JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

DEEP HYBRID SUPERVISED MODEL FOR REAL-TIME CYBERATTACKS DETECTION USING WINDOWS EVENT LOGS

[1]Ms. Rubina Kureshi, [2]Dr. R. K. Dhuware
[1]Research Scholar, Department of Electronics and Computer Science, RTM Nagpur University, Nagpur
[2]Head, Department of Computer Science, Dhote Bandhu Science College, Gondia

Abstract : Windows Event Logs provide critical system-level telemetry for monitoring enterprise security operations. However, their high dimensionality, heterogeneous feature composition, and temporal dependencies present significant challenges for accurate cyberattack detection. This paper proposes a deep hybrid supervised learning framework integrating a one-dimensional Convolutional Neural Network (CNN) with stacked Long Short-Term Memory (LSTM) layers for real-time multi-class cyberattack detection using Windows Event Logs. The proposed architecture incorporates structured preprocessing, categorical encoding, feature normalization, Synthetic Minority Oversampling Technique (SMOTE), and class-weight optimization to address data imbalance. Experimental evaluation demonstrates strong classification performance across multiple event categories, with low misclassification rates and balanced precision-recall behavior. The results confirm that the hybrid CNN–LSTM model effectively captures both structural feature correlations and sequential attack patterns, making it suitable for deployment within enterprise Security Information and Event Management (SIEM) systems.

Keywords **-** Windows Event Logs, Intrusion Detection System, Supervised Learning, CNN-LSTM, SMOTE, Cybersecurity Analytics, Deep Learning.

I. INTRODUCTION

Enterprise systems continuously generate Windows Event Logs that record authentication attempts, process execution traces, system warnings, and security alerts. These logs serve as a critical source for identifying malicious activities, including privilege escalation, brute-force attacks, and lateral movement within enterprise networks.Traditional rule-based and signature-based intrusion detection systems (IDS) often lack adaptability against evolving and previously unseen threats. Recent studies demonstrate that deep learning models significantly enhance detection performance by automatically learning complex feature representations from raw or structured security data [1], [2]. In particular, hybrid deep learning architectures combining convolutional neural networks (CNN) and long short-term memory (LSTM) networks have shown superior capability in jointly modeling spatial feature correlations and temporal dependencies [1], [3], [7].However, most existing research primarily focuses on network traffic datasets or IoT environments, with limited investigation into structured Windows Event Log–based supervised intrusion detection. Furthermore, class imbalance—an inherent characteristic of enterprise security logs—remains insufficiently addressed in many existing implementations.To bridge this gap, this study proposes a supervised hybrid CNN–LSTM framework specifically tailored for Windows Event Log–based cyberattack detection. The

proposed model incorporates Synthetic Minority Over-sampling Technique (SMOTE) and class-weight balancing to mitigate skewed class distributions and improve minority attack detection performance.

This work makes the following contributions:

1. **Proposes a supervised hybrid CNN–LSTM framework** for structured Windows Event Log classification in enterprise security environments.
2. **Integrates SMOTE-based oversampling and class-weight balancing** to effectively mitigate severe class imbalance in SYS log datasets.
3. **Performs comprehensive performance evaluation** using confusion matrix analysis and standard classification metrics, including precision, recall, F1-score, and ROC-AUC.
4. **Analyzes real-time deployment feasibility** of the proposed framework within enterprise Security Operations Center (SOC) environments.

II. RELATED WORK

Deep learning has emerged as a dominant paradigm in intrusion detection research due to its capacity to automatically learn hierarchical feature representations from high-dimensional security data. Comprehensive reviews such as [2] report that convolutional neural networks (CNN), long short-term memory (LSTM) networks, and hybrid deep architectures consistently outperform traditional machine learning approaches in complex multi-class intrusion detection tasks. Despite these advances, challenges remain in handling dataset imbalance, ensuring cross-environment generalization, and evaluating models on real-world enterprise log data.Hybrid CNN–LSTM architectures have gained significant attention for modeling both spatial and temporal characteristics of attack patterns. An attention-enhanced CNN–LSTM framework presented in [1] integrates convolutional feature extraction with sequential modeling and attention mechanisms, demonstrating improved detection accuracy and reduced false alarm rates. Similarly, the integration of Transformer-based self-attention with CNN–BiLSTM architectures has been shown to enhance spatiotemporal dependency learning in IoT intrusion detection scenarios [7]. However, these studies primarily evaluate performance on network flow–based datasets, such as CICIDS and IoT traffic traces, limiting their applicability to structured system log environments.

