Deep Learning–Based Network Intrusion Detection Systems

Yaxuan Wang

College of Global Talents, Beijing Institute of Technology, Zhuhai, China

stu22000754@cgt.bitzh.edu.cn

Abstract. With the rapid growth of the internet, the security threats to computer networks have escalated significantly, making the reduction and prevention of cybercrime a top priority in the digital age. Traditional Network Intrusion Detection Systems (NIDS) struggle with limitations in detection accuracy and real-time performance as attackers employ increasingly sophisticated techniques. In recent years, deep learning has emerged as a prominent solution in the NIDS field due to its powerful capabilities in feature extraction and classification. This paper reviews the application of deep learning in NIDS, with a focus on Convolutional Neural Networks (CNN), Long Short-Term Memory Networks (LSTM), and their hybrid models. The paper discusses the strengths of these models in capturing spatial and temporal features and examines their performance on key datasets such as KDD Cup 99 and UNSW-NB15. Additionally, the paper addresses challenges related to computational complexity, real-time performance, and model interpretability, while suggesting future research directions, including model optimization, lightweight architectures, and improved interpretability. Finally, the potential of Automated Machine Learning (AutoML) in advancing NIDS design and enhancing response capabilities is explored. This study offers valuable insights for further research and development in NIDS.

Keywords: Deep Learning, Intrusion Detection, Convolutional Neural Networks, Long Short-Term Memory Networks, Automated Machine Learning.

1. Introduction

Traditional Intrusion Detection Systems (IDS) play a critical role in network security and are generally categorized into signature-based IDS and anomaly-based IDS. However, with the increasing complexity and diversity of network attacks, traditional IDS reveals significant shortcomings in handling complex features and temporal data.

Firstly, traditional IDS mainly rely on predefined rules or signatures to detect intrusions, which prevents them from identifying new attack patterns. Moreover, to improve detection accuracy, manual feature extraction and selection are required, a process that is time-consuming and labor-intensive, often leading to the omission of potentially crucial features. Additionally, modern network traffic contains numerous complex and high-dimensional features, such as the combination of multiple protocols and encrypted traffic. Traditional IDS struggle with high-dimensional data, making it difficult to capture critical intrusion features.

Secondly, many network attacks manifest as a series of time-related behaviors or events, such as gradual infiltration or persistent attacks. However, traditional IDS typically rely on single, static events for detection, neglecting temporal correlations. This static analysis method often fails to detect

© 2024 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

distributed denial-of-service (DDoS) attacks or advanced persistent threats (APT), which involve time-dependent behaviors. For example, multiple suspicious connections from a single IP address may be indicative of an attack, but traditional IDS lack effective mechanisms to process such temporal data, rendering them incapable of detecting attacks with long latency periods.

Real-time performance is another issue. When faced with complex temporal data requiring fast responses, traditional IDS cannot keep up, failing to provide timely protection. This limitation is especially critical when addressing rapidly evolving attacks or zero-day threats.

Consequently, more researchers are exploring how advanced models, such as CNN and LSTM, can be integrated with deep learning technologies to overcome these deficiencies and enhance the overall performance of intrusion detection systems. Formatting the title, authors and affiliations

Please follow these instructions as carefully as possible so all articles within a conference have the same style to the title page. This paragraph follows a section title so it should not be indented.

Paper [1] proposed a deep learning-based intrusion detection system (DL-IDS) that combines CNN and LSTM to extract both spatial and temporal features of network traffic data. The model was tested on the CICIDS2017 dataset, achieving an accuracy of 98.67% and providing robust detection across various attack types, outperforming other machine learning models. Paper [2] proposed a hybrid deep neural network for intrusion detection, combining CNN to extract spatial features and LSTM to capture temporal dependencies in network traffic. Their model was evaluated using the CIC-IDS2017, UNSW-NB15, and WSN-DS datasets, demonstrating improved accuracy and detection rates, while effectively reducing false alarm rates compared to other machine learning and deep learning models.

