applied sciences

MDPI

Article

Enhancing Network Attack Detection Accuracy through the Integration of Large Language Models and Synchronized Attention Mechanism

Yuzhe Bai †, Min Sun †, Liman Zhang †, Yinong Wang, Sihan Liu, Yanqiu Liu, Jingling Tan, Yingqiu Yang and Chunli Lv *

China Agricultural University, Beijing 100083, China; byz@cau.edu.cn (Y.B.); 2021308160227@cau.edu.cn (M.S.); 2020308130508@cau.edu.cn (L.Z.); wangyinong@cau.edu.cn (Y.W.); liusihan@cau.edu.cn (S.L.); 2023311320326@cau.edu.cn (Y.L.); 2022308250124@cau.edu.cn (J.T.); yyq@cau.edu.cn (Y.Y.)

* Correspondence: lvcl@cau.edu.cn

† These authors contributed equally to this work.

Abstract: In this study, we propose a novel method for detecting cyberattack behaviors by leveraging the combined strengths of large language models and a synchronized attention mechanism. Extensive experiments conducted on diverse datasets, including server logs, financial behaviors, and comment data, demonstrate the significant advantages of this method over existing models such as Transformer, BERT, OPT-175B, LLaMa, and ChatGLM3-6B in key performance metrics such as precision, recall, and accuracy. For instance, on the server log dataset, the method achieved a precision of 93%, a recall of 91%, and an accuracy of 92%; on the financial behavior dataset, it reached a precision of 90%, a recall of 87%, and an accuracy of 89%; and on the comment data dataset, it excelled with a precision of 95%, a recall of 93%, and an accuracy of 94%. The introduction of a synchronized attention mechanism and a newly designed synchronized loss function proved especially effective, enhancing the method's ability to process multi-source data and providing superior performance in identifying complex cyberattack patterns. Ablation experiments further validated the crucial roles of these innovations in boosting model performance: the synchronous attention mechanism substantially improved the model's precision, recall, and accuracy to 93%, 89%, and 91% respectively, far exceeding other attention mechanisms. Similarly, the synchronized loss showcased a significant advantage, achieving the best performance across all tested metrics compared to traditional cross-entropy loss, focal loss, and MSE. These results underscore the method's ability to deeply mine and analyze semantic information and contextual relationships within text data as well as to effectively integrate and process multimodal data, thereby offering strong technical support for the accurate and efficient detection of cyberattack behaviors.

Keywords: network attack detection; large language models; synchronized attention mechanism; deep learning in cybersecurity; cross-domain application of NLP

Citation: Bai, Y.; Sun, M.; Zhang, L.; Wang, Y.; Liu, S.; Liu, Y.; Tan, J.; Yang, Y.; Lv, C. Enhancing Network Attack Detection Accuracy through the Integration of Large Language Models and Synchronized Attention Mechanism. *Appl. Sci.* **2024**, *14*, 3829. https://doi.org/10.3390/app14093829

Academic Editors: Sin Gee Teo, Rongxing Lu and Ruitao Feng

Received: 7 April 2024
Revised: 25 April 2024
Accepted: 26 April 2024
Published: 30 April 2024

1. Introduction

With the rapid development of the internet and the widespread use of social media [1], online platforms have become crucial venues for information exchange, life sharing, and opinion expression. However, these platforms also serve as breeding grounds for cyberattack behaviors, including cyberbullying [2], malicious speech dissemination [3], phishing attacks [4], and more. Such behaviors not only harm individuals' mental health but threaten the security and harmony of the online environment; consequently, the development of effective detection mechanisms to identify and prevent these attack behaviors has become an important task in the field of cybersecurity.

Traditional attack behavior detection methods primarily rely on rule matching [5] and keyword filtering. However, these methods often lack flexibility and struggle to adapt to

the increasingly complex and diversified cyberattack methods. Rezaimehr Fatemeh and colleagues [6] analyzed 25 collaborative filtering recommendation system (CFRS) research samples from 2009 to 2019, finding that many studies merely attempt to detect attacks, while the models are lacking in robustness and fail to integrate implicit features and social information for the detection of malicious users.

In recent years, with the advancement of deep learning technology, textual analysis methods based on these technologies have been widely applied to the automatic detection of attack behaviors [7], achieving a certain effectiveness. However, these methods continue to face challenges in understanding the deep meaning and context of texts; especially when dealing with complex language features such as sarcasm, puns, or metaphors, their effectiveness remains limited. For instance, Alshehri Abdullah and colleagues [8] used machine learning and user behavior analysis to detect cyberattacks, with their experimental results showing that RNN-LSTM achieved the highest accuracy of 97%; yet, their model's accuracy was affected when detecting internal attacks. In response, Elnakib Omar and colleagues [9] proposed an enhanced anomaly-based intrusion detection deep learning multi-class classification model to detect Internet of Things (IoT) intrusions. They classified fifteen behaviors, including fourteen types of attacks, achieving 95% accuracy; however, the model's accuracy and complexity came at the cost of increased time. Meddeb Rahma and colleagues [10] proposed a stacked autoencoder approach for intrusion detection systems (IDS) in Mobile Ad Hoc Networks (MANET), with empirical experiments showing that Stacked-IDS (a DNN with a stacked auto-encoder) exhibited higher accuracy than Deep-IDS (a DNN with a deep auto-encoder) in detecting malicious nodes. The authors noted that the complexity of attacks and imbalanced datasets require further research for validation. Elsaeidy Asmaa A and colleagues [11] developed a deep learning-based model for replay attack detection in smart cities. The model featured four hidden layers, a global average pooling layer, and an output layer, and achieved satisfactory performance in soil management and environmental monitoring. Modeling for environmental monitoring posed more challenges than soil management due to the complex nature of environmental monitoring datasets, testing the overall performance of the model. Moreover, in the financial sector, Nicholls Jack and colleagues [12] discussed various fraud methods by investigating the financial cybercrime ecosystem, including how graph-based techniques and neural network models can be used to counter financial cybercrime.

Large language models, such as OpenAI's GPT (Generative Pre-training Transformer) series [13], have demonstrated their powerful capabilities in understanding and generating text through pretraining on large-scale textual data. These models can capture the deep semantics and rich contextual information present in text, offering the possibility of deeply understanding human language. Hu and colleagues [14] applied large language models to construct knowledge graphs from unstructured open-source threat intelligence, with their proposed LLM-TIKG method showing good results in named entity recognition and TTP classification, achieving accuracies of 87.88% and 96.53%, respectively. Xu and colleagues [15] introduced AutoAttacker, a large language model-guided system for automatic cyberattack detection; yet, synchronizing information remains a challenge.

Furthermore, the attention mechanism [16], an important technique in deep learning, allows models to "focus" on different parts by assigning different "attention" to them, thereby enhancing the model's ability to focus on the most relevant information in text and improving the ability to process complex language features. For example, An and colleagues [17] proposed a financial system attack detection model called Finsformer based on a transformer and synchronized attention mechanism. The experimental results showed that their model significantly outperformed traditional models such as RNN and LSTM, achieving an accuracy of 97%; however, the model's computational efficiency was low, affecting accuracy and inference speed when dealing with large datasets. Wang and colleagues [18] combined transformer and expert systems and evaluated the resulting model on real phishing websites, showing that even after three months, it maintained an accuracy of 96%; however, deploying the model on edge devices posed a challenge.

The difficulty of detecting network attack behaviors lies in the need to accurately understand the meaning of texts and their context, especially when attack methods use language with obscure or double meaning. Additionally, the diversity and complexity of network attack behaviors demand highly flexible and adaptable detection methods. To address these challenges, we utilized the latest large language models as a foundation, leveraging their ability to pretrain on large-scale text data in order to capture subtle semantic differences and rich contextual information. Concurrently, the introduction of synchronized attention mechanisms enables models to adaptively focus on the most critical information for attack behavior detection, thereby enhancing the accuracy and efficiency of detection.

In this context, the present paper proposes a novel approach that combines large language models with synchronized attention mechanisms. This approach is specifically designed to enhance accuracy and efficiency when detecting cyberattack behaviors. This innovative integration marks the first time that large language models have been coupled with synchronized attention mechanisms, providing a robust solution for identifying a wide range of complex and covert attack behaviors more effectively than traditional methods. The synchronized attention mechanism is a key innovation that enables a dynamic and holistic integration of data features across different layers of the language model, facilitating a deeper understanding of complex interactions within data. This approach significantly enhances the ability to detect nuanced or obscured cyberthreats. Additionally, the introduction of a synchronized loss function further optimizes our model's ability to handle multi-source data, significantly improving detection capabilities across diverse cyber-environments such as server logs, financial transactions, and social media interactions. Extensive experimental validation across these varied datasets demonstrates the superior performance of our method over existing models, underscoring its effectiveness in practical applications. This introduction sets the stage with a background on cybersecurity and the challenges involved in detecting attack behaviors, then reviews related work, including the deployment of large language models and attention mechanisms in text analysis, and highlights the limitations of current detection techniques. Our proposed method, which integrates large language models with a novel synchronized attention mechanism and synchronized loss function, covers comprehensive aspects such as the model architecture, data processing, experimental design, and evaluation metrics. This approach is theoretically innovative and has been empirically validated, proving its effectiveness and superiority in practical applications.

2. Background and Related Work

2.1. Large Language Models

In recent years, the development of Large Language Models (LLMs) [19] has garnered widespread attention in the field of artificial intelligence, as shown in Figure 1.

This attention stems from their remarkable achievements in Natural Language Processing (NLP) tasks, ranging from basic text classification [20] to complex question-answering systems [21] and natural language understanding [22]. These achievements are built on the foundation of the Transformer architecture [23], proposed by Vaswani et al. in 2017, which rapidly became the dominant technology for NLP tasks [24]. The core of the Transformer architecture is the self-attention mechanism, allowing models to focus on different parts of the sequence data. The mathematical expression of the self-attention mechanism is as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \quad (1)$$

Here, Q, K, and V represent the matrices for Query, Key, and Value, respectively, with d_k denoting the dimension of the key vectors. This formula calculates the output by computing the similarity between the queries and keys, then using these softmax scores to weight the values. By stacking multiple attention layers and incorporating feed-forward neural networks, residual connections, and layer normalization in each layer, Transformer models build deep architectures. The advantage of this architecture lies in its strong

parallel processing capabilities, ability to capture long-distance dependencies, and flexible adjustment of model parameters through layer and width expansion. BERT (Bidirectional Encoder Representations from Transformers) is a Transformer-based model introduced by Google in 2018 [25], as shown in Figure 2.

