the Finsformer model employs a structure of multiple encoders and decoders, each comprising a cluster-attention module and a feed-forward neural network. These layers are stacked sequentially to extract and process features layer by layer. Following each cluster-attention module is a feed-forward neural network, which includes two linear transformations and an activation function.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{8}$$

The input–output dimensions of the model depend on the number of features in the financial data and the requirements of the classification task. Generally, after processing through the embedding layer, the dimensionality of the input data matches the size of the model's internal hidden layers.

Performance analysis. The cluster-attention mechanism enhances the model's understanding of financial transaction patterns through cluster analysis. This method allows the attention mechanism to focus more on groups of transactions with similar features, thereby improving the accuracy in identifying anomalous behaviors. By incorporating the cluster-attention mechanism, the Finsformer model retains the advantages of the Transformer architecture while being more suited to handle the characteristics of financial data, especially in identifying complex and covert attack patterns.

### 3.3.3. Cluster-Attention Mechanism

A key innovation in the Finsformer model, as presented in this paper, is the introduction of the cluster-attention mechanism, as shown in Figure 4. The following sections provide an in-depth explanation of the design details of the cluster-attention mechanism, its mathematical formulation, and the advantages it offers in detecting financial attacks.

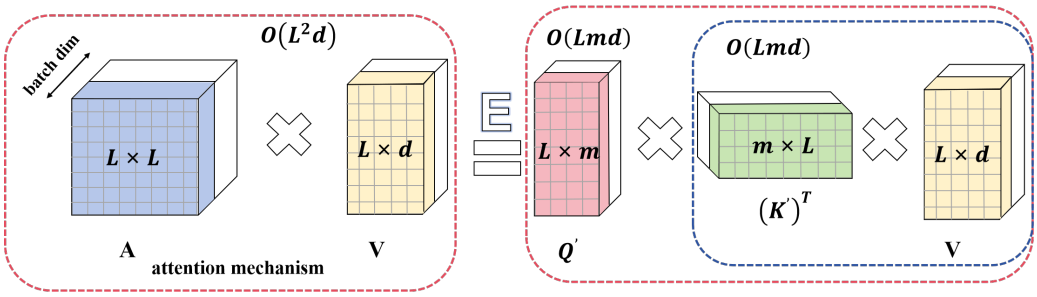


**Figure 4.** Illustration of the cluster-attention mechanism.

Differences from multi-head attention mechanism. The primary distinction between the cluster-attention mechanism and the traditional multi-head attention mechanism lies in the method of processing input data. While the multi-head attention mechanism focuses on capturing relationships between different positions within a sequence, cluster-attention emphasizes identifying inherent groups or patterns in the data. This involves two main steps:

1. Data clustering: The cluster-attention mechanism initially clusters the input data, aiming to identify potential groups or patterns within it. This approach allows the model to concentrate on transactions with similar features, thereby more effectively identifying anomalous behavior.
2. Attention weight calculation: Following the clustering, attention weights are calculated within each cluster. This method results in a more concentrated distribution of attention, helping to highlight significant transaction patterns.

Design of clustering algorithms. The choice of clustering algorithm is crucial to the effectiveness of the cluster-attention mechanism. In this model, clustering methods based on feature similarity, specifically k-means and DBSCAN, are employed to ensure the effective identification of inherent patterns in financial transaction data.

K-means clustering, one of the most common clustering algorithms, aims to minimize the sum of squared distances of each point to the center of its cluster. The objective of k-means clustering can be expressed as minimizing the sum of squared distances between data points and their respective cluster centers:

$$\min_{S} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} |\mathbf{x} - \boldsymbol{\mu}_i|^2 \tag{9}$$

where $k$ is the number of clusters, $S_i$ is the $i$-th cluster, $\boldsymbol{\mu}_i$ is the center of cluster $S_i$, and $\mathbf{x}$ is a data point. The algorithmic process is described in Algorithm 1. The specific logic of the algorithm is as follows: (1) Input and initialization: The inputs to the algorithm include a set of data points $X$ and the number of clusters $k$ to be formed. Initially, $k$ data points are randomly selected from the dataset $X$ to serve as the initial cluster centers $\mu_1, \mu_2, \ldots, \mu_k$. (2) Cluster assignment: The algorithm then enters a repetitive looping process. In each loop, for each data point $x$ in the dataset $X$, the nearest cluster center is identified based on a distance measure (here, the square of the Euclidean distance), and the point is assigned to the corresponding cluster. Specifically, for each point $x$, the distance between it and each cluster center $\mu_i$ is calculated, and the cluster center with the minimum distance is selected as the category $c(x)$ to which $x$ belongs. (3) Updating cluster centers: After all data points have been assigned to their respective clusters, the centers of each cluster are updated. Specifically, for each cluster $i$, the new cluster center $\mu_i$ is the mean of all points $x$ in that cluster. (4) Termination condition: The above steps of assignment and updating of cluster centers are repeated until the cluster centers no longer change, indicating that the algorithm has converged. (5) Output: Finally, the algorithm outputs the updated cluster centers $\mu$ and the cluster labels $c(x)$ for each point $x$ in the dataset $X$.