This paper explores a deep learning-based network intrusion detection system that combines the strengths of CNN and LSTM networks to extract spatiotemporal features from network traffic. First, the paper describes the research background and existing methods in the related field, followed by a detailed introduction to the architecture of the proposed hybrid model and its implementation process. Through empirical analysis, the paper validates the model's effectiveness across multiple datasets, demonstrating significant improvements in detection rate and accuracy, while also effectively reducing the false alarm rate. Overall, the CNN and LSTM-based intrusion detection system show great potential in practical applications by simultaneously processing spatial and temporal features.

2. Overview of Related Technologies

2.1. Convolutional Neural Network (CNN)

Deep learning is an artificial intelligence technology based on neural networks, offering stronger feature learning and data prediction capabilities compared to traditional machine learning methods. Deep learning often employs multi-layered complex neural networks for feature learning. Within the network structure, each layer extracts features from the input data and transforms its dimensions, enabling the identification of data representations in higher dimensions, which facilitates precise classification or prediction of the data. Among the various neural network architectures, CNN is one of the most commonly used structures. The core of Convolutional Neural Networks lies in its convolutional layers and pooling layers. The convolutional layers apply several convolutional kernels to the input data, transforming it into a higher-dimensional representation to identify its features. The pooling layers, through pooling techniques, reduce the spatial dimensions of the feature maps.

As a classical deep learning model, CNN has been widely applied in various fields due to its excellent performance in image processing. In Network Intrusion Detection Systems (NIDS), the main role of CNN is to extract key spatial features from network traffic data. According to the literature [1], CNN is used in deep learning models to extract spatial features from network traffic data, including protocol distribution and statistical features of source and destination IP addresses. By leveraging CNN, researchers can better analyze patterns within network traffic, thereby improving the accuracy of intrusion detection. These features are particularly effective in detecting specific types of attacks, especially those with distinct patterns or structures.

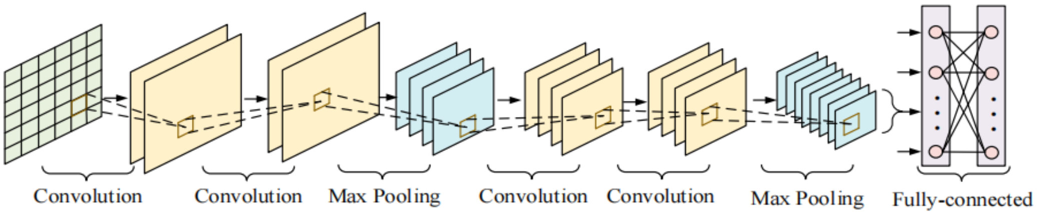

Figure 1. Convolutional Neural Network Architecture [3]

2.2. Long Short-Term Memory Network (LSTM)

LSTM is a deep learning model specifically designed for processing time-series data. Through its unique memory cells and gating mechanisms, LSTM can capture long-term dependencies within input data, which is particularly important in the temporal analysis of network traffic. The core concept of LSTM lies in its ability to store and translate input data over time through its memory cells. These memory cells are processed via gating mechanisms, represented by activation functions. By adjusting the weights and values of the activation functions, LSTM networks effectively generate temporal features between input and output data. In network intrusion detection, LSTM is primarily used to handle gradually evolving attacks, [2] such as Advanced Persistent Threats (APT). These types of attacks often consist of multiple stages, and LSTM can improve detection accuracy by analyzing time-series data to identify the correlations between these stages.