Multi-class intrusion detection using hybrid deep learning models has also been explored extensively. A CNN–BiLSTM-based IDS introduced in [3] demonstrates robustness in detecting diverse attack categories under imbalanced conditions. While such hybrid architectures effectively capture local and sequential dependencies, their validation is generally confined to packet-level or flow-level datasets. Consequently, the adaptability of these approaches to structured Windows Event Logs and SYS log datasets remains insufficiently investigated.Log-based anomaly detection represents another important direction in cybersecurity research. A notable LSTM-based framework for log sequence modeling is presented in [4], where sequential system event patterns are learned for anomaly detection and failure diagnosis. Although this approach demonstrates the effectiveness of semantic sequence modeling, it primarily focuses on unsupervised anomaly detection rather than supervised multi-class attack prediction. Furthermore, many log-based methods rely on unstructured log parsing, which may introduce preprocessing inconsistencies and reduce scalability in structured enterprise logging systems.Despite substantial progress in deep learning–

based IDS research, three major limitations persist. First, limited studies systematically evaluate supervised hybrid CNN–LSTM architectures on structured Windows Event Logs. Second, class imbalance mitigation—critical in enterprise security datasets—is often insufficiently integrated into model development and benchmarking. Third, comparative analyses across multiple deep learning architectures under consistent preprocessing and evaluation conditions are rarely conducted for SYS log datasets.

To address these limitations, this work presents a structured comparative analysis of deep learning models for security attack prediction using Windows Event Logs. Unlike prior studies that predominantly focus on network traffic data, the proposed framework applies systematic preprocessing, feature structuring, and class balancing techniques tailored to structured SYS log data. Moreover, multiple deep learning architectures are evaluated under identical experimental conditions to ensure fair and rigorous benchmarking. This approach contributes to bridging the gap between hybrid deep learning research and practical enterprise log-based intrusion detection systems.

Table 1. Summary of Representative Deep Learning-Based IDS Approaches

Ref.	Architecture	Dataset Type	Key Contribution	Identified Limitation
[1]	Attention-CNN-LSTM	IoT traffic data	Joint spatial–temporal modeling with attention mechanism	Not evaluated on structured Windows Event Logs
[2]	Survey of DL-based IDS	Multiple benchmark datasets	Comprehensive taxonomy of CNN, LSTM, and hybrid models	No experimental validation on SYS log datasets
[3]	CNN–BiLSTM	Network traffic (multi-class)	Robust hybrid deep learning architecture	Focus limited to packet/flow-level data
[4]	LSTM (DeepLog)	System logs	Sequential log anomaly detection	Primarily unsupervised; not multi-class supervised classification
[5]	GAN-based IDS (S2CGAN-IDS)	Imbalanced IoT datasets	Addresses minority class imbalance	Increased model complexity and IoT-specific evaluation
[7]	Transformer + CNN–BiLSTM	IoT intrusion detection	Enhanced long-range dependency learning	Dataset limited to IoT environments
[8]	Deep Neural Network	Network intrusion dataset	Intelligent intrusion classification	Limited log-based validation
[9]	Autoencoder-based DL	Network intrusion data	Unsupervised feature learning	Focused on network traffic only
[10]	Survey (Emerging DL IDS)	Multiple domains	Analysis of IDS challenges and trends	Lacks structured Windows log experimentation
Proposed Work	Structured CNN–LSTM (Supervised)	**Windows Event Logs (SYS Logs)**	Structured preprocessing + class balancing + fair comparative evaluation	Designed specifically for enterprise log environments

Table 1 summarizes representative deep learning-based intrusion detection approaches and highlights existing research limitations. As observed, most studies focus on network traffic datasets, whereas structured Windows Event Log-based supervised comparative evaluations remain limited.

III. DATASET AND PREPROCESSING

A. Dataset Description

The dataset used in this study consists of structured Windows Event Logs exported in CSV format. Each log entry contains several key attributes, including *Event ID*, *Timestamp*, *Source*, *System Metadata*, and *Log Level*, where the Log Level serves as the target variable for classification.The target variable represents multi-class event severity levels, enabling the development and evaluation of a multi-class classification model. The collected Windows Event Log dataset comprises 58,332 records, including instances with blank fields and missing values, which were addressed during the data preprocessing phase to ensure data quality and model robustness.