**Algorithm 1** K-means clustering algorithm

**Input:** Data points $X$, Number of clusters $k$
**Output:** Cluster centers $\mu$, Cluster labels for data points
Initialize cluster centers $\mu_1, \mu_2, \ldots, \mu_k$ randomly from $X$
**repeat**
  Assign each point $x \in X$ to the nearest cluster center:
  **for** each point $x \in X$ **do**
    $c(x) = \arg\min_i \|x - \mu_i\|^2$
  **end for**
  Update each cluster center to the mean of assigned points:
  **for** $i = 1$ to $k$ **do**
    $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$
  **end for**
**until** cluster centers do not change
**return** $\mu$, Cluster labels $c(x)$ for each $x \in X$

K-means clustering is effective in identifying spherical or convex clusters. By minimizing the distances between data points and cluster centers, the algorithm can identify inherent patterns in the data.

DBSCAN, an algorithm that forms clusters based on the density of data points within a region, is particularly effective for discovering clusters of arbitrary shapes. The algorithmic process is also outlined in Algorithm 2. The algorithm mainly involves two parameters: $\epsilon$ (neighborhood radius) and *MinPts* (minimum number of points in the neighborhood). The logic of the algorithm is as follows: (1) Input and initialization: The algorithm's input includes a dataset of points $X$, a neighborhood radius $\epsilon$, and a minimum number of points *MinPts*. Initially, all data points are marked as unvisited. (2) Traversing data points: For each point $p$ in the dataset $X$, if $p$ has already been visited, the process continues to the next point; otherwise, the following steps are executed. (3) Region query: For each point $p$, all points within its $\epsilon$ neighborhood are identified, forming a neighborhood set *NeighborPts*. (4) Core point determination and cluster expansion: If the number of points in *NeighborPts* is less than *MinPts*, $p$ is marked as a noise point; otherwise, $p$ is considered a core point, and a new cluster $C$ is created for $p$. Then, the *expandCluster* function is called to expand this cluster. (5) Cluster expansion: In the *expandCluster* function, for each point $q$ in *NeighborPts*, if $q$ has not been visited, it is marked as visited, and the number of points in the $\epsilon$ neighborhood of $q$ is checked. If the number of points in the neighborhood of $q$ is greater than or equal to *MinPts*, these new points are added to *NeighborPts*. If $q$ does not belong to any known cluster, $q$ is added to the current cluster $C$. (6) Output: Finally, the algorithm outputs the cluster labels for each data point.

The mathematical description of the above algorithm is as follows: for a given point $p$, its $\epsilon$-neighborhood is defined by

$$N_\epsilon(p) = q \in D | \text{dist}(p, q) \leq \epsilon \tag{10}$$

where $D$ is the dataset, and $\text{dist}(p, q)$ represents the distance between points $p$ and $q$. A point $p$ is a core point if and only if its $\epsilon$-neighborhood contains at least *MinPts* points, as in

$$|N_\epsilon(p)| \geq MinPts \tag{11}$$

where $|N_\epsilon(p)|$ denotes the number of points in the $\epsilon$-neighborhood of $p$. For each core point, all points in its $\epsilon$-neighborhood (including other core points and boundary points) belong to the same cluster. If the $\epsilon$-neighborhood of a core point overlaps with that of another core point, then all points in these neighborhoods are part of the same cluster.

**Algorithm 2** DBSCAN clustering algorithm

**Input:** Data points $X$, Radius $\epsilon$, Minimum points $MinPts$
**Output:** Cluster labels for data points
Initialize all points as unvisited
**for** each point $p \in X$ **do**
  **if** $p$ is visited **then**
    continue to next point
  **end if**
  Mark $p$ as visited
  $NeighborPts = \text{regionQuery}(p, \epsilon)$
  **if** number of points in $NeighborPts < MinPts$ **then**
    Mark $p$ as noise
  **else**
    $C = \text{nextCluster}()$
    $\text{expandCluster}(p, NeighborPts, C, \epsilon, MinPts)$
  **end if**
**end for**

**function** $\text{expandCluster}(p, NeighborPts, C, \epsilon, MinPts)$
**for** each point $q \in NeighborPts$ **do**
  **if** $q$ is not visited **then**
    Mark $q$ as visited
    $NeighborPts' = \text{regionQuery}(q, \epsilon)$
    **if** number of points in $NeighborPts' \geq MinPts$ **then**
      $NeighborPts = NeighborPts \cup NeighborPts'$
    **end if**
  **end if**
  **if** $q$ is not yet member of any cluster **then**
    Add $q$ to cluster $C$
  **end if**
**end for**

**function** $\text{regionQuery}(p, \epsilon)$
**return** all points within $p$'s $\epsilon$-neighborhood (including $p$)

### *3.4. Experiment Design*

The experimental design of the Finsformer model was aimed at comprehensively evaluating its performance in the domain of financial attack detection. This included the partitioning of the dataset, selection of appropriate baseline models, choice of optimizer, hyperparameter settings, and the design of ablation studies. The following sections detail these aspects of the experimental design.