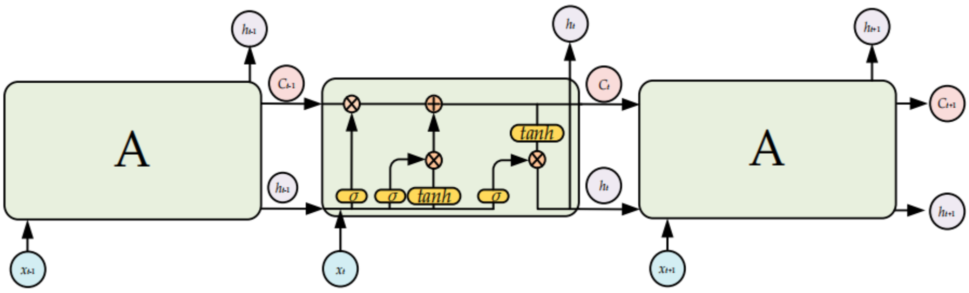

Figure 2. Long Short-Term Memory Architecture [3]

2.3. Advantages of the CNN-LSTM Combined Model

Table 1. Performance comparison of CNN-LSTM and traditional Deep Learning Models on KDD Cup 99 datasets

Datasets	Model	Accuracy(%)	Detection Rate (DR, %)	False Positive Rate (FPR, %)
KDD Cup 99	CNN-LSTM	99.70	99.60	/
KDD Cup 99	LSTM-RNN	96.93	98.88	10.04
KDD Cup 99	GA-ELM	98.90	99.16	1.36
KDD Cup 99	AE-CNN	93.99	77.94	6.82

a Detection Rate (DR) represents the percentage of true positives among all actual positives.
b False Positive Rate (FPR) represents the percentage of false positives among all actual negatives.

A glance at the graph provided reveals a comparison of the accuracy, detection rate (DR), and false positive rate (FPR) across different models.[4] It is evident that the proposed CNN-LSTM model demonstrates a significant advantage over traditional deep learning models on the KDD Cup 99 Datasets, achieving an accuracy of 99.70% and a detection rate of 99.60%, with a notably low false positive rate

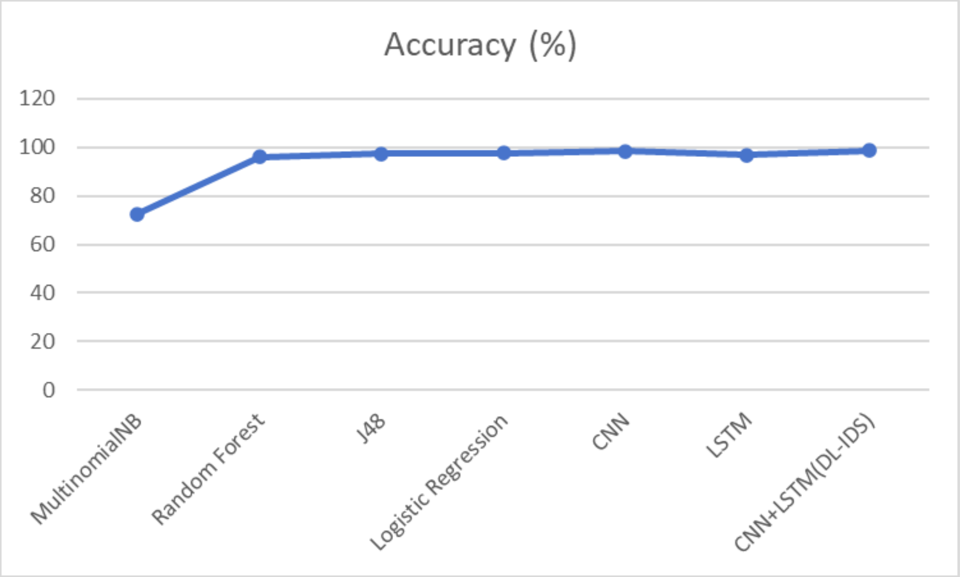

Figure 3. Comparison of model accuracy for intrusion detection systems on CICIDS2017 datasets