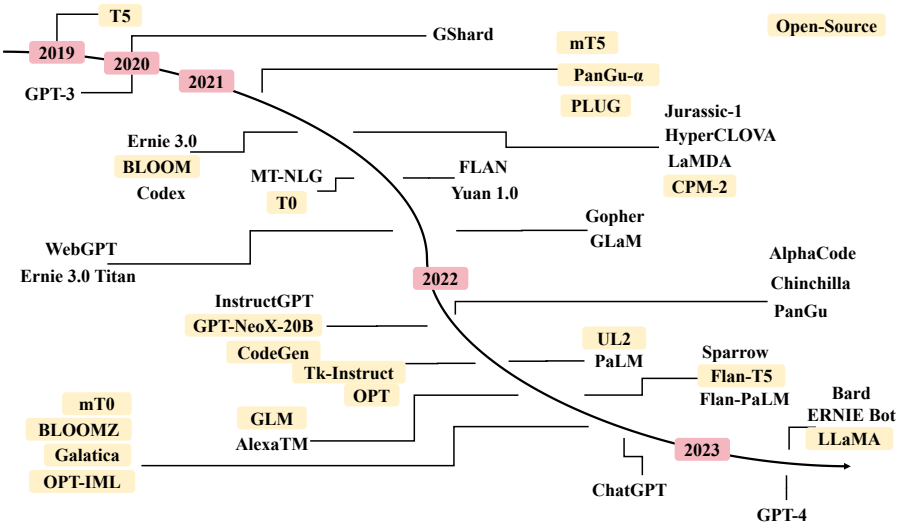

Figure 1. The depiction illustrates the development trajectory of Large Language Models (LLMs) from 2019 to 2023, meticulously documenting the introduction of significant models and technologies during this period. This includes the launch of models, from T5 and GPT-3 to the latest GPT-4, LLaMA, demonstrating the continual progress and evolution of large language models in the field of Natural Language Processing (NLP). Moreover, the illustration includes models optimized for specific domains and tasks, such as Codex, AlphaCode, reflecting the expansion and deepening of large language model technology in multiple directions.

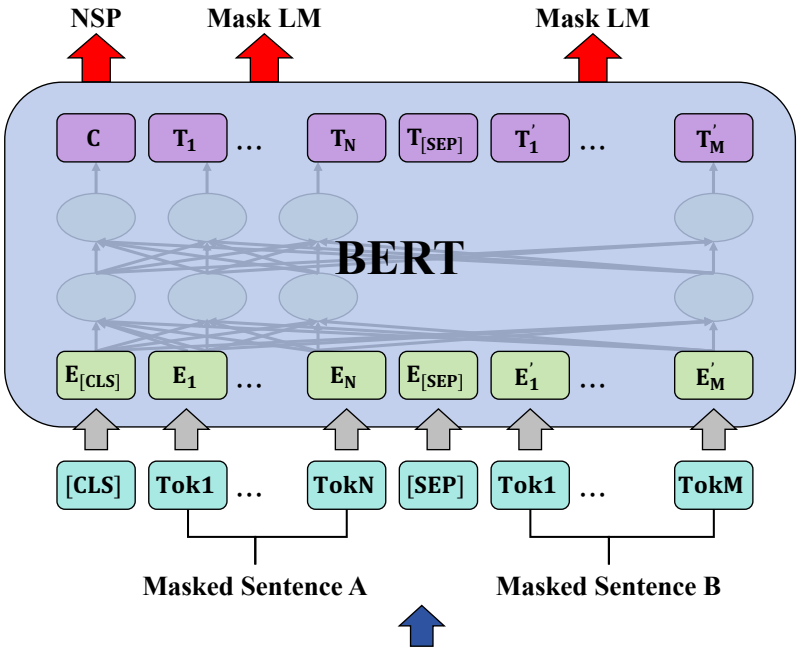

Figure 2. The depiction reveals the fundamental architecture of the BERT model, which encompasses the Masked Language Model (MLM) and Next Sentence Prediction (NSP) pretraining tasks. It can be seen that Sentence A and Sentence B are connected through the special tokens [CLS] and [SEP], along with the process of certain vocabulary being masked, with the aim of training the model to capture deep linguistic features and understand the relationships between sentences.

Its innovation lies in the use of a bidirectional Transformer encoder. BERT's training process includes two stages: pretraining and fine-tuning. During the pretraining stage, BERT learns deep representations of language through the Masked Language Model (MLM) and Next Sentence Prediction (NSP) tasks. The MLM task involves predicting randomly masked tokens, while the NSP task predicts whether two sentences are consecutive. Unlike BERT, GPT adopts a pretraining plus fine-tuning framework [26]; however, as an autoregressive model, it uses all previous words as context when generating each word. The key to GPT is its extensive pretraining on a large corpus, followed by fine-tuning on specific tasks, leading to significant generalization capabilities.

Large language models such as BERT and GPT learn rich linguistic features and knowledge information by pretraining on large-scale text data [27]. These models can understand semantic information in text data and capture complex contextual relationships, offering potential for in-depth analysis of network attack behaviors. Specifically, large language models have shown unique advantages in the network attack behavior detection application. For instance, in the analysis of malicious software codes, large language models can conduct an in-depth semantic analysis of the code, effectively identifying hidden malicious behavior patterns [28]. Traditional malware detection methods often rely on specific feature codes or behavior patterns, which may fail when malware authors modify the code through obfuscation, encryption, or variation techniques. In contrast, large language models, by learning the semantic differences between malicious and normal codes, can identify even highly mutated malware. For example, by comparing the structure and semantic features of malicious and normal codes, models can effectively distinguish and label potential malicious codes, even those disguised through complex transformation and obfuscation techniques. Similarly, in the analysis of network traffic and logs [29], large language models demonstrate strong application potential. Attacks such as DDoS and phishing attacks [30] often leave abnormal behavior patterns in network traffic or server logs. By conducting deep learning and semantic analysis on these log data, large language models can quickly identify key information related to attack behaviors from massive data, such as abnormal access frequencies and suspicious request parameters. This deep learning-based approach is more flexible and effective than traditional rule-based detection methods, adapting to the rapid changes and evolution of network attack behaviors. Additionally, for specific types of network attacks such as SQL injection [31] and cross-site scripting (XSS) attacks [32], large language models can provide technical support for precise detection and defense by learning the characteristics and patterns of these attacks. For example, in detecting SQL injection attacks, models can identify illegal SQL operations by analyzing the SQL statements in query logs, effectively blocking malicious database access requests. This method improves detection accuracy while significantly reducing false positives.

These applications demonstrate the potential and value of large language models in detecting network attack behaviors, offering more fine-grained in-depth analysis than traditional methods. However, applying large language models to network attack behavior detection faces challenges. On one hand, data in the cybersecurity field typically exhibit high specificity, requiring models to adapt to specific application scenarios; on the other, the constantly changing patterns of network attacks require that models be able to adapt to new attack methods in a timely fashion.

A key step in applying large language models to network attack behavior detection is designing appropriate input representations and pretraining tasks that meet specific detection needs. Different detection scenarios necessitate different forms of input representation. For example, in analyzing malicious codes, the input might be a token sequence of the source code, while for log file analysis the input could be a text sequence of log records. In order to adapt these inputs for model processing, they first need to be converted into vector forms understandable by the model, typically achieved through word embedding techniques. The process of word embedding can be mathematically expressed as folows:

$$\mathbf{E} = \text{Embedding}(\mathbf{X}). \tag{2}$$

Here, **X** represents the input sequence and **E** denotes the vector representation after word embedding. Designing appropriate pretraining tasks is crucial in order for large language models to learn valuable knowledge for attack behavior detection. Self-supervised learning tasks such as the MLM task are a common and effective choice:

$$\mathcal{L}_{\text{MLM}} = -\sum_{i \in \mathcal{M}} \log p(x_i | x_{\setminus i}; \Theta). \tag{3}$$

Here, $\mathcal{M}$ is the index set of tokens selected for masking, x_i represents the i-th token in the sequence, $x_{\setminus i}$ denotes the other tokens excluding x_i, and Θ refers to the model parameters. The MLM task promotes the model's ability to capture and learn the deep semantic information of text by predicting the masked tokens in the sequence. After the pretraining stage, fine-tuning the model on specific attack behavior detection tasks is a necessary step. The fine-tuning process mainly involves optimizing the following loss function:

$$\mathcal{L} = -\sum_{(x,y) \in \mathcal{D}} \log p(y | x; \Theta'). \tag{4}$$

Here, (x, y) represents the labeled sample pair, $\mathcal{D}$ is the training dataset, y is the label of the sample, indicating whether the sample is "normal" or "attack" behavior, and Θ' are the fine-tuned model parameters. Through this step, the model learns how to predict the corresponding label y based on the input data x, thereby achieving accurate detection of network attack behaviors. As discussed, applying large language models to network attack behavior detection requires careful design of input representations and pretraining tasks as well as effective fine-tuning on specific detection tasks [33]. These steps collectively ensure that the model can fully leverage its language understanding capability to accurately identify and analyze potential network attack behaviors. To overcome these challenges, researchers must meticulously adjust the model's pretraining and fine-tuning strategies to ensure that it captures the specific knowledge and language patterns of the cybersecurity field. Additionally, incorporating domain experts' knowledge into model training and evaluation is key to enhancing model performance. Lastly, in light of the severity of cybersecurity incidents, the interpretability and trustworthiness of models are non-negligible factors. Researchers need to develop new methods to ensure that the model's decision-making process is transparent and can provide reliable evidence to support its detection results.

In summary, the development of large language models offers new technical means and research directions for network attack behavior detection. By delving into these models' potential and continuously adapting to the specific needs of the cybersecurity field, detection capabilities with respect to network attack behaviors can be effectively enhanced, making a significant contribution to maintaining network security. As model design, training strategies, and application methods continue to innovate and optimize, future large language models will have even broader prospects in the field of network attack behavior detection.

2.2. U-Net

The U-Net model was initially introduced by Olaf Ronneberger, Philipp Fischer, and Thomas Brox in their 2015 paper on biomedical image segmentation [34]. Renowned for its unique symmetry and the use of skip connections, this model has achieved remarkable success across various fields, particularly in medical image processing. The core design principle of U-Net involves an intricate upsampling process and the incorporation of high-resolution features from previous layers through skip connections, ensuring that information is preserved in the deeper layers of the network, which is crucial for precise segmentation of detail-rich areas within images. The U-Net architecture features a symmetric "U" shape, divided into two parts: the contraction path and the expansion path. The contraction path, also known as the encoder, progressively reduces the spatial dimensions of the input image while increasing the number of feature channels. This

process involves convolutional layers and max pooling steps, aimed at capturing abstract features of the image and reducing computational complexity. Conversely, the expansion path, or decoder, gradually restores the spatial dimensions through transposed convolution (upsampling) while decreasing the number of feature channels. Each upsampling step is accompanied by skip connections from corresponding downsampling steps, allowing the network to combine high-resolution features with upscaled features, thereby effectively restoring details and local structures of the image.

A key advantage of the U-Net model is its ability to achieve highly precise results with limited data. This is particularly important for medical image segmentation tasks [35], where high-quality annotated data are often scarce. U-Net addresses data scarcity by effectively utilizing data augmentation techniques such as rotation, scaling, and elastic deformations, significantly enhancing the model's generalization capabilities. Additionally, U-Net excels in handling multi-task learning and multimodal inputs; for example, in processing MRI and CT images, U-Net can integrate information from diverse sources and output precise segmentation through a unified framework. This capability means that U-Net is not limited to medical image processing; for instance, it has found applications in satellite imagery, road detection, and plant disease detection in agriculture. Although initially designed for image segmentation, U-Net's powerful feature extraction and information restoration capabilities make it suitable for a broader range of applications, including detecting cyberattack behaviors. In such tasks, U-Net can assist models in learning and identifying complex patterns in network traffic data or log files, which are often related to anomalous behaviors. By transforming dispersed unstructured data from network logs into meaningful feature representations, U-Net plays a crucial role in the early identification of potential network threats. In the field of network security, the application of U-Net introduces a novel approach, namely, leveraging its validated structure in the domain of imagery to parse and understand network behavior data akin to "images". For instance, time series data of network traffic can be treated as "images" and processed by U-Net to identify and classify various types of network attacks, such as DDoS and SQL injection. This approach can enhance detection precision while maintaining high computational efficiency and low false positive rates when handling complex data.