#### 3.4.1. Experiment Configuration

The financial transaction dataset used in the experiments was divided into training, validation, and test sets to ensure effective testing of the model on independent data. Typically, the dataset was split into 70% for training, 15% for validation, and 15% for testing. The training set was used for model training, the validation set for adjusting model parameters during the training process, and the test set for final performance evaluation. This division method helped to reduce the risk of overfitting and ensured good generalization of the model to new data.

The Adam [45] optimizer was chosen for the training of the deep learning models. Combining the advantages of Momentum and RMSprop, the Adam optimizer adapts the learning rate across different parameter spaces, thereby enhancing the stability and efficiency of training. Hyperparameter settings significantly impacted model performance. In the experiments, key hyperparameters such as the learning rate, batch size, dimension

of hidden layers, and the number of attention heads were adjusted. These parameters were fine-tuned through multiple rounds of experimentation and performance evaluation on the validation set to achieve optimal training results.

#### 3.4.2. Testbed

In the experimental section of this article, the Finsformer model was compared with traditional deep learning models such as RNN [38], LSTM [41], Transformer [39], and BERT [46]. The selection of these models aimed to comprehensively assess the performance and innovative aspects of Finsformer in the domain of financial attack detection. Each model possesses unique architectural characteristics and application contexts, resulting in diverse performances in processing financial data. Therefore, their use as comparative objects deeply reveals the advantages and application scope of Finsformer.

Firstly, RNN (recurrent neural network) and its variant LSTM (long short-term memory) are classical deep learning models for processing sequential data. They are capable of capturing dependencies within time series, which is crucial for financial transaction data analysis. RNN and LSTM have been extensively applied in areas such as financial time-series prediction and fraud detection. However, RNN suffers from the problem of vanishing gradients, while LSTM, although mitigating this issue to some extent, has a high computational complexity. Comparing Finsformer with these two models enables the examination of whether Finsformer can more effectively capture long-term dependencies in financial time-series data and whether it has advantages in computational efficiency. Secondly, the Transformer model is renowned for its powerful self-attention mechanism, achieving revolutionary results in the field of natural language processing (NLP). It processes the entire data sequence in parallel, effectively capturing long-distance dependencies within the sequence. Given the complexity and diversity of financial data, the Transformer model also shows significant potential in the financial domain. Comparing it with Finsformer not only demonstrates the baseline performance of Transformer in financial attack detection tasks but also highlights the innovations and optimizations of Finsformer in this area. Furthermore, BERT, as a pre-trained model based on Transformer, excels in understanding complex contextual relationships. With its pre-training on large-scale corpora, BERT captures rich semantic information. In the realm of financial attack detection, BERT can provide a deep understanding of complex financial transaction patterns. Comparing it with Finsformer verifies whether Finsformer can surpass classic NLP models in understanding the complexity and subtle differences in financial data.

In summary, the selection of these specific models as comparison objects is not only due to their representativeness and advancement in processing time-series data and understanding complex patterns but also because of their potential in financial attack detection applications. By comparing with these models, this article not only demonstrates the superior performance of Finsformer in financial attack detection tasks but also reveals its unique advantages in processing financial data, especially in capturing complex transaction patterns and reducing false positives. Additionally, this comparison helps in understanding the application limitations of each model in the field of financial security, providing valuable insights and directions for improvement in future research.

#### 3.4.3. Evaluation Index

In the experimental section of this study, several key performance indicators were utilized to evaluate the effectiveness of the Finsformer model in the field of financial attack detection, including precision, recall, accuracy, and F1-score. The use of these evaluation metrics is crucial for comprehensively understanding the model's performance, as they reflect the model's characteristics and strengths from different perspectives. Precision, an important indicator of the model's predictive accuracy, represents the proportion of samples correctly identified as attack behaviors out of all samples labeled as such. A high precision implies greater reliability of the model in marking transactions as attacks, reducing the possibility of falsely identifying normal behaviors as attacks, which is particularly

important in the financial domain due to the potential for unnecessary economic losses and inconvenience to users. Recall measures the model's ability to identify positive (attack) samples, namely, the proportion of actual attack behaviors correctly identified by the model. A high recall indicates the model's effectiveness in detecting most attack behaviors, crucial for preventing financial fraud and reducing missed detections. Accuracy is a more comprehensive measure of performance, denoting the proportion of samples correctly predicted by the model out of the total samples. In financial attack detection tasks, high accuracy signifies the model's capability to effectively differentiate between attacks and normal behaviors. However, it should be noted that in unbalanced datasets, accuracy might not be a very reliable indicator. In financial attack detection tasks, F1-score is particularly important because it balances the need to identify attacks with the need to reduce false positives. Overall, these evaluation metrics collectively assess the model's performance in financial attack detection tasks, encompassing not only the model's ability to identify attack behaviors but also its reliability and efficiency in practical applications. Through these multi-dimensional assessments, a more comprehensive understanding and validation of the Finsformer model's application value and effectiveness in the field of financial security can be achieved.