The accuracy of the MultinomialNB model is the lowest (approximately 72%), indicating that it performs poorly in handling network intrusion detection problems. The accuracy of the Random Forest, J48, and Logistic Regression models is roughly similar, ranging between 96% and 98%, suggesting that these traditional machine learning methods are somewhat effective in handling intrusion detection, but still fall short compared to deep learning models. The CNN and LSTM models outperform traditional machine learning models, with accuracies between 96% and 98.5%. This demonstrates their superior ability to capture complex network traffic features. The CNN-LSTM (DL-IDS) model achieves the highest accuracy, close to 99%, indicating that the combination of CNN and LSTM provides the best detection performance by more comprehensively extracting spatial and temporal features, further improving detection accuracy. The CNN-LSTM model (DL-IDS) demonstrates the highest accuracy in detecting network intrusions, proving the advantage of this combined model in capturing complex patterns and temporal sequences, making it more reliable for practical network security applications.[5]

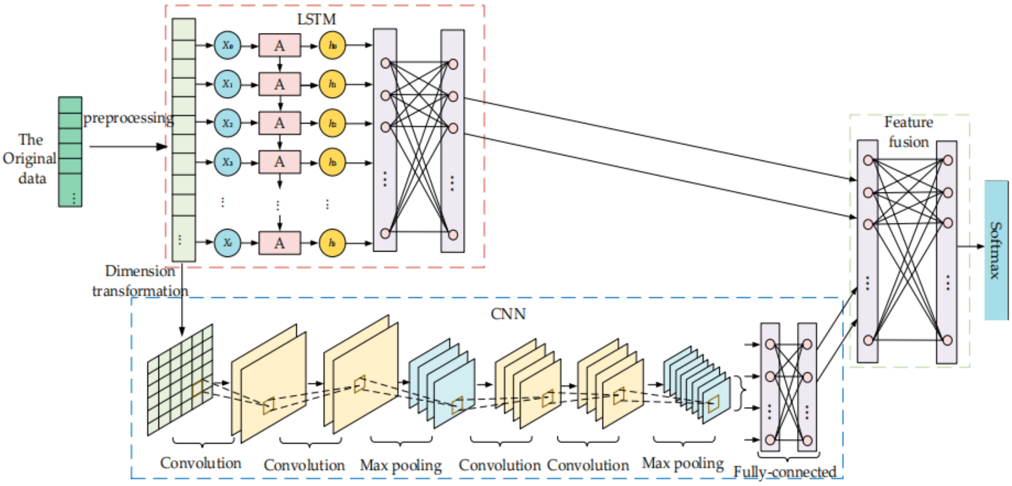

Figure 4. Cross-Layer Feature Fusion CNN-LSTM Intrusion Detection Model [3]

The success of this combined model lies in its ability to capture the multidimensional information in network traffic. CNN can identify complex patterns within data packets, while LSTM is able to track the evolution of these patterns over time. By combining these two capabilities, the model can more accurately detect abnormal behavior in the network and respond to potential threats in a timely manner.

3. Problems and Solutions

3.1. Computational Complexity

The computational complexity of the combined CNN and LSTM model is primarily reflected in the multi-layer convolution operations and the analysis of time series. CNN processes large amounts of high-dimensional data, while LSTM captures dependencies between sequences through complex time-step processing. Although this combination improves the model's detection accuracy, it significantly increases the demand for computational resources, impacting the real-time performance of network intrusion detection systems. To reduce computational complexity while maintaining high detection accuracy, the following optimization methods can be considered:

3.1.1. Model Compression. Model compression techniques can effectively reduce the computational requirements and storage space of deep learning models. Common model compression methods include:

- **Pruning**: Pruning techniques reduce the size of the model by removing unimportant connections or neurons. For the CNN-LSTM model, convolutional kernels and LSTM units that do not significantly impact performance can be pruned, reducing computational load and storage requirements. [6] Studies show that pruned models can greatly decrease computational demands while maintaining performance.
- **Quantization**: Quantization reduces the precision of model weights and activation values from high precision (e.g., 32-bit floating-point) to lower precision (e.g., 8-bit integers), thereby lowering computational requirements and memory usage. Quantized models not only achieve faster inference speeds but also require fewer computational resources, making them suitable for embedded systems or edge devices.
- **Knowledge Distillation**: Knowledge distillation involves training a smaller "student" model to mimic the behavior of a larger "teacher" model, reducing computational complexity while maintaining model performance. The CNN-LSTM combined model can be distilled into a more lightweight version, making it more suitable for real-time detection tasks.