2.3. Attention Mechanism

The proposal of the attention mechanism in the deep learning field was inspired by the human visual attention mechanism [36]. The basic idea is that the model can pay more attention to the important parts of the input data while paying less attention to the less important parts during information processing. This mechanism enables the model to dynamically adjust its focus on the input data, thereby improving processing efficiency and enhancing model performance and accuracy. In deep learning models, especially when processing sequence data, the attention mechanism is implemented by calculating the mutual weights between input features to reflect their importance. The initial attention mechanism can be expressed as shown in Equation (1) and Figure 3.

One of the key challenges in detecting network attack behaviors is accurately and quickly identifying attack activities from vast amounts of network data [37]. These data include, but are not limited to, server logs, network traffic, and user behavior data, which contain both normal business activity information and potential malicious attack information. The attention mechanism, by providing models with the ability to filter and focus on key information, greatly enhances detection accuracy and efficiency. For instance, in analyzing network traffic data models need to identify which traffic patterns are normal and which may belong to DDoS attacks [38]. By incorporating an attention mechanism, models can focus more on key features such as abnormal access frequency and request sources, thereby improving their identification accuracy. Specific attention models can be designed for scenarios involving network attack behavior detection; for example, models employing self-attention mechanisms [39] are particularly suitable for processing sequence data such as log files. In this scenario, Q, K, and V might represent different representations

of the same log data sequence, allowing the model to learn internal dependencies in the sequence in order to identify potential attack patterns. Furthermore, for complex attack behavior detection tasks, multi-head attention mechanisms can be designed to further enhance model performance:

$$\text{Multi-Head}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_n)W^O, \quad (5)$$

$$\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \quad (6)$$

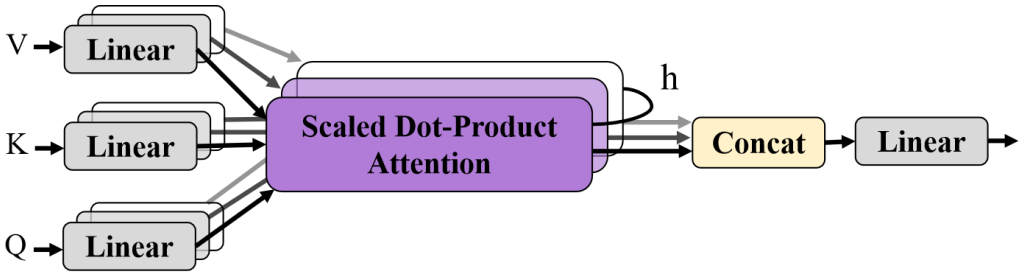

Figure 3. The depiction presents the core mechanism of the Transformer model structure with Scaled Dot-Product Attention. Detailed here is the flow of information through linear transformation layers, processing by the attention mechanism's weighted sum, followed by concatenation and another linear layer, culminating in the transformation of the input information. This structure forms the basis for the Transformer model's ability to capture the complex relationships between elements within a sequence.

Here, W_i^Q, W_i^K, W_i^V are different weight matrices used to project Q, K, V into different representation spaces, with W^O being the weight matrix of the output layer. Through this method, the model can learn different aspects of the input data in parallel in multiple representation spaces, capturing richer information to improve detection accuracy and the ability to identify complex attack behaviors. The attention mechanism shows strong application potential in detecting specific types of network attacks, such as SQL injection and XSS attacks [40,41]. Through deep learning of the characteristics of attack behaviors, models can accurately identify key operations and patterns in attacks. For example, when detecting SQL injection attacks, models can identify atypical query patterns or structurally abnormal query statements by learning the differences between normal and malicious SQL queries, thereby effectively identifying and blocking SQL injection attacks. Moreover, models can accurately identify attack behaviors in more complex scenarios, such as phishing and malware dissemination, as the attention mechanism dynamically adjusts the model's focus to accurately identify traces of attack behaviors from extensive network communication data [42,43]. For instance, in phishing email detection, models can focus on deceptive vocabulary or sentence structures in email content to effectively identify and intercept phishing emails.

The introduction of the attention mechanism provides a highly efficient, flexible, and powerful technical means for detecting network attack behaviors. By dynamically adjusting focus, such models have improved accuracy when recognizing network attack features and are able to adapt to the continuous evolution and changes in network attack methods. In the future, network attack behavior detection methods based on attention mechanisms can be expected to demonstrate even more outstanding performance, contributing greater strength to network security protection.

3. Materials and Method

3.1. Dataset Collection

In this study, a multi-source dataset approach was adopted to support the effective detection of various network attack behaviors. The collection of these datasets encompassed data from web crawls, server logs, financial transaction behavior data, and comments from

social media and forums, aiming to construct a comprehensive multi-dimensional data foundation for more accurate identification and prediction of network attack behaviors.

A self-developed crawler program was utilized to systematically collect a large amount of publicly available online data within a specific time window. Primarily, these data included contents from malicious websites and malware distribution sites. The intention was to capture information related to phishing and malware dissemination. Data collection spanned from January to June 2023, with the collection devices being servers equipped with the Linux operating system and the crawler program mainly developed in Python 3.8. Through collaboration with multiple partners, access logs from their servers were obtained. These logs recorded server access information from January to June 2023, including access times, IP addresses, and request contents, and were used for training and testing the model's performance in detecting server attacks such as SQL injection and cross-site scripting attacks. Financial transaction data from internet finance platforms were acquired, including transaction times, amounts, and types of transactions. These data were primarily used for identifying and detecting financial fraud behaviors such as credit card fraud. Data collection covered the entire year of 2023, aiming to capture potential fraudulent transactions. Comments from various social media platforms and forums were also collected, reflecting user interactions and expressions in the online environment; special attention was paid to comments containing malicious content, such as cyberbullying and hate speech. The collection period for comment data was from January to June 2023, with batch downloading and storage facilitated by self-developed data collection tools. To specifically illustrate the scale and distribution of the data, Table 1 displays the basic information of each type of dataset:

1. **Server Log Dataset:** The server log dataset included detailed records of server accesses, which were primarily utilized for the detection of server attacks such as SQL injections and Cross-Site Scripting (XSS) attacks. Access logs from servers have been obtained through collaboration with multiple partners.
   SQL Injection: Attempts by attackers to gain database privileges by entering SQL commands or modify database information, for example, "105 OR $1 = 1$".
   Cross-Site Scripting (XSS): Malicious scripts are injected into web pages by attackers. When other users visit these web pages, the scripts are executed, for example, "$< script > alert('XSS'); < /script >$".
2. **Financial Behavior Dataset:** Data collected from internet financial platforms, including transaction times, amounts, and types, were primarily used to identify and detect financial fraud such as credit card fraud.
   Credit Card Fraud: Unlawful transactions are carried out by criminals who have stolen credit card information, shown by transaction records indicating that a card was used for high-value transactions in different countries within a short time.
3. **Social Media Comment Dataset:** Comments were collected from various social media platforms and forums, with a particular focus on those containing malicious content such as cyberbullying and hate speech.
   Cyberbullying: Malicious attacks on individuals via social media comments were identified, for example, "$Youarereallystupid$!".
   Hate Speech: Comments that exhibit racial or gender discrimination, for example, "$Wedonotwelcomeyourkindhere$!".

Table 1. Details of dataset collection.

Data Type	Amount	Characteristics
Web Crawl Data	100,000	Including malicious website contents, malware distribution sites, etc.
Server Logs	500,000	Records server access information, used for detecting server attacks.
Financial Behavior	200,000	Transaction times, amounts, types, used for detecting financial fraud.
Comment Data	300,000	User comments from social media and forums.

3.2. Dataset Annotation and Preprocessing

Standardization and vectorization of data are key steps to ensure effective learning by the model. All collected data underwent appropriate preprocessing to convert them into forms suitable for processing by deep learning models. For text data, in addition to preprocessing steps, advanced word embedding techniques were employed to convert text into dense vectors, capturing the deep semantic information of the text:

$$\mathbf{v}_{\text{text}} = \text{WordEmbedding}(\text{Preprocessing}(\text{Text})). \tag{7}$$

For numerical and categorical data, one-hot encoding and standardization were applied to ensure unbiased information learning from each feature by the model. By collecting data from multiple channels and meticulous preprocessing and standardization, a comprehensive and multi-dimensional dataset was constructed, providing a solid foundation for detecting network attack behaviors. This process focused on the accumulation of data volume as well as on enhancing data quality and deep mining of data features with the aim of effectively improving the accuracy and efficiency of network attack behavior detection through a data-driven approach.

3.2.1. Dataset Annotation

Before the start of the annotation work, all team members underwent a training series which included the use of annotation tools, understanding annotation rules, and practice with actual annotation operations. This training process ensured that each annotator could accurately understand the requirements of the annotation task and how to apply the annotation rules. To ensure the accuracy and consistency of data annotation, a detailed set of annotation rules was established. These rules were based on the best practices and latest research findings in the field of cybersecurity, covering scenarios from simple network access behaviors to complex financial fraud and malicious software attacks. For example, any abnormal access pattern in server log data, such as a large number of requests in a short time or abnormal request paths, was marked as an "attack". For financial behavior data, transactions were judged as "credit card fraud" based on transaction frequency, amount, and location. In the actual annotation process, each piece of data was independently reviewed and annotated by at least two annotators in order to reduce subjective bias and improve annotation accuracy. The annotation results were determined by majority vote, that is, when the majority of annotators' results were consistent, that result was determined as the final annotation. To ensure consistency and accuracy of the annotation standards, the following measures were taken:

1. Regular Review: Annotation teams held regular review meetings to discuss and resolve issues and disagreements encountered during the annotation process. This helped to continuously refine the annotation rules, ensuring a unified understanding of the annotation standards among all personnel.
2. Double-Blind Annotation: A double-blind mechanism was implemented during the annotation process, meaning that each annotator could not see the results of others, reducing mutual influence and ensuring the objectivity and independence of the annotation.
3. Quality Control: A quality control step was introduced in which annotation results were regularly evaluated through random checks. Problems identified during checks were corrected and feedback was provided promptly, ensuring annotation quality.