## 4. Results and Discussion

### *4.1. Attack Behavior Detection Results*

The primary objective of the experiments described in this article was to evaluate the performance of various deep learning models in the task of detecting financial attack behaviors. The experiments conducted a thorough analysis of the strengths and limitations of each model by comparing their performance across three key metrics: precision, recall, and accuracy. The results of the experiments are presented in Table 1 and Figure 5.

**Table 1.** Detection results of different models.

| Model | Precision | Recall | Accuracy | F1-Score |
|---|---|---|---|---|
| RNN [38] | 0.83 | 0.81 | 0.82 | 0.82 |
| LSTM [41] | 0.88 | 0.86 | 0.87 | 0.87 |
| Transformer [39] | 0.90 | 0.88 | 0.89 | 0.89 |
| BERT [46] | 0.93 | 0.91 | 0.92 | 0.92 |
| Finsformer | 0.97 | 0.94 | 0.95 | 0.95 |

It was observed that recurrent neural networks (RNNs) exhibited relatively weaker performance in this task. This is primarily attributed to the challenges RNNs face in handling long-term dependencies, limiting their ability to recognize complex patterns in financial transaction data. Long short-term memory (LSTM) networks, as an improvement of RNNs, effectively address the long-term dependency issue through the introduction of gating mechanisms, thereby showing better performance in recognizing and memorizing complex patterns in financial transaction data. The Transformer model outperformed both RNN and LSTM, benefiting from its self-attention mechanism that processes all elements of a sequence simultaneously, effectively capturing global dependencies. This ability enables the Transformer to more accurately capture complex relationships between various transactions in financial data, thus enhancing the accuracy of attack detection. The BERT model, a pre-trained language model based on the Transformer, excelled across all metrics due to its strong contextual understanding capabilities and the advantage of large-scale pre-training. This indicates BERT's effectiveness in handling complex contextual information, which is particularly important for identifying financial attack behaviors. Ultimately, the Finsformer model achieved the best performance on all evaluation metrics. This can be attributed mainly to its innovative cluster-attention mechanism, which, by performing cluster analysis on financial transaction data, captures key patterns in the data more accurately. Compared to traditional attention mechanisms, the cluster-attention

mechanism focuses more on significant areas in the data, significantly enhancing the accuracy in identifying anomalous behaviors.

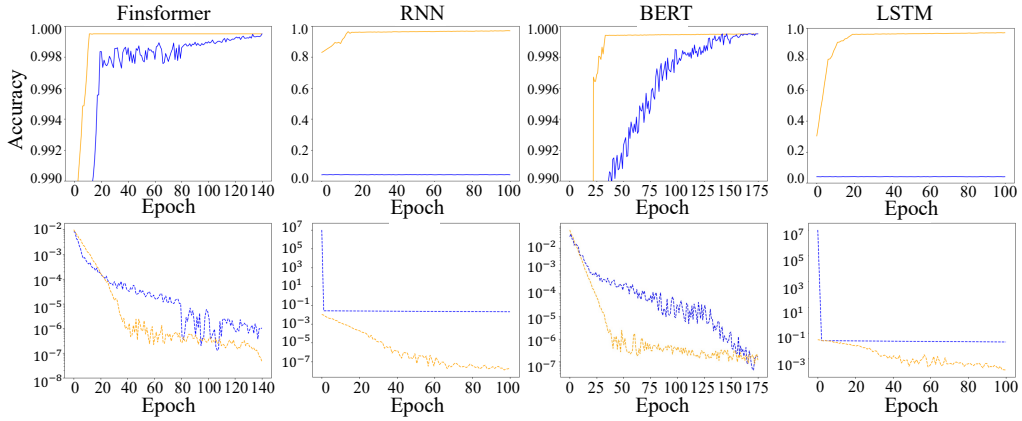


**Figure 5.** Curves of detection results. Orange line is the training, blue one is the validation.

In summary, the experimental results not only demonstrate the characteristics and performance of different models in the task of detecting financial attack behaviors but also confirm the exceptional capability of Finsformer in handling such tasks from both mathematical and machine learning perspectives. These findings provide valuable insights and references for future research in the field of financial security.

### *4.2. Ablation Study on Cluster-Attention Mechanism*

In the ablation study of cluster-attention mechanism within this article, the focus was on exploring the impact of different attention mechanisms on the performance of models for detecting financial attack behaviors. The experiments, by comparing the performance of models without attention mechanism, with multi-head attention mechanism, and with cluster-attention mechanism across key metrics such as precision, recall, and accuracy, provided an in-depth analysis of the strengths and limitations of each mechanism, thereby demonstrating the effectiveness of cluster-attention in specific applications, as shown in Table 2.