3.1.2. Parallel Computing. Parallel computing techniques can accelerate the training and inference processes of deep learning models by distributing computational tasks across multiple processing units simultaneously. For the CNN-LSTM combined model, the following parallel computing strategies can significantly enhance real-time performance:

- **Data Parallelism:** Data parallelism is one of the most commonly used parallel computing methods, and it is suitable for the CNN-LSTM combined model. Data parallelism works by splitting the input data into multiple subsets, processing them in parallel on multiple processing units, and then aggregating the results. This approach effectively reduces the computational load on each processing unit and accelerates the overall model's processing speed.
- **Model Parallelism**: Model parallelism is applicable when the model is too large to fit on a single device. For the CNN-LSTM combined model, the CNN and LSTM components can be assigned to different computing units and run in parallel, reducing the burden on individual processing units and increasing the overall inference speed.
- **Distributed Computing**: In a distributed computing environment, the computational tasks of the model are executed in parallel across multiple servers or computing nodes. This method is suitable for real-time analysis of large-scale network traffic, significantly improving the processing speed of the CNN-LSTM combined model and reducing latency.

By leveraging model compression and parallel computing techniques, the computational complexity of the CNN-LSTM combined model can be effectively mitigated, thus improving the real-time performance of NIDS. These optimization methods not only reduce the model's computational resource requirements while maintaining high detection accuracy but also meet the strict real-time demands of practical applications. These improvements are crucial for promoting the widespread use of CNN-LSTM models in network intrusion detection.

3.2. Data Dependency

In the development of NIDS, the CNN-LSTM hybrid model heavily relies on large-scale, high-quality datasets that need to cover diverse attack types and sufficient normal traffic. However, obtaining such datasets is challenging due to issues like data imbalance, privacy concerns, and resource limitations. To effectively address the issue of data dependency, researchers have actively explored and implemented several strategies, with data augmentation and transfer learning standing out as prominent approaches.

3.2.1. Strategy 1: Data Augmentation. Data augmentation techniques increase the diversity and quantity of training data by applying various transformations to existing data, generating new samples. Common data augmentation methods include:

- **Data Noise Injection**: By adding noise to network traffic (e.g., altering packet sequences, introducing random delays), this simulates the random disturbances in real networks. This helps models learn more robust features and reduces overfitting to specific patterns.
- **Data Slicing and Stitching**: Existing data is sliced into smaller fragments and then randomly stitched together to generate new traffic sequences. This method increases data diversity, especially when dealing with temporal data, effectively enhancing the model's ability to learn time dependencies.
- **GANs (Generative Adversarial Networks)**: GANs generate new network traffic samples, including both normal and attack traffic. GANs can produce high-quality, diverse samples, alleviating the issue of data scarcity. Studies have shown that using GAN-generated samples can significantly improve model performance in scenarios with limited data. [7]

3.2.2. Strategy 2: Transfer Learning. Transfer learning reduces the dependence on large amounts of labeled data by transferring knowledge learned by a pre-trained model on large-scale datasets to the target task. For network intrusion detection, common transfer learning methods include:

- **Feature Transfer**: Applying feature extractors learned from general datasets (such as ImageNet) to network traffic data and fine-tuning them with network-specific features. This method can greatly reduce training time and improve the model's performance on smaller datasets.
- **Domain Adaptation:** Transfer learning is applied between different network environments or traffic patterns by adapting the distribution differences between domains. This allows the model to maintain efficient detection capabilities in new environments. [5] demonstrates how domain adaptation can be used to transfer a model from one network environment to another, thereby improving the model’s generalization.