3.2.2. Dataset Preprocessing

For the study of detecting network attack behaviors, data preprocessing is a crucial step that directly impacts the efficiency of model training and the final performance of the model. The purpose of data preprocessing is to convert raw data into formats more suitable for machine learning models while improving data quality and minimizing noise interference. Data cleaning primarily involves removing irrelevant, erroneous, or duplicate

data from the dataset. For example, server logs may contain a large amount of normal access records that are not helpful for attack behavior detection and need to be removed. For text data, preprocessing steps include tokenization and removal of stop words. Tokenization refers to the process of splitting continuous text strings into sequences of tokens, which is foundational for text processing. The removal of stop words eliminates common but meaningless words from the text, such as "the" and "is". The aim of these steps is to reduce data dimensionality and alleviate the burden on the model. Subsequently, the TF-IDF (Term Frequency-Inverse Document Frequency) technique is employed to convert text into vector form. TF-IDF is a statistical method used to evaluate the importance of a word to a document in a corpus. Term Frequency (TF) indicates the frequency of a word in a document, while Inverse Document Frequency (IDF) is a measure of the general importance of the term. The TF-IDF value can be calculated using the following formula:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t) \tag{8}$$

where t is a term and d is a document.

$$\text{TF}(t, d) = \frac{\text{Number of occurrences of term } t \text{ in document } d}{\text{Total number of terms in document } d} \tag{9}$$

$$\text{IDF}(t) = \log \frac{\text{Total number of documents in the corpus}}{\text{Number of documents containing term } t + 1} \tag{10}$$

By calculating TF-IDF values, text data can be converted into vector form, where each dimension represents the TF-IDF value of a term, providing input data for subsequent deep learning model training. Standardization is an essential step for numerical data such as financial transaction amounts and server access frequencies. The purpose of standardization is to transform numerical data of different scales into a uniform scale, making it more suitable for model processing. The standardization method used in this paper is Z-score standardization, with the formula as follows:

$$Z = \frac{(X - \mu)}{\sigma} \tag{11}$$

where X is the original data, μ is the mean, σ is the standard deviation, and Z is the standardized data. This method adjusts the scale and distribution of the data, making it more suitable for processing by machine learning models. For categorical data such as transaction types and request methods, vectorization is necessary to convert the categorical data into numerical data recognizable by models. The method used in this paper is one-hot encoding, which converts each category value into a binary vector, with only one position in the vector having a value of 1 and the rest having values of 0. The vectorization process of one-hot encoding can be mathematically represented as

$$\text{One-Hot}(x_i) = [0, \ldots, 1, \ldots, 0], \tag{12}$$

where x_i is the original categorical data and $\text{One-Hot}(x_i)$ is the vector after one-hot encoding, with the position of 1 indicating the category. After completing data preprocessing, it is necessary to divide the dataset into training, validation, and test sets. This step is taken to evaluate model performance and prevent model overfitting. Dataset division typically relies on random sampling methods to ensure consistency in data distribution.

3.3. Proposed Method

3.3.1. Overview

In this study, a novel method for detecting network attack behaviors based on LLMs and synchronized attention mechanisms is proposed. The aim is to enhance the accuracy and efficiency of detecting network attack behaviors by delving into the deep semantic information and contextual relationships within text data. The method is grounded on the utilization of LLMs and synchronized attention mechanisms, as illustrated in Figure 4.

In the presented methodological framework, the data flow initiates with data input, progresses through various processing levels, and culminates in the output. Initially, data are input into the layers representing text and images, processed via the synchronized attention mechanism, and subsequently refined and integrated through a denoising U-Net to yield the final output. Large Language Models (LLMs) are tasked with processing and understanding textual data. This segment transforms textual data into a format that can be input to neural networks. The core process of the presented method comprises the following steps:

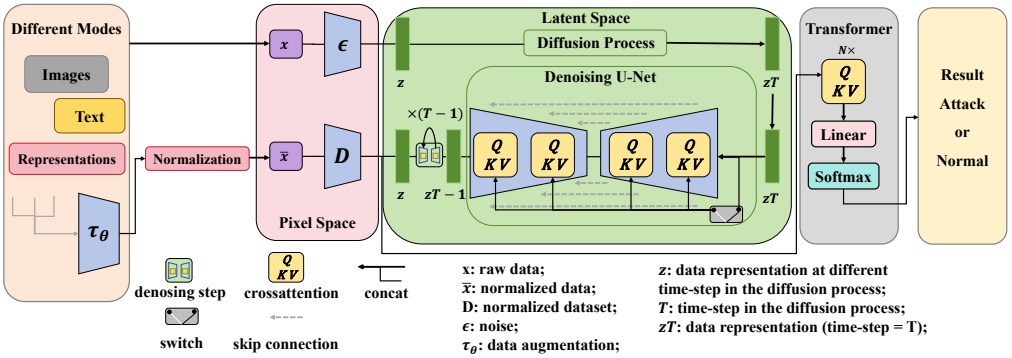

Figure 4. This figure displays an overall schematic of the model framework proposed in this paper, including the input processing of multimodal data, the application of the synchronized attention mechanism, and the integration of the denoising U-Net architecture. Initially, the raw data are enhanced (denoted as τ_θ in the figure), then subjected to regularization and noise addition processes (represented by $\bar{x}$ and ϵ in the figure), forming the model's input. The noised data z undergo processing through the diffusion process to obtain zT (where T represents the time steps in the diffusion process); subsequently, data zT are processed through the U-Net network for T steps of denoising to yield z'; finally, the regularized data $\bar{x}$ and z' are fed into the transformer for classification to produce the final result, which is the model's output.

1. Data preprocessing and vectorization: Collected multisource data undergo preprocessing, including data cleaning, deduplication, tokenization, etc., followed by transformation of text into vector form through advanced word embedding techniques, providing suitable input data for subsequent model training.
2. Application of large language models: Pretrained large language models are employed for deep feature extraction from vectorized data. These models comprehend the profound meanings and complex contextual relations in the text, offering robust technical support for detecting network attack behaviors.
3. Synchronized attention mechanism: The synchronized attention mechanism proposed in this paper enhances the model's focus on the characteristics of network attack behaviors by synchronously processing information within data blocks (intra-block) and across data blocks (inter-block). This mechanism allows the model to capture key information while considering the interactions and dependencies among the information, thereby improving the accuracy of detection.
4. Synchronized loss function: To further optimize the model's training process and enhance detection performance, a synchronized loss function is designed. This function synchronously considers the model's performance across different tasks to optimize and update the model parameters.

In Figure 4, "Diffusion" denotes a diffusion model, a generative model primarily utilized during the data generation process to restore the details and structure of signals by progressively reducing noise. Within the methodological framework of this study, diffusion models are incorporated to enhance the performance of detecting network attack behaviors, especially when addressing highly nonlinear and complex data features. The roles played by diffusion models in this framework are as follows. Data Enhancement and Recovery:

In the field of network security, attack data often appear sparse and incomplete; diffusion models, through their generative process, can simulate and augment real attack behavior data. By introducing controlled noise into the data and progressively reducing this noise throughout the generation process, the models learn to fill in missing portions of the data, thereby restoring attack behavior features more completely. Feature Extraction: During the feature extraction phase, diffusion models are applied to high-dimensional data, aiding in the capture of deep (often unobservable) features. This level of feature extraction is crucial for understanding complex attack patterns and behaviors, providing more precise information than traditional deep learning models. Denoising and Anomaly Detection: Noise data or false alarms are frequent issues in network attack detection. Diffusion models exhibit superior performance in denoising, effectively distinguishing and isolating the differences between real attack actions and noise through the model's reverse generation steps (transitioning from noisy data back to original data). Generating Adversarial Attack Samples: To enhance the model's robustness and defensive capabilities, diffusion models are also employed to generate adversarial attack samples. These samples simulate unseen and potential attack methods, aiding in training the model to predict and counter these potential new threats. The detailed operational procedures are as follows:

1. Initialization: An appropriate noise distribution is selected (represented by ϵ in the diagram) to serve as the starting point for the model (typically a Gaussian distribution).
2. Forward process: Noise is incrementally introduced into the actual data at each step, with each addition of noise calculated to ensure reversibility. This process generates a series of data states transitioning from entirely real to completely noisy.
3. Reverse process: Starting from the noisy data state, the model's learned inverse transformations are progressively applied, recovering increasingly clearer data states until a state close to the original data is reached.

During the application of this process, precise control over the amount and quality of noise at each step is essential if useful information is to be recovered from noisy data. Through this method, diffusion models enhance the accuracy of detection models, providing deeper insights and greater adaptability when dealing with complex data and attack patterns.

3.3.2. Large Model for Attack Behavior Detection

A model integrating synchronous attention mechanisms and a denoising U-Net architecture was developed to enhance the detection capabilities for complex cyberattack behaviors. Designed with multimodal inputs, this model is capable of processing various data types, including text and images, to utilize multisource information for attack detection comprehensively. The core components are as follows:

1. Input layer: The model accepts preprocessed multimodal data, such as text sequences transformed into vectors via word embedding and image data standardized.
2. Synchronous attention layer: Text data are processed using a self-attention mechanism, while image data are handled through a denoising U-Net architecture. The introduction of a synchronous mechanism is crucial, enabling the model to consider information from other modalities while processing one type of data modality. For instance, the text processing component relies on its sequence information as well as on relevant features from image data:

$$\mathrm{SA}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}} + S\right)V \tag{13}$$

   where Q, K, V represent the query, key, and value matrices, respectively, d_k denotes the dimension of the key vectors, and S is the synchronous signal from other modalities.
3. Denoising U-Net architecture: Image data undergo feature extraction and denoising using the U-Net structure, in which skip connections maintain the spatial information

of image features. The denoising step removes irrelevant noise, preserving key features of attack behaviors.

4. Fusion and classification layer: Processed text and image features are fused and then classified through a fully connected layer to determine the type of attack behavior.

The introduction of a synchronous attention mechanism enables the model to fully leverage complementary information from other modalities while processing one type of data, enhancing the detection capabilities for complex attack behaviors. The use of a denoising U-Net architecture capitalizes on its advantages in feature extraction and spatial information retention. Denoising further improves the model's feature quality and generalization capability. This model emphasizes synchronous processing of multimodal data and denoising, addressing aspects not covered by models that focus on a single modality or lack a synchronous processing mechanism. By combining synchronous attention with the denoising U-Net, the model is able to processes complex semantic information within textual data while efficiently utilizing spatial features from non-textual data, significantly enhancing its detection accuracy and robustness. Furthermore, adjusting the weight of the synchronous signal S allows for more effective integration of information from different modalities, improving recognition precision for attack behavior characteristics.

In summary, the large model proposed in this study demonstrates significant performance advantages on the task of network attack behavior detection, particularly through its unique design in synchronous processing of multimodal data and efficient denoising of image data. The application of these design concepts and methodologies enhances the model's effectiveness in the field of cybersecurity and provides valuable insights for handling other complex multimodal data analysis tasks.

3.3.3. Synchronous Attention Mechanism

The synchronous attention mechanism comprises two main parts, namely, intra-block attention and inter-block attention. These components work together to enable the model to simultaneously consider the association information with other data blocks while processing a single data block, such as text segments or image areas, as illustrated in Figure 5.