**Table 2.** Detection results of different attention mechanisms.

| Attention Mechanism | Precision | Recall | Accuracy | F1-Score |
|---|---|---|---|---|
| No Attention [36] | 0.83 | 0.79 | 0.82 | 0.81 |
| Multi-head Attention [39] | 0.90 | 0.91 | 0.90 | 0.90 |
| Cluster-attention | 0.97 | 0.94 | 0.95 | 0.95 |

The results indicated that models lacking an attention mechanism performed relatively poorly across all metrics. This was mainly attributed to their inability to dynamically focus on specific parts of the data, particularly in processing complex sequential data such as financial transactions. This static approach to information processing limited their capacity to capture key patterns and relationships in financial transaction data, resulting in less accurate identification of attack behaviors. The introduction of the multi-head attention mechanism significantly improved model performance. By simultaneously processing multiple parts of a sequence and calculating the interrelationships between different parts, the multi-head attention mechanism effectively enhanced the model's understanding of complex relationships. This indicated that the mechanism was capable of effectively handling complex patterns in financial transaction data, especially excelling in capturing associations between transactions, thereby improving the precision in detecting attack behaviors. Finally, the cluster-attention mechanism exhibited the best performance in the experiments. This outcome reflected that cluster-attention, by clustering data based on

feature similarity and then calculating attention weights on these clusters, focused attention more on key areas within the data. Through precise cluster analysis, the cluster-attention mechanism was able to more accurately focus on important patterns and relationships in the data, particularly enhancing the ability to identify anomalous behaviors in financial data.

In summary, the experimental results clearly showed the roles and effects of different attention mechanisms in the task of detecting financial attack behaviors. Models without an attention mechanism showed poorer performance due to a lack of focus on important parts of the data. The multi-head attention mechanism significantly enhanced model performance by providing dynamic attention to the data. The cluster-attention mechanism, on the other hand, further optimized the allocation of attention through precise cluster analysis, significantly improving the accuracy and efficiency of the model in processing complex financial data. These findings not only prove the potential application of the cluster-attention mechanism in the domain of financial attack detection but also provide valuable references for future research in similar fields.

### *4.3. Ablation Study on Transformer Feature Extractor*

This section is dedicated to evaluating the performance of different feature extractors in the task of detecting financial attack behaviors. By comparing the performance of manual feature extraction (represented by support vector machine, SVM), recurrent neural network (RNN), convolutional neural network (CNN), and Transformer feature extractors across key metrics such as precision, recall, and accuracy, the experiment conducted an in-depth analysis of the advantages and limitations of each feature extractor and explored their effectiveness in handling financial data.

As shown in Table 3, the performance of manual feature extractors was relatively lower. This was mainly attributed to the inability of manually selected features to fully capture complex and non-linear patterns in financial data, limiting the model's accuracy in identifying attack behaviors. While SVM performs well in linear problems, its efficacy diminishes in the face of complex financial data. The RNN feature extractor, suitable for processing sequential data, showed better performance than manual feature extraction. Its ability to capture temporal dependencies is particularly important for financial transaction data analysis. However, RNNs may face challenges in handling long-term dependencies due to the issue of vanishing gradients. The performance of the CNN feature extractor was slightly inferior to that of RNN, possibly because temporal dependencies in financial transaction data are more critical than spatial patterns. Nonetheless, CNN showed certain effectiveness in capturing local patterns, especially in data with strong spatial correlations. The Finsformer feature extractor exhibited the best performance across all metrics, demonstrating its strong capability in global feature extraction. The Finsformer, through its attention mechanism, processes all elements of a sequence simultaneously, effectively capturing the global dependencies in financial transaction data. This mechanism allows the Finsformer to consider all elements in a sequence and capture the complex relationships between them, especially in complex and non-linear financial data.

**Table 3.** Detection results of different feature extractors.

| Feature Extractor | Precision | Recall | Accuracy | F1-Score |
|---|---|---|---|---|
| Manual Feature Extractor (SVM) [47] | 0.87 | 0.89 | 0.89 | 0.88 |
| RNN [38] | 0.93 | 0.91 | 0.91 | 0.92 |
| CNN [36] | 0.92 | 0.91 | 0.92 | 0.91 |
| Finsformer | 0.97 | 0.94 | 0.95 | 0.95 |

### *4.4. Limitations and Future Work*

This study has achieved certain innovations and progress in the field of financial attack detection, yet it also faces some limitations and proposes directions for future research. Firstly, regarding the innovations of this work, the Finsformer model, by integrating the

Transformer architecture and cluster-attention mechanism, has effectively enhanced the accuracy and efficiency of financial attack detection. The Transformer's powerful capability in processing sequential data allows Finsformer to capture long-term dependencies and complex patterns in financial data. Concurrently, the introduction of the cluster-attention mechanism strengthens the model's ability to identify key patterns in financial transaction data, particularly demonstrating superior performance in dealing with large-scale and diverse datasets. The experimental results show that Finsformer surpasses traditional deep learning models such as RNN, LSTM, Transformer, and BERT in several key performance metrics. However, this study also has limitations. Primarily, the computational efficiency of the Finsformer model, especially in processing large datasets, remains a challenge. Although the Transformer architecture offers good parallel processing capabilities, the training and inference speed of the model might be impacted when handling very large-scale data. Additionally, the performance of the model largely depends on the quality and diversity of the data. In practical applications, financial transaction data might be plagued with issues of noise, inconsistency, and incompleteness, potentially affecting the model's accuracy and robustness.