3.3. Model Design Complexity

Using deep learning models, particularly hybrid models that combine CNN and LSTM, can indeed improve detection accuracy and response speed. However, the architecture design of such combined models is often highly complex, requiring consideration of various factors such as the number of layers, the number of neurons per layer, and the choice of activation functions. This complexity not only increases the difficulty of model design but can also limit the model's generalization ability, as different datasets and application scenarios often require different model architectures. To address this complexity, Automated Model Design (AutoML) has emerged as a viable solution.

3.3.1. Complexity of Hybrid Model Architecture. The CNN-LSTM hybrid model aims to fully leverage the strengths of Convolutional Neural Networks (CNN) in feature extraction and Long Short-Term Memory (LSTM) networks in sequence modeling. However, this innovative design introduces significant architectural complexity. Precisely configuring the number of convolutional layers in CNN, the size of the convolutional kernels, and the number of units in LSTM layers requires close alignment with the specific dataset characteristics and task requirements. Any improper configuration can lead to overfitting or underfitting, thus affecting the stability and accuracy of detection performance. Moreover, hyperparameter tuning during the model training process, such as learning rate, regularization coefficient, and batch size, can have a profound impact on the final training outcome. Fine-tuning these parameters often requires extensive experimentation and time investment. Additionally, given the diversity of network attack types and the complex, ever-changing data scenarios, targeted model optimization strategies become crucial. For example, high-dimensional network traffic data may require deeper CNN structures to enhance feature capture, while long sequence attack patterns might necessitate more complex LSTM layers to improve sequence modeling and prediction capabilities.

3.3.2. Feasibility of AutoML. AutoML aims to optimize model architecture and parameters through automation, reducing the time and effort involved in manual tuning. To improve the performance of Network Intrusion Detection Systems (NIDS), researchers have proposed the feasibility of using AutoML for model optimization. [8] noted that using semi-dynamic hyperparameter optimization methods can significantly enhance the performance of various models, including the automatic selection of optimization algorithms, learning rates, and batch sizes. These automated techniques have led to significant improvements in model accuracy on the UNSW-NB15 dataset. In NIDS model design, AutoML provides the following feasibility and advantages:

- **NAS (Neural Architecture Search):** NAS in AutoML can automatically search for the optimal neural network architecture. By evaluating the performance of numerous potential network structures (e.g., accuracy, recall), NAS identifies the best model architecture. For the CNN-LSTM hybrid model, NAS can automatically explore the optimal configuration of layers to suit different datasets and tasks.
- **Hyperparameter Optimization:** AutoML can automatically tune a model's hyperparameters, using techniques like random search and Bayesian optimization to find the best parameter combinations, thus improving training efficiency and detection performance. This process eliminates the tedious task of manual tuning and may even uncover the best parameter configurations that human designers could easily overlook.
- **Rapid Iteration and Deployment:** AutoML tools typically support fast model iteration and deployment. For practical NIDS applications, AutoML can quickly generate and test multiple models, swiftly selecting the most suitable architecture for the current environment and data. Furthermore, as new data becomes available, AutoML can automatically adjust the model to maintain optimal performance.