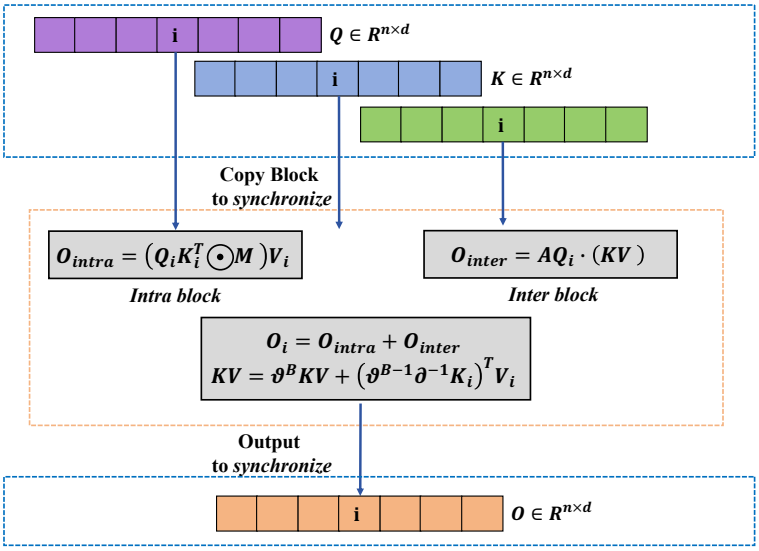

Figure 5. The depiction elaborates on the structure of the synchronized attention mechanism proposed in this paper, including the information processing flows within (intra-) and across (inter-)blocks. It demonstrates how the combination of self-attention operations within internal blocks and synchronized strategies across blocks enables the model to capture the characteristics of a single data block while integrating information from other blocks, thereby optimizing the detection performance of network attack behaviors.

Intra-block attention is primarily focused on the processing of information within a single data block. This mechanism employs a traditional self-attention mechanism to analyze and understand the relationships of features within the data block. Each data block is considered a subset of the input data, where the intra-block mechanism determines the importance of each element for the output by calculating the interactions among the elements within the data block. This usually involves calculating the dot product of the query and the key followed by normalization using the softmax function. This mechanism aids in capturing complex internal features such as local semantic relationships in textual data. The mathematical expression can be described as follows:

$$\text{Intra-Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{14}$$

where Q, K, and V respectively represent the query, key, and value matrices and d_k is the dimension of the key vectors.

Inter-block attention, unlike intra-block attention, is designed to enable the model to consider information from other data blocks while processing a particular data block. This mechanism assumes that there may be associations between different data blocks; if these associations are effectively utilized, they can greatly enhance the model's information processing capacity and decision accuracy. Inter-block mechanisms achieve this by transferring and synchronizing information between different data blocks, such as by weighted merging of feature representations from different blocks. The mathematical expression can be described as follows:

$$\text{Inter-Attention}(Q, K, V) = \text{softmax}\left(\frac{Q(K+S)^T}{\sqrt{d_k}}\right)V \tag{15}$$

where S represents the synchronization signal from other data blocks, which can be the feature representation of another block or specially processed information. Such attention mechanisms have the following advantages:

1. They enhance the model's global understanding ability. By synchronizing information between different data blocks, the model can achieve a more in-depth understanding on a global level, especially for processing cross-modal or cross-domain data, significantly improving the model's comprehensive performance.
2. They enhance the model's ability to capture complex data patterns. Complex cyberattack behaviors often involve multiple types of data and interactions at various levels. The synchronous attention mechanism helps the model to capture these complex data patterns and relationships, improving detection accuracy.
3. They improve the model's adaptability and flexibility. The mechanism allows the model to flexibly adjust the focus of attention and optimize the information processing flow based on task requirements and data characteristics, enhancing the model's adaptability and applicability in different scenarios.

Single block and parameter design: In the specific network design, the basic processing unit for each data block is implemented through a multi-layer stacked transformer network structure. Each transformer layer contains a multi-head self-attention mechanism and a feed-forward neural network, with these layers stacked in sequence to enhance the model's representational power. In our model, typically 6 to 12 layers of transformers are stacked, with the specific number of layers adjustable based on the complexity of the task and characteristics of the data. For specific settings of each transformer layer, the dimension of the model's hidden layers is usually set to 512, and the number of heads in the multi-head attention mechanism is set to 8, meaning that each head processes a dimension of 64. Furthermore, the inner dimension of the feed-forward networks is typically four times that of the hidden layer dimensions, i.e., 2048. This design ensures that the model has sufficient capability to capture and process complex data features.

3.3.4. Synchronous Loss Function

In deep learning and NLP tasks, the loss function is a crucial part of model training that defines the discrepancy between the model's output and the true labels. For models based on the transformer architecture, the cross-entropy loss is a common loss function that is widely applied in classification tasks. However, for specific tasks, including cyberattack behavior detection, more specialized loss functions are designed to improve model performance. The synchronous loss function described here aims to optimize the model's processing capability for complex tasks by synchronizing the loss incurred when processing multisource data. Compared to the cross-entropy loss, the synchronous loss function incorporates additional terms to harmonize the information processing and model learning synchronization between different data sources. Assuming that the model's predicted output is $\hat{y}$ and that the true label is y, the traditional cross-entropy loss is expressed as

$$L_{CE} = -\sum_{i} y_i \log(\hat{y}_i), \tag{16}$$

where i represents the index of categories and y_i and $\hat{y}_i$ are the probabilities of the true label and predicted output for the ith category, respectively. Building on this, the synchronous loss function introduces an additional synchronization term L_{sync} to enhance the model's synchronicity and coordination when processing different data sources. The design of the synchronization term is based on the consistency between the model output and the cross-data source synchronization signals, specifically expressed as follows:

$$L_{sync} = \sum_{j} \| F(\hat{y}_j) - S_j \|_2^2 \tag{17}$$

where j represents the index of different data sources, $F(\hat{y}_j)$ is the synchronization signal generated based on the model output $\hat{y}_j$, S_j is the actual synchronization signal from other data sources, and $\|\cdot\|_2^2$ denotes the L2 norm. Therefore, the total expression for the synchronous loss function is

$$L_{total} = L_{CE} + \lambda L_{sync}, \tag{18}$$

where λ is a tuning parameter used to balance the impact of the cross-entropy loss and the synchronization term. The design of the synchronous loss function is based on the following considerations:

1. It improves the model's synchronicity when processing multisource data. In complex tasks such as cyberattack behavior detection, different data sources may have inherent connections and complementary information. The L_{sync} term encourages the model to maintain consistency and coordination when processing different data sources, allowing it to make better use of cross-data source information.
2. It enhances the model's generalization ability by promoting consistency between the model output and cross-data source synchronization signals. This helps the model to capture the commonalities between different data sources, improving its generalization ability for unseen data.

Overall, the design of the synchronous loss function aims to enhance the model's capability to process complex multisource data, which is particularly suitable for tasks such as cyberattack behavior detection that require comprehensive utilization of various data sources. By introducing a synchronization term into the loss function, the model can better coordinate and synchronize the information processing of different data sources during learning, improving its detection accuracy and efficiency.

3.4. Experimental Design

3.4.1. Hardware and Software Platform

In this study, constructing a high-performance experimental environment was deemed crucial, as the training and testing of models for detecting cyberattack behaviors demands extensive computational resources and data processing capabilities. Consequently, a platform combining hardware and software was established to provide robust support tailored to the requirements of developing deep learning models. The hardware platform was based on a high-performance computing system, at the core of which lay multiple Intel Xeon processors (Intel, Beijing, China). These processors, known for their high core count, are capable of parallel processing of large datasets, offering strong support for complex computational tasks. Additionally, NVIDIA Tesla V100 (Nvidia, Beijing, China) graphic processing units (GPUs) were equipped. Renowned in the deep learning domain for their exceptional computational power, these GPUs significantly accelerate the training process, especially for deep neural network models characterized by complex structures and numerous parameters. The memory was configured to 256 GB RAM, allowing the system to efficiently handle large datasets without encountering data swapping issues due to insufficient memory, ensuring smooth data processing and model training. For the storage system, SSD drives were chosen as the data storage solution, as their rapid data read and write capabilities ensure efficient data processing, particularly during extensive input/output operations, significantly reducing waiting times.

On the software side, the experimental environment was installed with the Python 3.8 programming language. Python, favored in deep learning research for its concise syntax, powerful library support, and extensive community resources, became the language of choice. To support the development and training of complex models, TensorFlow 2.1.5 and PyTorch 1.8, two deep learning frameworks, were deployed in the environment. Developed by Google, TensorFlow is widely acclaimed in both industry and academia for its flexible architecture and robust ecosystem. PyTorch, favored for its intuitive API and dynamic computation graph feature, is particularly suitable for rapid prototyping and experimentation. These frameworks offer a wealth of tools for model construction and training; in addition, they support GPU acceleration, making full use of the computational resources of our hardware platform. For text data processing, two natural language processing libraries, NLTK (Natural Language Toolkit) and Spacy, were selected. NLTK, a powerful natural language processing toolkit for Python, provides a wide range of text processing libraries and data resources suitable for linguistic research and teaching. Spacy is widely used in the industrial and research sectors for its efficient text processing capabilities, especially excelling in text analysis and entity recognition. By integrating these software libraries and tools, efficient completion of tasks such as text data preprocessing and feature extraction was facilitated, providing a solid data foundation for model training. In summary, through meticulous hardware configuration and software selection, the experimental environment of this study offers robust support for deep learning research in detecting cyberattack behaviors, providing an efficient and stable platform for the current research task while reserving ample space and possibilities for future research exploration. Through such experimental design, it is anticipated that research progress in the field of cybersecurity will be propelled, contributing to the construction of a more secure cyber-environment.

3.4.2. Training Strategies

In this study, to ensure optimal training efficiency and effectiveness of the model for cyberattack behavior detection, a series of training strategies were employed encompassing aspects such as hyperparameter settings and model evaluation methods. Through systematic experimental design and parameter optimization, the aim was to explore the model's full potential, thereby enhancing its accuracy in detecting cyberattack behaviors. Regarding hyperparameter settings, the complexity of model training and the characteristics of the data were fully considered, employing various methods including grid search and cross-validation to find the optimal hyperparameter configuration. Tuning hyperparameters is crucial for the speed of model training, and directly impacts the model's generalization abil-

ity and final performance; hence, detailed research and experimentation were conducted on key hyperparameters of the model, including but not limited to the learning rate, batch size, and the number of units in the hidden layers, to ensure that the most suitable model configuration for this research task was found.

The learning rate, a critical parameter controlling the speed of model weight updates, directly affects the stability and convergence speed of model training. Preliminary experiments were conducted around several candidate values, including 0.001, 0.01, and 0.1. By observing the change in the model's loss function during training and its performance on the validation set, it was found that a learning rate of 0.001 allowed the model to converge more smoothly while avoiding the risk of overfitting. The choice of batch size is equally crucial, as it impacts both the training speed and the final performance of the model. After several rounds of experiments, it was determined that a batch size of 64 achieved a good balance between training efficiency and model performance with the current hardware configuration and data scale. For the number of units in the hidden layers, another important parameter determining model complexity, we experimented with configurations including 256, 512, and 1024. By comparing the performance of the model under different configurations, 512 units were ultimately chosen for the hidden layers, providing sufficient learning capacity for the model while avoiding overfitting caused by excessive complexity. For model evaluation, cross-validation was utilized to comprehensively assess the model's performance. Specifically, five-fold cross-validation was employed, meaning that the dataset was divided into five subsets, with each subset being used once as the test set while the remaining four subsets were used for training. This process was rotated five times, yielding five performance evaluation results. By calculating the average of these five evaluations, the final performance metric of the model was obtained. This evaluation method makes full use of limited data resources while effectively reducing the randomness of the model evaluation results, increasing the accuracy and reliability of the assessment.