Future research directions could unfold in several areas: firstly, optimizing the computational efficiency of the Finsformer model, particularly in handling large datasets, and exploring more efficient training and inference methods. Secondly, further enhancing the model's robustness to noise and inconsistent data, for instance, by introducing more advanced data preprocessing and augmentation techniques. Additionally, considering integrating the Finsformer model with other types of security defense mechanisms, such as traditional network security technologies, to improve the overall level of financial security. Furthermore, exploring the application of the Finsformer model in a broader range of financial security scenarios, such as fraud detection and abnormal transaction monitoring, to validate the model's generalization ability and practical application value. Simultaneously, testing and optimizing Finsformer on more diverse and larger datasets is considered, to assess its adaptability and effectiveness in different financial environments.

In conclusion, although this paper has made some progress in the realm of financial attack detection, continued in-depth research and constant optimization are required in the face of ever-changing and increasingly complex financial security threats. Through future efforts, significant achievements in the field of financial security are anticipated, providing robust technical support for the stability and development of the finance industry.

## 5. Conclusions

With the rapid development of financial technology, the financial system is increasingly facing security threats. The Finsformer model proposed in this paper enhances the detection capability of financial attack behaviors by introducing advanced deep learning technologies. Its core innovation lies in combining the Transformer architecture with the cluster-attention mechanism for detecting financial attack behaviors. Experimental results demonstrate that Finsformer surpasses traditional models such as RNN, LSTM, Transformer, and BERT in key metrics such as precision, recall, and accuracy, achieving scores of 0.97, 0.94, and 0.95, respectively. Additionally, ablation experiments on different feature extractors have confirmed the effectiveness of the Transformer feature extractor in processing complex financial data. Future research will focus on further optimizing the Finsformer model, including improving the computational efficiency of the model, expanding its application scenarios, and exploring its application on larger and more diverse datasets.

**Author Contributions:** Conceptualization, H.A., R.M. and C.L.; methodology, H.A., R.M. and P.L.; software, H.A., Y.Y., Y.Z. and D.F.; validation, R.M., J.L. and D.F.; formal analysis, Y.Y., T.C. and P.L.; investigation, Y.Z.; resources, T.C. and P.L.; data curation, Y.Y., T.C. and X.W.; writing—original draft, H.A., R.M., Y.Y., T.C., Y.Z., P.L., J.L., X.W. and C.L.; writing—review and editing, X.W., D.F. and C.L.; visualization, Y.Z., J.L. and D.F.; 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 National Natural Science Foundation of China grant number 61202479.

**Data Availability Statement:** The raw data supporting the conclusions of this article will be made available by the authors on request.