3.4. Case Analysis

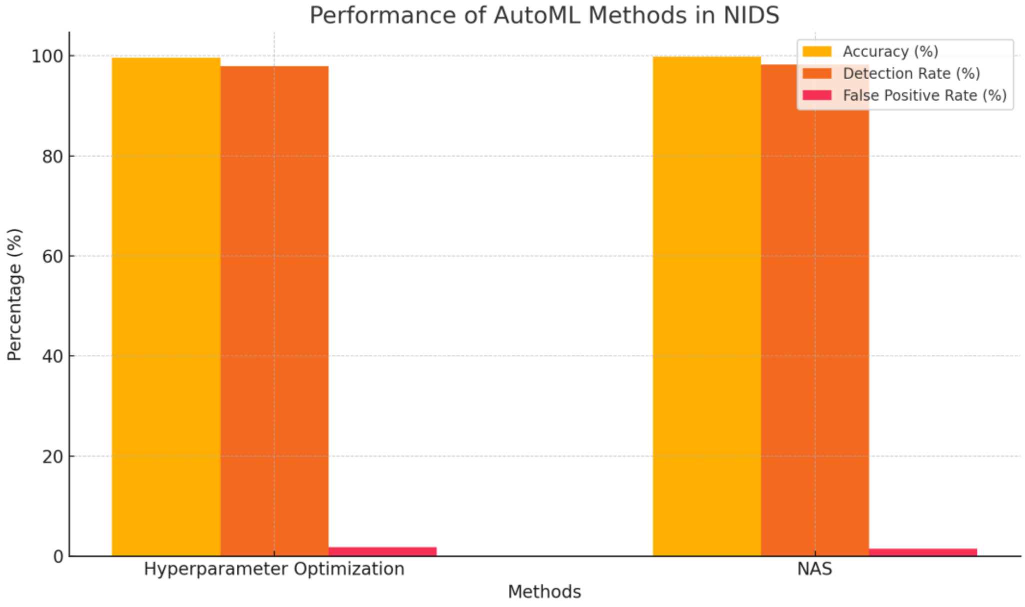

Figure 5. Performance of AutoML methods in NIDS

The researchers optimized the CNN-LSTM model by adjusting the learning rate, batch size, and regularization parameters. The study demonstrated that the optimized model performed exceptionally well on the CIC-IDS2017 and UNSW-NB15 datasets, with accuracy improving to 99.64% and 94.53%, respectively, while also reducing training time.

AutoML has demonstrated great potential in the design and optimization of NIDS models. By leveraging NAS technology and automated hyperparameter optimization, researchers can generate superior model architectures, significantly reduce training time, and enhance the adaptability and performance of models across different datasets and scenarios. As AutoML technology continues to evolve, the design of NIDS models will become more efficient and intelligent.

4. Proposed Future Research Directions

4.1. Innovative Network Architecture Design

First, the attention mechanism, which has demonstrated outstanding performance in tasks such as natural language processing and image recognition, can be applied to NIDS in the future. This would allow the model to automatically focus on key features in network traffic, particularly for extracting features related to abnormal patterns or specific attacks, thereby improving the model's accuracy and recall. Second, residual connections can facilitate the training of deeper neural networks by addressing the vanishing gradient problem, enhancing the NIDS's ability to handle high-dimensional and complex data. Additionally, multi-task learning enables the model to simultaneously detect various types of attacks and identify normal traffic. By sharing representations between tasks, this approach improves the model's generalization ability and reduces the risk of overfitting. Finally, hybrid architectures combining CNN, LSTM, and other deep learning models (such as GANs and VAEs) can strengthen the NIDS’s ability to handle diverse network attacks, improving detection accuracy and robustness. These innovative architectures offer new approaches for the development of NIDS, addressing the ever-evolving demands of network security [9].

4.2. Real-Time Optimization

To improve the real-time performance of NIDS, model lightweighting is an important direction. Quantization techniques [10], which convert floating-point parameters into low-precision integers, significantly reduce computational complexity, making it suitable for deployment on resource-limited devices. Model pruning further reduces the model size by removing redundant neurons and connections, enhancing computational efficiency. By combining these two methods, NIDS can maintain high detection accuracy while improving response speed when processing high-frequency data streams. The application of efficient algorithms, such as knowledge distillation, allows a smaller "student" model to learn from a larger "teacher" model’s behavior, greatly reducing the model size and computational demands while preserving detection accuracy. Additionally, parallel computing techniques decompose complex computational tasks and process them simultaneously [11], drastically shortening model inference time, making it particularly useful for handling large-scale network traffic [12].