B. Data Preprocessing

To ensure data quality, consistency, and suitability for deep learning-based classification, several preprocessing steps were systematically applied to the Windows Event Log dataset.As shown in Figure 1 and Figure 2 showing dataset after preprocessing consisting clean 23,629 records in csv file.

1. **Timestamp Conversion**

   The *Timestamp* attribute, originally recorded in datetime format, was converted into Unix timestamp representation. This transformation enabled efficient numerical processing and ensured compatibility with machine learning algorithms that require numeric input features.

2. **Data Cleaning**

   Data cleaning procedures were performed to enhance dataset integrity. Records containing missing, null, or corrupted values were carefully examined and removed where appropriate. Furthermore, the *Event ID* attribute was explicitly converted into integer format to maintain type consistency and facilitate numerical analysis.

3. **Categorical Encoding**

   Categorical attributes, including system-related metadata and source identifiers, were transformed into numerical representations using label encoding. This step preserved categorical distinctions while ensuring compatibility with deep learning models.

4. **Feature Normalization**

   To prevent scale dominance among features and to improve convergence during model training, Min–Max normalization was applied. The transformation scales each feature to the range [0,1][0,1][0,1] according to:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad \ldots\ldots (1)$$

   where X represents the original feature value, and Xmin and Xmax denote the minimum and maximum values of the feature, respectively.

5. **Class Imbalance Handling**

   Class imbalance poses a significant challenge in intrusion detection systems (IDS), as it can bias the model toward majority classes and degrade minority-class detection performance [2], [3]. To

mitigate this issue, rare classes with extremely low representation were removed to reduce noise and instability. Subsequently, the Synthetic Minority Oversampling Technique (SMOTE) with k=2k = 2k=2 nearest neighbors was employed to generate synthetic samples for minority classes. Additionally, class weights were incorporated during model training to further balance the learning process and enhance classification robustness.

6. Dataset Splitting

To ensure fair model evaluation and prevent data leakage, stratified sampling was applied to preserve class distribution across subsets. The dataset was divided as 70% for training,15% for validation and 15% for testing

Figure 1. Data Preprocessing Pipeline for Structured Windows Event Logs

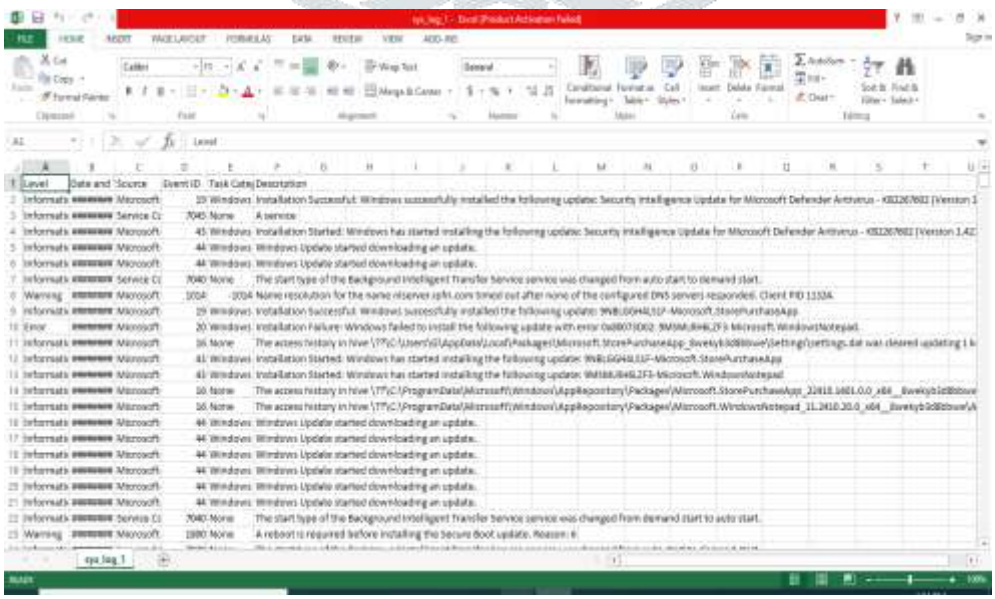

Fig:2 Dataset after preprocessing

IV. PROPOSED CNN–LSTM ARCHITECTURE

A. Model Architecture

To effectively learn discriminative patterns from structured Windows Event Log sequences, a hybrid deep learning architecture integrating convolutional and recurrent layers was designed. The combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks enables the model to capture both localized feature patterns and long-range temporal dependencies present in sequential log data as shown in Figure 3.