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

## References

1. 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]
2. Yang, X.; Zhang, C.; Sun, Y.; Pang, K.; Jing, L.; Wa, S.; Lv, C. FinChain-BERT: A High-Accuracy Automatic Fraud Detection Model Based on NLP Methods for Financial Scenarios. *Information* **2023**, *14*, 499. [CrossRef]
3. Zhang, L.; Wang, R.; Li, Z.; Li, J.; Ge, Y.; Wa, S.; Huang, S.; Lv, C. Time-Series Neural Network: A High-Accuracy Time-Series Forecasting Method Based on Kernel Filter and Time Attention. *Information* **2023**, *14*, 500. [CrossRef]
4. Diaz-Verdejo, J.; Munoz-Calle, J.; Estepa Alonso, A.; Estepa Alonso, R.; Madinabeitia, G. On the Detection Capabilities of Signature-Based Intrusion Detection Systems in the Context of Web Attacks. *Appl. Sci.* **2022**, *12*, 852. [CrossRef]
5. Saez-de Camara, X.; Luis Flores, J.; Arellano, C.; Urbieta, A.; Zurutuza, U. Clustere d fe derate d learning architecture for network anomaly detection in large scale heterogeneous IoT networks. *Comput. Secur.* **2023**, *131*, 103299. [CrossRef]
6. Abdulganiyu, O.H.; Tchakoucht, T.A.; Saheed, Y.K. A systematic literature review for network intrusion detection system (IDS). *Int. J. Inf. Secur.* **2023**, *22*, 1125–1162. [CrossRef]
7. Yang, W.; Lam, K.Y. Effective Anomaly Detection Model Training with only Unlabeled Data by Weakly Supervised Learning Techniques. In *Lecture Notes in Computer Science, PT I, Proceedings of the ICICS 2021: Information and Communications Security, Chongqing, China, 19–21 November 2021*; Gao, D., Li, Q., Guan, X., Liao, X., Eds.; Springer: Cham, Switzerland, 2021; Volume 12918, pp. 402–425. [CrossRef]
8. Yang, S.; Ding, Y.; Xie, B.; Guo, Y.; Bai, X.; Qian, J.; Gao, Y.; Wang, W.; Ren, J. Advancing Financial Forecasts: A Deep Dive into Memory Attention and Long-Distance Loss in Stock Price Predictions. *Appl. Sci.* **2023**, *13*, 12160. [CrossRef]
9. Elsaeidy, A.A.; Jagannath, N.; Sanchis, A.G.; Jamalipour, A.; Munasinghe, K.S. Replay Attack Detection in Smart Cities Using Deep Learning. *IEEE Access* **2020**, *8*, 137825–137837. [CrossRef]
10. Waqar, M.; Fareed, S.; Kim, A.; Malik, S.U.R.; Imran, M.; Yaseen, M.U. Malware Detection in Android IoT Systems Using Deep Learning. *CMC—Comput. Mater. Contin.* **2023**, *74*, 4399–4415. [CrossRef]
11. Sandouka, S.B.; Bazi, Y.; Al Rahhal, M.M. EfficientNet Combined with Generative Adversarial Networks for Presentation Attack Detection. In Proceedings of the 2020 International Conference on Artificial Intelligence & Modern Assistive Technology (ICAIMAT), Riyadh, Saudi Arabia, 24–26 November 2020. [CrossRef]
12. Alshingiti, Z.; Alaqel, R.; Al-Muhtadi, J.; Haq, Q.E.U.; Saleem, K.; Faheem, M.H. A Deep Learning-Based Phishing Detection System Using CNN, LSTM, and LSTM-CNN. *Electronics* **2023**, *12*, 232. [CrossRef]
13. Ozcan, A.; Catal, C.; Donmez, E.; Senturk, B. A hybrid DNN-LSTM model for detecting phishing URLs. *Neural Comput. Appl.* **2023**, *35*, 4957–4973. [CrossRef] [PubMed]
14. Afzal, S.; Asim, M.; Javed, A.R.; Beg, M.O.; Baker, T. URLdeepDetect: A Deep Learning Approach for Detecting Malicious URLs Using Semantic Vector Models. *J. Netw. Syst. Manag.* **2021**, *29*, 21. [CrossRef]
15. Pastor, A.; Mozo, A.; Vakaruk, S.; Canavese, D.; Lopez, D.R.; Regano, L.; Gomez-Canaval, S.; Lioy, A. Detection of Encrypted Cryptomining Malware Connections With Machine and Deep Learning. *IEEE Access* **2020**, *8*, 158036–158055. [CrossRef]
16. Wang, B.; Yuan, X.; Duan, L.; Ma, H.; Su, C.; Wang, W. DeFiScanner: Spotting DeFi Attacks Exploiting Logic Vulnerabilities on Blockchain. *IEEE Trans. Comput. Soc. Syst.* **2022**. [CrossRef]
17. Alkhatib, I.K.; Al-Aiad, I.A.; Almahmoud, M.H.; Elayan, O.N. Credit Card Fraud Detection Based on Deep Neural Network Approach. In Proceedings of the 2021 12th International Conference on Information and Communication Systems (ICICS), Valencia, Spain, 24–26 May 2021; pp. 153–156. [CrossRef]
18. Fursov, I.; Morozov, M.; Kaploukhaya, N.; Kovtun, E.; Rivera-Castro, R.; Gusev, G.; Babaev, D.; Kireev, I.; Zaytsev, A.; Burnaev, E. Adversarial Attacks on Deep Models for Financial Transaction Records. In Proceedings of the KDD '21: 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual, 14–18 August 2021; pp. 2868–2878. [CrossRef]
19. Qasaimeh, M.; Abu Hammour, R.; Yassein, M.B.; Al-Qassas, R.S.; Torralbo, J.A.L.; Lizcano, D. Advanced security testing using a cyber-attack forecasting model: A case study of financial institutions. *J. Softw. Evol. Process* **2022**, *34*, e2489. [CrossRef]
20. Iftikhar, S.; Asim, M.; Zhang, Z.; Muthanna, A.; Chen, J.; El-Affendi, M.; Sedik, A.; Abd El-Latif, A.A. Target Detection and Recognition for Traffic Congestion in Smart Cities Using Deep Learning-Enabled UAVs: A Review and Analysis. *Appl. Sci.* **2023**, *13*, 3995. [CrossRef]
21. Rizvi, S.; Orr, R.; Cox, A.; Ashokkumar, P.; Rizvi, M.R. Identifying the attack surface for IoT network. *Internet Things* **2020**, *9*, 100162. [CrossRef]
22. Eisenbach, T.M.; Kovner, A.; Lee, M.J. Cyber risk and the US financial system: A pre-mortem analysis. *J. Financ. Econ.* **2022**, *145*, 802–826. [CrossRef]