Through the implementation of the aforementioned training strategies, the scientific and systematic nature of the model training process was ensured, laying a solid foundation for the effective detection of cyberattack behaviors. In subsequent experiments, we will continue with further exploration into additional parameter optimization techniques and evaluation methods to enhance model performance. Additionally, close attention will be paid to various indicators during the model training process, such as the trend of the loss function and performance on the validation set, to ensure that the model completes training in the best possible state. For issues that may arise during model training, such as overfitting or underfitting, effective solutions will be sought through adjusting the model structure, introducing regularization techniques, and adjusting the learning rate. The generalization ability of the model was emphasized during the experimental process. To prevent a decline in generalization performance due to the model being overly dependent on the training set data, techniques such as early stopping were employed. In order to avoid overfitting, training was halted when the model's performance on the validation set ceased to improve or even began to decline after a certain number of training epochs. This method effectively preserves the model's generalization ability, ensuring good performance on unseen data.

3.4.3. Performance Metrics

In this research, a comprehensive evaluation of the model's performance in detecting cyberattack behaviors was conducted by selecting and applying a series of key performance metrics, including Precision, Recall, and Accuracy. These metrics assess the model's performance from different dimensions, reflecting not only the accuracy of the model in detecting attack behaviors but also its ability to recognize true attack actions. Precision, also known as positive predictive value, is an important metric measuring the accuracy of the model's predictions. It indicates the proportion of samples correctly identified as attack behaviors out of all samples identified as such by the model. The formula for Precision is as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{19}$$

where TP (True Positives) denotes the number of true positive cases, i.e., the samples correctly identified as attack behaviors by the model, while FP (False Positives) denotes the number of false positive cases, i.e., the samples incorrectly identified as attack behaviors. The higher the Precision, the more accurate the model is in identifying attack behaviors, with fewer false positives. Recall, also known as sensitivity, is another crucial metric measuring the model's coverage. It represents the proportion of true attack behavior samples correctly identified by the model out of all actual attack behavior samples. The formula for Recall is as follows:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{20}$$

where FN (False Negatives) represents the number of false negative cases, i.e., the samples incorrectly identified as normal behavior by the model. The higher the Recall, the more complete the model is in identifying all attack behaviors, with fewer omissions. Accuracy is the most intuitive performance metric, representing the proportion of samples correctly identified by the model (including both attack and normal behaviors) out of the total samples. The formula for Accuracy is as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{21}$$

where TN (True Negatives) represents the number of true negative cases, i.e., the samples correctly identified as normal behavior. The higher the Accuracy, the stronger the model's overall identification ability. These evaluation metrics are significant for assessing model performance. Precision and Recall evaluate the model's ability to recognize attack behaviors from different perspectives, one focusing on prediction accuracy and the other on prediction completeness. In practical applications, depending on the requirements of different scenarios, more emphasis may be placed on one of these metrics. For instance, in scenarios highly sensitive to false positives, Precision may be prioritized, while in scenarios where missing any attack behavior is critical, Recall may be emphasized. Accuracy, as a comprehensive metric, intuitively reflects the model's performance across all samples and is crucial for assessing the model's overall performance. However, when dealing with imbalanced datasets, where there is a significant difference in the number of attack behavior samples compared to normal behavior samples, relying solely on Accuracy may be misleading. In such cases, even if the model classifies all samples as the more numerous class, it may still achieve high Accuracy; however, this does not indicate good recognition ability. Therefore, when evaluating model performance, rather than depending solely on Accuracy, other metrics such as Precision and Recall should be considered as well.

To find a balance between Precision and Recall, the F1 Score was introduced as a performance evaluation metric. The F1 Score is the harmonic mean of Precision and Recall, considering both the accuracy and completeness of the model, and is calculated as follows:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{22}$$

The higher the F1 Score, the better the balance between Precision and Recall, indicating better overall performance. In practice, depending on different business needs, a preference may exist for models with higher Precision or Recall; the F1 Score provides a metric to weigh these factors, aiding researchers and engineers in making more rational choices. During the implementation of model performance evaluation, dataset preprocessing and division were first carried out to ensure the consistency of the distribution between the training and testing sets, thereby reducing bias in the evaluation process. The stability and reliability of the evaluation results were further enhanced through methods such as cross-validation. The model's performance on various metrics was recorded over multiple

rounds of experiments while analyzing the impact of different parameter configurations and model structures on performance along with the relationships and balances among Precision, Recall, and Accuracy. Through these thorough evaluation processes, the model's effectiveness in detecting cyberattack behaviors was verified and a deep understanding of the model's applicability and limitations in different scenarios was gained. The application of these evaluation metrics enabled a comprehensive and objective assessment of the model's performance, providing a solid foundation for future optimization and application of the model.

3.4.4. Baseline Models

In this study, several large models widely recognized in the field of artificial intelligence were selected as baseline models for comparison in order to comprehensively verify the effectiveness and superiority of the proposed model for detecting cyberattack behaviors. These baseline models, covering different architectures and learning mechanisms, have demonstrated their powerful performance on multiple NLP tasks. Comparison with these models allows for a more objective assessment of our model's performance in detecting cyberattack behaviors.

First, the OPT-175B model proposed by OpenAI is discussed [44]. OPT-175B, a large language model based on the transformer architecture, boasts 175 billion parameters, representing the cutting edge of current deep learning technology. Through pretraining on a vast scale of text data, OPT-175B has learned rich language representation capabilities, effectively completing a variety of NLP tasks such as text generation, understanding, and translation. The core of the OPT-175B model is the transformer architecture, which utilizes the self-attention mechanism to capture long-distance dependencies within input sequences, significantly enhancing the model's ability to process sequential data. Next, the LLaMa model [45] proposed by Facebook is mentioned. LLaMa, also a transformer-based large language model, has shown excellent multi-task learning capabilities through pretraining across various languages and tasks. LLaMa particularly emphasizes fine-tuning performance on limited data, achieving rapid adaptation and performance improvement on multiple downstream tasks. When discussing transformer-based models [46], the transformer model itself, introduced by Vaswani et al. in 2017, must be acknowledged. It was the first to introduce the self-attention mechanism, revolutionizing the approach to sequence modeling and laying the foundation for subsequent developments in deep learning models. The success of the transformer model lies in its ability to process all elements in a sequence in parallel, significantly improving the model's learning efficiency and performance. The BERT model [25], another successful application of the transformer architecture, was introduced by Google in 2018. Through the Masked Language Model and Next Sentence Prediction pretraining tasks, BERT has learned powerful language representation capabilities. The advent of BERT initiated the pretraining revolution in the NLP field, greatly advancing natural language processing technology. Lastly, ChatGLM3-6B [47] is a large model specially designed for chat and dialogue systems. It has learned the ability to understand and generate natural language dialogue through pretraining on a large corpus of dialogue data. ChatGLM3-6B has shown outstanding performance in handling dialogue and interactive tasks, providing strong technical support for building intelligent dialogue systems.

By comparing our model with these baseline models, not only can the absolute performance of our model in detecting cyberattack behaviors be assessed, its the relative performance across multiple dimensions can be understood as well, revealing the model's strengths and limitations in handling specific types of data and tasks. This comparison can help to validate the effectiveness of our model while providing important references for future model improvements. In conducting the model comparison, attention is paid to the overall performance of the models on the dataset as well as to their performance differences in handling various types of cyberattack behaviors. For example, certain models may excel in detecting malicious software distribution behaviors but perform less well in identifying

phishing websites. By analyzing these differences in depth, a better understanding of each model's characteristics can be achieved, allowing for the targeted selection or design of models to address specific security threats.

4. Results and Discussion

4.1. Detection Results for Cyberattack Behaviors

This study conducted experiments on three distinct datasets, namely, server logs, financial behaviors, and comment data, to validate the effectiveness of the proposed method in detecting cyberattack behaviors. The experimental design aimed to demonstrate whether the method combining synchronous attention mechanisms with large language models could achieve higher Precision, Recall, and Accuracy compared to existing advanced models such as Transformer, BERT, OPT-175B, LLaMa, and ChatGLM3-6B, indicate more effective identification and classification of cyberattack behaviors. The experimental results, as shown in Tables 2–4, indicate that the proposed method outperformed the others across all three datasets, achieving Precision, Recall, and Accuracy of 0.93, 0.91, and 0.92 on the server log dataset, 0.90, 0.87, and 0.89 on the financial behavior dataset, and 0.95, 0.93, and 0.94 on the comment dataset, respectively. In contrast, the other models (Transformer, BERT, OPT-175B, LLaMa, and ChatGLM3-6B), despite showcasing high performance, fell short of the proposed method across all evaluated metrics. These results distinctly illustrate the superiority of the proposed method in handling the task of cyberattack behavior detection.

Table 2. Attack detection results for server logs.

Model	Precision	Recall	Accuracy
Transformer	0.81	0.77	0.78
BERT	0.83	0.79	0.80
OPT-175B	0.85	0.82	0.83
LLaMA	0.88	0.85	0.86
ChatGLM3-6B	0.90	0.88	0.89
Ours	0.93	0.91	0.92

Table 3. Attack detection results for financial behaviors.

Model	Precision	Recall	Accuracy
Transformer	0.79	0.76	0.77
BERT	0.81	0.78	0.79
OPT-175B	0.83	0.80	0.81
LLaMA	0.85	0.82	0.84
ChatGLM3-6B	0.87	0.84	0.86
Ours	0.90	0.87	0.89

Table 4. Attack detection results for comment data.

Model	Precision	Recall	Accuracy
Transformer	0.85	0.82	0.84
BERT	0.87	0.84	0.86
OPT-175B	0.89	0.86	0.88
LLaMA	0.91	0.88	0.90
ChatGLM3-6B	0.93	0.90	0.92
Ours	0.95	0.93	0.94

The theoretical analysis of these experimental results is based on several points. First, large language models, particularly optimized ones such as BERT, LLaMa, and ChatGLM3-6B, have proven their powerful capabilities in understanding the deep meanings and contextual relations of text. However, by incorporating synchronous attention mechanisms, the proposed method has enhanced ability to capture key information and understand the interactions between different levels of information. This is especially effective in multimodal data processing scenarios, where it can more efficiently integrate information from various data sources, thereby improving the model's comprehensive judgment ability. Second, the design of the synchronous attention mechanism allows the model to consider

supplementary information from other data sources while capturing information from a single data source. This aspect is crucial in detecting cyberattack behaviors, as such behaviors often involve complex patterns and cross-verification of information from multiple sources. Additionally, the proposed method fine-tunes the model's focus on different tasks during the learning process through the design of a synchronous loss function, ensuring a better balance between synchronous processing and feature extraction, leading to significant improvements in Accuracy, Recall, and Precision. From a mathematical perspective, the synchronous attention mechanism, through its formula design, achieves effective synchronization and integration of information between different data blocks by adjusting the introduction of synchronous signals. This introduction of mathematical mechanisms enhances the model's deep understanding of individual data blocks and improves the model's grasp of the overall data structure and relationships across data blocks. Meanwhile, the synchronous loss function optimizes the management of information synchronicity and task relevance during the model training process through mathematical optimization, ensuring that the model can more accurately and comprehensively identify attack behaviors and characteristics in complex cyberattack detection tasks.