5. Conclusion

This review thoroughly explores the widespread application and recent advancements of deep learning technology in NIDS, with a particular emphasis on the superior performance of CNN and LSTM in extracting complex features from network traffic. By analyzing the experimental results of hybrid models such as CNN-LSTM, this paper highlights the significant role of deep learning in improving NIDS detection accuracy and real-time responsiveness, as well as demonstrating its vast potential in defending against modern network attacks.

Looking ahead, NIDS research will focus on several cutting-edge areas. First, innovative network architecture design will further enhance model detection capabilities, such as integrating attention mechanisms to strengthen the capture of key features or utilizing residual connections to mitigate the vanishing gradient problem in deep networks, thereby improving generalization. Second, continued research into model lightweight and efficient algorithms will provide more efficient and real-time network intrusion detection in resource-constrained environments, ensuring stable NIDS operation in various complex scenarios.

At the same time, the development of Automated Machine Learning (AutoML) will drive the intelligent progression of NIDS. AutoML not only simplifies the model design and optimization process but, when combined with reinforcement learning, could enable NIDS to adapt dynamically to changing network environments, enhancing its ability to respond to emerging network threats. As technology continues to evolve and innovate, deep learning-driven NIDS will play an increasingly intelligent and automated role in network security, providing a solid foundation for building a secure and reliable cyberspace.

References

[1] Sun, P., Liu, P., Li, Q., Liu, C., Lu, X., Hao, R., & Chen, J. (2020). DL-IDS: Extracting Features Using CNN-LSTM Hybrid Network for Intrusion Detection System. Security and communication networks, 2020(1), 8890306.

[2] Halbouni, A., Gunawan, T. S., Habaebi, M. H., Halbouni, M., Kartiwi, M., & Ahmad, R. (2022). CNN-LSTM: hybrid deep neural network for network intrusion detection system. IEEE Access, 10, 99837-99849.

[3] Yao, R., Wang, N., Liu, Z., Chen, P., & Sheng, X. (2021). Intrusion detection system in the advanced metering infrastructure: a cross-layer feature-fusion CNN-LSTM-based approach. Sensors, 21(2), 626.

[4] Shao, R. R., Liu, Y., Zhang, W., & Wang, J. (2022). A Review of Knowledge Distillation Research in Deep Learning. Journal of Computer Science, 45(8), 1638-1673.

[5] Song, S., Zhang, Y., Zhang, L., Cen, Y. G., & Li, H. D. (2022). Lightweight Object Detection Algorithm Based on Deep Learning. Systems Engineering & Electronics, 44(9).

[6] Gao, H., Tian, Y. L., Xu, F. Y., & Zhong, S. (2020). A Review of Deep Learning Model Compression and Acceleration. Journal of Software, 32(1), 68-92.

[7] Wang, Y. T., Zhou, H. Q., Yan, J. X., He, C., & Huang, L. L. (2021). Research Progress in Computational Optics Based on Deep Learning Algorithms. Chinese Journal of Lasers, 48(19), 1918004.

[8] Wu, S. Q., & Li, X. M. (2020). A Review of Research Progress on Generative Adversarial Networks. Computer Science and Exploration, 14(3), 377.

[9] Jastrzębski, S., Arpit, D., Ballas, N., Verma, V., Che, T., & Bengio, Y. (2017). Residual connections encourage iterative inference. arXiv preprint arXiv:1710.04773.

[10] Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of deep learning. Neurocomputing, 452, 48-62.

[11] Liu, J. W., Liu, J. W., & Luo, X. L. (2021). Research progress in attention mechanism in deep learning. Chinese Journal of Engineering, 43(11), 1499-1511.

[12] Crawshaw, M. (2020). Multi-task learning with deep neural networks: A survey. arXiv preprint arXiv:2009.09796.