23. Ismagilova, E.; Hughes, L.; Rana, N.P.; Dwivedi, Y.K. Security, privacy and risks within smart cities: Literature review and development of a smart city interaction framework. *Inf. Syst. Front.* **2022**, *24*, 393–414. [CrossRef]
24. Mohammadpourfard, M.; Khalili, A.; Genc, I.; Konstantinou, C. Cyber-resilient smart cities: Detection of malicious attacks in smart grids. *Sustain. Cities Soc.* **2021**, *75*, 103116. [CrossRef]
25. Zhu, L.; Li, M.; Metawa, N. Financial risk evaluation Z-score model for intelligent IoT-based enterprises. *Inf. Process. Manag.* **2021**, *58*, 102692. [CrossRef]
26. Alkhalil, Z.; Hewage, C.; Nawaf, L.; Khan, I. Phishing attacks: A recent comprehensive study and a new anatomy. *Front. Comput. Sci.* **2021**, *3*, 563060. [CrossRef]
27. Barraclough, P.A.; Fehringer, G.; Woodward, J. Intelligent cyber-phishing detection for online. *Comput. Secur.* **2021**, *104*, 102123. [CrossRef]
28. GILL, M.A.; AHMAD, N.; KHAN, M.; ASGHAR, F.; RASOOL, A. Cyber Attacks Detection Through Machine Learning in Banking. *Bull. Bus. Econ. (BBE)* **2023**, *12*, 34–45.
29. Sagduyu, Y.E.; Shi, Y.; Erpek, T. IoT network security from the perspective of adversarial deep learning. In Proceedings of the 2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), Boston, MA, USA, 10–13 June 2019; pp. 1–9.
30. Yuan, C.; Yang, H. Research on K-value selection method of K-means clustering algorithm. *J* **2019**, *2*, 226–235. [CrossRef]
31. Sinaga, K.P.; Yang, M.S. Unsupervised K-means clustering algorithm. *IEEE Access* **2020**, *8*, 80716–80727. [CrossRef]
32. Ikotun, A.M.; Ezugwu, A.E.; Abualigah, L.; Abuhaija, B.; Heming, J. K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. *Inf. Sci.* **2023**, *622*, 178–210. [CrossRef]
33. Starczewski, A.; Goetzen, P.; Er, M.J. A new method for automatic determining of the DBSCAN parameters. *J. Artif. Intell. Soft Comput. Res.* **2020**, *10*, 209–221. [CrossRef]
34. Hahsler, M.; Piekenbrock, M.; Doran, D. dbscan: Fast density-based clustering with R. *J. Stat. Softw.* **2019**, *91*, 1–30. [CrossRef]
35. Deng, D. DBSCAN clustering algorithm based on density. In Proceedings of the 2020 7th International Forum on Electrical Engineering and Automation (IFEEA), Hefei, China, 25–27 September 2020; pp. 949–953.
36. 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, 27–30 June 2016.
37. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
38. Zhao, Y. Design of a corporate financial crisis prediction model based on improved ABC-RNN+ Bi-LSTM algorithm in the context of sustainable development. *PeerJ Comput. Sci.* **2023**, *9*, e1287. [CrossRef]
39. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
40. Binoy, S.J.; Jos, J. Financial Market Forecasting using Macro-Economic Variables and RNN. In Proceedings of the 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 28–29 April 2022; pp. 1366–1371.
41. Chen, J.; Zhang, Y.; Wu, J.; Cheng, W.; Zhu, Q. SOC estimation for lithium-ion battery using the LSTM-RNN with extended input and constrained output. *Energy* **2023**, *262*, 125375. [CrossRef]
42. Pirani, M.; Thakkar, P.; Jivrani, P.; Bohara, M.H.; Garg, D. A comparative analysis of ARIMA, GRU, LSTM and BiLSTM on financial time series forecasting. In Proceedings of the 2022 IEEE International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), Ballari, India, 23–24 April 2022; pp. 1–6.
43. Al Duhayyim, M.; Alsolai, H.; Al-Wesabi, F.N.; Nemri, N.; Mahgoub, H.; Hilal, A.M.; Hamza, M.A.; Rizwanullah, M. Optimized stacked autoencoder for IoT enabled financial crisis prediction model. *CMC—Comput. Mater. Contin.* **2022**, *71*, 1079–1094.
44. Koch, K.R.; Koch, K.R. Bayes' theorem. In *Bayesian Inference with Geodetic Applications*; Springer: Berlin/Heidelberg, Germany, 1990; pp. 4–8.
45. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
46. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
47. Kurani, A.; Doshi, P.; Vakharia, A.; Shah, M. A comprehensive comparative study of artificial neural network (ANN) and support vector machines (SVM) on stock forecasting. *Ann. Data Sci.* **2023**, *10*, 183–208. [CrossRef]