Specifically, on the server log dataset, our method successfully identified multiple complex SQL injection attempts by analyzing anomalous SQL query patterns, which some baseline models such as BERT and Transformer failed to detect due to their insufficient context correlation analysis capabilities. Additionally, on the comment dataset, our model used synchronized attention mechanisms to detect XSS attack scripts hidden within normal comments, attacks that are often overlooked by other models due to their covert textual representation. The experimental results and theoretical analysis across different datasets conclude that the proposed method leverages the deep text understanding capabilities of large language models and effectively enhances the performance of detecting cyberattack behaviors through the design of synchronous attention mechanisms and a synchronous loss function. This approach demonstrates a clear advantage, especially in multimodal data processing and the identification of complex attack patterns. These results validate the effectiveness of the proposed method and provide new insights and methodologies for subsequent research in related fields.

4.2. Ablation Experiments on Different Attention Mechanisms

Through the design of ablation experiments on various attention mechanisms, this study aimed to explore the effects of different attention mechanisms on the task of cyberattack behavior detection and their impact on model performance. The experiments compared the performance of self-attention, spatial attention, channel attention, and the proposed synchronized attention mechanisms on the same dataset, including the metrics of Precision, Recall, and Accuracy, in order to verify the enhancement effects of different attention mechanisms on the model's capability to recognize cyberattack behaviors. These experimental results visually demonstrated the contribution of different attention mechanisms to improving model performance in the task of detecting cyberattack behaviors, providing theoretical justification and empirical support for selecting suitable attention mechanisms.

From Table 5, it can be observed that different attention mechanisms have distinctly varied impacts on model performance. Specifically, the self-attention mechanism exhibited Precision, Recall, and Accuracy of 0.73, 0.70, and 0.71, respectively, the spatial attention mechanism 0.81, 0.76, and 0.78, the channel attention mechanism 0.89, 0.85, and 0.87, and the proposed synchronized attention mechanism the highest at 0.93, 0.89, and 0.91. This series of results clearly indicates that, as compared to self-attention, spatial attention, and channel attention mechanisms, the synchronized attention mechanism performed best on the task of cyberattack behavior detection. By calculating the relationships between elements within a sequence, the self-attention mechanism captures long-distance dependency information; however, it may overlook features in the spatial or channel dimensions. The spatial attention mechanism focuses on the spatial location information of feature

maps, aiding the model in concentrating on important features at spatial locations, whereas the channel attention mechanism focuses on the importance distribution across feature channels, enhancing the model's discriminative ability on feature channels. In contrast, the synchronized attention mechanism considers the interrelationships within the sequence while fully utilizing supplementary information between different data sources or blocks through the synchronization mechanism, achieving a more comprehensive and profound understanding of cyberattack behavior features.

Table 5. Results of different attention mechanisms in the ablation experiment.

Model	Precision	Recall	Accuracy
Self-Attention	0.73	0.70	0.71
Spacial Attention	0.81	0.76	0.78
Channel Attention	0.89	0.85	0.87
Synchronized Attention	0.93	0.89	0.91

4.3. Ablation Experiments on Different Loss Functions

Through the design of ablation experiments with different loss functions, this study aimed to explore the impact of various loss functions on model performance on the task of detecting cyberattack behaviors. By comparing the performance of traditional Cross-Entropy Loss, Focal Loss, MSE (Mean Squared Error Loss), and the proposed Synchronized Loss on the same dataset, the effects of these loss functions were assessed across the three dimensions of Precision, Recall, and Accuracy. The objective of the experimental design was to verify whether the proposed synchronized loss function could provide superior performance compared to other common loss functions on the specific task of cyberattack behavior detection.

As can be observed from Table 6, Synchronized Loss achieved the highest values in terms of Precision, Recall, and Accuracy, at 0.93, 0.89, and 0.91, respectively, significantly surpassing the other three loss functions. Cross-Entropy Loss showed the poorest performance, with Precision, Recall, and Accuracy of 0.69, 0.64, and 0.67, respectively. The performance of Focal Loss and MSE fell between the two, reaching scores of 0.79, 0.73, 0.76 and 0.86, 0.82, 0.84, respectively. These results indicate that for the task of cyberattack behavior detection, the proposed synchronized loss function can more effectively optimize the model, enhancing detection precision and robustness compared to other loss functions. Although cross-entropy loss, the most commonly used loss function for classification tasks, performs well in many scenarios, it may not offer sufficient performance optimization when faced with imbalanced datasets or tasks that require a finer delineation of differences between samples. Focal loss, by adjusting weights to address class imbalance, makes the model focus more on hard-to-classify samples, thereby improving the model's recognition capability for minority classes to some extent. Another commonly used loss function in regression problems, MSE shows some applicability in the task of cyberattack behavior detection, especially in providing more continuous gradient information when evaluating the difference between model outputs and actual labels. However, the proposed synchronized loss function, by introducing a synchronization term, considers both the prediction accuracy of individual samples and the synchronicity and consistency of the model when processing multi-source data. This design enables the model to more comprehensively capture and utilize information across data sources during the learning process, thereby improving the model's ability to recognize complex attack patterns.

Table 6. Results of different loss functions in the ablation experiment.

Model	Precision	Recall	Accuracy
Cross-Entropy Loss	0.69	0.64	0.67
Focal Loss	0.79	0.73	0.76
MSE	0.86	0.82	0.84
Synchronized Loss	0.93	0.89	0.91

4.4. Testing on a Different Dataset: Stock Price Prediction

In this section, the primary goal was to validate the generalization ability and adaptability of the proposed method for detecting cyberattack behaviors on different types of datasets. Applying the method to the domain of stock price prediction, which is significantly different from the original task, was intended to demonstrate the robustness and efficacy of the proposed method in dealing with various types of problems.

Table 7 shows that the proposed method achieved the highest Precision, Recall, and Accuracy at 0.91, 0.87, and 0.90, respectively, significantly outperforming the other models. Compared to the other models, the proposed method demonstrated the best performance, showcasing its excellent adaptability and generalization capability when dealing with different types of datasets. Among these, ChatGLM3-6B also displayed high performance, with Precision, Recall, and Accuracy at 0.88, 0.84, and 0.87, respectively. The baseline Transformer model exhibited the poorest performance, with Precision, Recall, and Accuracy at 0.78, 0.75, and 0.77, respectively. From a mathematical perspective, the proposed method, with its synchronous attention mechanism and loss function tailored for specific tasks, has significant advantages in capturing and processing complex data patterns. The synchronous attention mechanism effectively integrates multi-source information, enhancing the model's capability to capture factors influencing stock prices, while the specific loss function further optimizes the model's performance in stock price prediction tasks, leading to its superior performance in Precision, Recall, and Accuracy over other models. Furthermore, the design of the proposed method fully considers the temporality and dynamism of the data, improving the accuracy of the model in predicting stock price changes through refined feature extraction and information processing strategies. These experimental results indirectly reflect the importance of the proposed method's design philosophy and mathematical characteristics in enhancing model generalization capability and adapting to different task types, providing new perspectives and ideas for subsequent research in related fields.

Table 7. Stock Price Prediction Results.

Model	Precision	Recall	Accuracy
Transformer	0.78	0.75	0.77
BERT	0.80	0.77	0.79
OPT-175B	0.83	0.79	0.82
LLaMA	0.85	0.82	0.84
ChatGLM3-6B	0.88	0.84	0.87
Ours	0.91	0.87	0.90

4.5. Limitations and Future Work

The method proposed in this paper, based on large language models and synchronous attention mechanisms for detecting cyberattack behaviors, has been validated on multiple datasets for its effectiveness and superiority, especially in handling complex and multimodal data. However, despite the encouraging results achieved in this study, there remain a number of limitations and challenges for practical application and further research, which will be important directions for future work. First, although the synchronous attention mechanism effectively integrates and processes multimodal data, enhancing the model's detection capability for complex attack behaviors, its performance on extremely imbalanced datasets has not been fully verified. In the field of cybersecurity, attack events are often much less frequent than normal behaviors, and this class imbalance problem may affect the learning efficiency and detection performance of the model. Therefore, designing more effective data sampling strategies or optimization algorithms to improve the model's performance on imbalanced datasets is a critical issue for future research. Second, while the proposed method shows good generalization ability on certain specific datasets, such as the stock price prediction dataset, this does not mean that it will maintain the same performance across all types of tasks. The adaptability and stability of the model still need further exploration and validation, especially on tasks with high dynamics and uncertainty.

Thus, further enhancing the model's generalization capability and robustness to adapt to a broader range of application scenarios will be an important direction for future research. Lastly, the computational complexity of the proposed method is relatively high, especially when processing large-scale datasets, which may entail significant computational resource consumption and time costs.

5. Conclusions

In this study, we successfully developed a novel method based on large language models and synchronous attention mechanisms for detecting cyberattack behaviors. The method aimed to improve the accuracy and efficiency of cyberattack behavior detection by deeply mining the semantic information and contextual relationships in text data. Extensive experiments conducted on datasets consisting of server logs, financial behavior, and comment data demonstrated that the proposed method excels in key performance metrics such as Precision, Recall, and Accuracy.

On the server log dataset, the proposed method achieved a Precision of 0.93, Recall of 0.91, and Accuracy of 0.92, significantly higher than other models, including Transformer, BERT, OPT-175B, LLaMa, and ChatGLM3-6B. This result proves the efficiency and accuracy of the proposed method in handling the task of cyberattack behavior detection. Similarly, on the financial behavior dataset, the proposed method led with Precision of 0.90, Recall of 0.87, and Accuracy of 0.89, showcasing its powerful capability in capturing complex financial fraud behaviors. On the comment data dataset, the proposed method further confirmed its exceptional performance in text data analysis, with Precision of 0.95, Recall of 0.93, and Accuracy of 0.94. Additionally, our analysis of different attention mechanisms through ablation experiments revealed that the proposed synchronous attention mechanism played a key role in enhancing our model's performance. Compared to traditional self-attention, spatial attention, and channel attention mechanisms, the proposed synchronous attention mechanism demonstrated superior results in Precision, Recall, and Accuracy, reaching 0.93, 0.89, and 0.91, respectively. This further illustrates the significant effect of the synchronous attention mechanism in integrating and processing multimodal data, enhancing the model's ability to recognize features of cyberattack behaviors. In our ablation experiments on different loss functions, the proposed synchronized loss function, with its outstanding performance in Precision, Recall, and Accuracy (0.93, 0.89, and 0.91, respectively), highlighted its unique advantage over Cross-Entropy Loss, Focal Loss, and MSE in optimizing model performance. This result affirms the importance of the proposed synchronized loss function in the proposed method, and provides new insights into loss function design for future research. Moreover, testing on a stock price prediction dataset showcased our method's good generalization capability and adaptability, achieving satisfactory results in the financial domain, with Precision of 0.91, Recall of 0.87, and Accuracy of 0.90. This successful cross-domain application further proves the effectiveness and practicality of the proposed method. The integration of large language models with synchronous attention mechanisms marks a pioneering approach to enhancing the precision and efficiency of cyberattack detection. Extensive experimental validations across various datasets have shown our method's superior performance compared to existing models, highlighting its ability to handle complex detection tasks effectively. Additionally, the development of a synchronized loss function offers unique advantages in processing multi-source data, significantly improving the model's comprehensive recognition capabilities of cyberattack behaviors.

In summary, the method proposed in this paper for detecting cyberattack behaviors based on large language models and synchronous attention mechanisms has proven its significant advantages in improving the accuracy and efficiency of cyberattack behavior detection through experimental results on multiple datasets. By analyzing and processing complex text data in depth, the proposed method can effectively identify various cyberattack behaviors, providing strong technical support for the field of cybersecurity. This study showcases the important roles of the synchronous attention mechanism and syn-

chronized loss in optimizing model performance, offering new perspectives and methods for future research in this domain. Despite certain limitations, with further research and optimization the proposed method is expected to play a greater role in wider applications in cybersecurity and other fields, contributing to the construction of a safer and smarter cyber-environment.

Author Contributions: Conceptualization, Y.B., M.S., Y.Y. and C.L.; Data curation, L.Z., Y.W., S.L. and J.T.; Formal analysis, M.S., L.Z., S.L. and Y.L.; Funding acquisition, C.L.; Investigation, M.S. and J.T.; Methodology, Y.B. and C.L.; Project administration, C.L.; Resources, Y.W. and J.T.; Software, Y.B., Y.L. and J.T.; Supervision, Y.Y.; Validation, L.Z. and Y.L.; Visualization, Y.W. and S.L.; Writing—original draft, Y.B., M.S., L.Z., Y.W., S.L., Y.L., Y.Y. and C.L.; Writing—review and editing, J.T., Y.Y. and 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 data presented in this study are available on request from the corresponding author.

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

References

1. Nalendra, A. Rapid Application Development (RAD) model method for creating an agricultural irrigation system based on internet of things. In *Proceedings of the IOP Conference Series: Materials Science and Engineering, Sanya, China, 12–14 November 2021*; IOP Publishing: Bristol, UK, 2021; Volume 1098, p. 022103.
2. Chun, J.; Lee, J.; Kim, J.; Lee, S. An international systematic review of cyberbullying measurements. *Comput. Hum. Behav.* **2020**, *113*, 106485. [CrossRef]
3. Wu, J.; Zhang, C.; Liu, Z.; Zhang, E.; Wilson, S.; Zhang, C. Graphbert: Bridging graph and text for malicious behavior detection on social media. In Proceedings of the 2022 IEEE International Conference on Data Mining (ICDM), Orlando, FL, USA, 28 November–1 December 2022; IEEE: New York, NY, USA, 2022; pp. 548–557.
4. 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]
5. Liu, Q.; Hagenmeyer, V.; Keller, H.B. A review of rule learning-based intrusion detection systems and their prospects in smart grids. *IEEE Access* **2021**, *9*, 57542–57564. [CrossRef]
6. Rezaimehr, F.; Dadkhah, C. A survey of attack detection approaches in collaborative filtering recommender systems. *Artif. Intell. Rev.* **2021**, *54*, 2011–2066. [CrossRef]
7. Alraizza, A.; Algarni, A. Ransomware detection using machine learning: A survey. *Big Data Cogn. Comput.* **2023**, *7*, 143. [CrossRef]
8. Alshehri, A.; Khan, N.; Alowayr, A.; Alghamdi, M.Y. Cyberattack Detection Framework Using Machine Learning and User Behavior Analytics. *Comput. Syst. Sci. Eng.* **2023**, *44*, 1679–1689. [CrossRef]
9. Elnakib, O.; Shaaban, E.; Mahmoud, M.; Emara, K. EIDM: Deep learning model for IoT intrusion detection systems. *J. Supercomput.* **2023**, *79*, 13241–13261. [CrossRef]
10. Meddeb, R.; Jemili, F.; Triki, B.; Korbaa, O. A deep learning-based intrusion detection approach for mobile Ad-hoc network. *Soft Comput.* **2023**, *27*, 9425–9439. [CrossRef]
11. 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]
12. Nicholls, J.; Kuppa, A.; Le-Khac, N.A. Financial cybercrime: A comprehensive survey of deep learning approaches to tackle the evolving financial crime landscape. *IEEE Access* **2021**, *9*, 163965–163986. [CrossRef]
13. Chu, J.; Sha, Z.; Backes, M.; Zhang, Y. Conversation Reconstruction Attack Against GPT Models. *arXiv* **2024**, arXiv:2402.02987.
14. Hu, Y.; Zou, F.; Han, J.; Sun, X.; Wang, Y. Llm-Tikg: Threat Intelligence Knowledge Graph Construction Utilizing Large Language Model. *arXiv* **2023**, arXiv:2308.13916; *Available at SSRN 4671345.*
15. Xu, J.; Stokes, J.W.; McDonald, G.; Bai, X.; Marshall, D.; Wang, S.; Swaminathan, A.; Li, Z. AutoAttacker: A Large Language Model Guided System to Implement Automatic Cyber-attacks. *arXiv* **2024**, arXiv:2403.01038.
16. Yang, Y.; Tu, S.; Ali, R.H.; Alasmary, H.; Waqas, M.; Amjad, M.N. Intrusion detection based on bidirectional long short-term memory with attention mechanism. *Comput. Mater. Contin.* **2023**, *74*, 801–815. [CrossRef]
17. 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. [CrossRef]

18. Wang, Y.; Ma, W.; Xu, H.; Liu, Y.; Yin, P. A lightweight multi-view learning approach for phishing attack detection using transformer with mixture of experts. *Appl. Sci.* **2023**, *13*, 7429. [CrossRef]
19. Chang, Y.; Wang, X.; Wang, J.; Wu, Y.; Yang, L.; Zhu, K.; Chen, H.; Yi, X.; Wang, C.; Wang, Y.; et al. A survey on evaluation of large language models. *Acm Trans. Intell. Syst. Technol.* **2023**, *15*, 1–45.
20. Meng, Y.; Zhang, Y.; Huang, J.; Xiong, C.; Ji, H.; Zhang, C.; Han, J. Text classification using label names only: A language model self-training approach. *arXiv* **2020**, arXiv:2010.07245.
21. Kasneci, E.; Seßler, K.; Küchemann, S.; Bannert, M.; Dementieva, D.; Fischer, F.; Gasser, U.; Groh, G.; Günnemann, S.; Hüllermeier, E.; et al. ChatGPT for good? On opportunities and challenges of large language models for education. *Learn. Individ. Differ.* **2023**, *103*, 102274. [CrossRef]
22. Min, B.; Ross, H.; Sulem, E.; Veyseh, A.P.B.; Nguyen, T.H.; Sainz, O.; Agirre, E.; Heintz, I.; Roth, D. Recent advances in natural language processing via large pre-trained language models: A survey. *Acm Comput. Surv.* **2023**, *56*, 1–40. [CrossRef]
23. Han, K.; Xiao, A.; Wu, E.; Guo, J.; Xu, C.; Wang, Y. Transformer in transformer. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 15908–15919.
24. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc. (NeurIPS): Long Beach, CA, USA, 2017; Volume 30.
25. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
26. Liu, X.; Zheng, Y.; Du, Z.; Ding, M.; Qian, Y.; Yang, Z.; Tang, J. GPT understands, too. *AI Open* 2023, *in press*. [CrossRef]
27. Min, S.; Lewis, M.; Hajishirzi, H.; Zettlemoyer, L. Noisy channel language model prompting for few-shot text classification. *arXiv* **2021**, arXiv:2108.04106.
28. Ebrahimi, M.; Zhang, N.; Hu, J.; Raza, M.T.; Chen, H. Binary black-box evasion attacks against deep learning-based static malware detectors with adversarial byte-level language model. *arXiv* **2020**, arXiv:2012.07994.
29. He, S.; Zhu, J.; He, P.; Lyu, M.R. Loghub: A large collection of system log datasets towards automated log analytics. *arXiv* **2020**, arXiv:2008.06448.
30. Hazell, J. Large language models can be used to effectively scale spear phishing campaigns. *arXiv* **2023**, arXiv:2305.06972.
31. Liu, M.; Li, K.; Chen, T. DeepSQLi: Deep semantic learning for testing SQL injection. In Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis, Virtual Event, USA, 18–22 July 2020; pp. 286–297.
32. Kaur, J.; Garg, U.; Bathla, G. Detection of cross-site scripting (XSS) attacks using machine learning techniques: A review. *Artif. Intell. Rev.* **2023**, *56*, 12725–12769. [CrossRef]
33. Kereopa-Yorke, B. Building resilient SMEs: Harnessing large language models for cyber security in Australia. *J. Ai Robot. Workplace Autom.* **2024**, *3*, 15–27.
34. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of the International Conference on Medical Image Computing and Computer—Assisted Intervention, Munich, Germany, 5–9 October 2015*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
35. Zhang, Y.; Liu, X.; Wa, S.; Liu, Y.; Kang, J.; Lv, C. GenU-Net++: An Automatic Intracranial Brain Tumors Segmentation Algorithm on 3D Image Series with High Performance. *Symmetry* **2021**, *13*, 2395. [CrossRef]
36. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. *Neurocomputing* **2021**, *452*, 48–62. [CrossRef]
37. Fazil, M.; Sah, A.K.; Abulaish, M. Deepsbd: A deep neural network model with attention mechanism for socialbot detection. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4211–4223. [CrossRef]
38. Muthukumar, S.; Ashfauk Ahamed, A. A novel framework of DDoS attack detection in network using hybrid heuristic deep learning approaches with attention mechanism. *J. High Speed Netw.* **2024**, 1–27. [CrossRef]
39. Wang, D.; Zhang, Z.; Jiang, Y.; Mao, Z.; Wang, D.; Lin, H.; Xu, D. DM3Loc: Multi-label mRNA subcellular localization prediction and analysis based on multi-head self-attention mechanism. *Nucleic Acids Res.* **2021**, *49*, e46. [CrossRef] [PubMed]
40. Hu, T.; Xu, C.; Zhang, S.; Tao, S.; Li, L. Cross-site scripting detection with two-channel feature fusion embedded in self-attention mechanism. *Comput. Secur.* **2023**, *124*, 102990. [CrossRef]
41. Wen, P.; He, C.; Xiong, W.; Liu, J. SQL injection detection technology based on BiLSTM-attention. In Proceedings of the 2021 4th International Conference on Robotics, Control and Automation Engineering (RCAE), Wuhan, China, 4–6 November 2021; IEEE: New York, NY, USA, 2021; pp. 165–170.
42. Zhu, E.; Yuan, Q.; Chen, Z.; Li, X.; Fang, X. CCBLA: A lightweight phishing detection model based on CNN, BiLSTM, and attention mechanism. *Cogn. Comput.* **2023**, *15*, 1320–1333. [CrossRef]
43. Chen, J.; Guo, S.; Ma, X.; Li, H.; Guo, J.; Chen, M.; Pan, Z. Slam: A malware detection method based on sliding local attention mechanism. *Secur. Commun. Netw.* **2020**, *2020*, 6724513. [CrossRef]
44. Viggiato, M.; Bezemer, C.P. Leveraging the OPT Large Language Model for Sentiment Analysis of Game Reviews. *IEEE Trans. Games* **2023**, 1–4. [CrossRef]
45. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. Llama: Open and efficient foundation language models. *arXiv* **2023**, arXiv:2302.13971.

46. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 16259–16268.
47. Song, C.W.; Tsai, Y.T. Hyacinth6B: A large language model for Traditional Chinese. *arXiv* **2024**, arXiv:2403.13334.