The architecture is structured as follows:

1. **One-Dimensional Convolutional Layer (Conv1D):**

The initial layer consists of 64 filters with a kernel size of 3 and employs the Rectified Linear Unit (ReLU) activation function. This layer extracts meaningful local patterns from sequential input data by performing sliding window convolution operations.

2. **First LSTM Layer (50 units, return sequences = True):**

The extracted feature maps are passed to an LSTM layer configured to return full sequences. This allows the model to retain temporal information at each time step for deeper sequential analysis.

3. **Dropout Layer (rate = 0.2):**

A dropout regularization technique is applied to reduce overfitting by randomly deactivating 20% of neurons during training.

4. **Second LSTM Layer (50 units):**

A subsequent LSTM layer further processes the sequential representation and outputs a condensed temporal feature vector summarizing the learned dependencies.

5. **Fully Connected (Dense) Layer (50 units, ReLU):**

The dense layer transforms the learned features into a higher-level representation suitable for classification.

6. **Dropout Layer (rate = 0.2):**

An additional dropout layer enhances model generalization.

7. **Softmax Output Layer:**

The final layer produces probability scores across multiple event severity classes using the Softmax activation function.This layered configuration ensures comprehensive feature learning by combining spatial feature extraction with temporal sequence modeling.

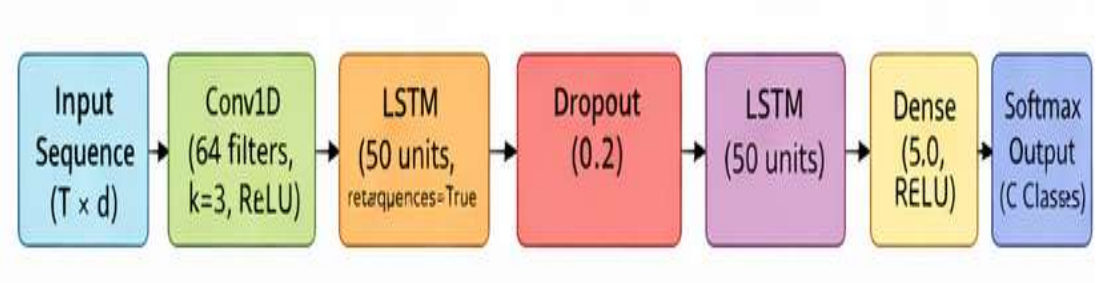

Fig 3:Proposed Hybrid CNN-LSTM Architecture for Windows Event Log Classification

B. Mathematical Representation

Let the input log sequence be defined as:

$$X = \{x_1, x_2, \ldots, x_T\} \qquad (2)$$

where x_t denotes the feature vector at time step t, and T represents the sequence length.

1. Convolutional Feature Extraction

The convolutional operation transforms the input sequence into feature maps:

$$F = Conv1D(X) \qquad (3)$$

where F represents the extracted local feature representations derived from sliding convolutional filters.

2. Sequential Modeling with LSTM

The temporal dependencies within the feature maps are modeled using LSTM units:

$$H_t = LSTM(F_t) \qquad (4)$$

where H_t denotes the hidden state at time step t, encoding contextual information accumulated over previous time steps.

3. Classification Layer

The final hidden representation is mapped to class probabilities using a Softmax function:

$$\hat{y} = Softmax(WH_t + b) \qquad (5)$$

Here W represents the trainable parameter matrix, b denotes the associated bias vector, and $\hat{y}$ refers to the estimated class probability produced by the model.

4. Loss Function

To train the model for multi-class classification, the Sparse Categorical Cross-Entropy loss function is employed:

$$L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \qquad (6)$$

where y_i denotes the ground-truth label and $(\hat{y}_i)$ represents the predicted probability of class i

5. Optimization Strategy

Model parameters are updated using the Adam optimization algorithm, which adaptively adjusts learning rates based on first- and second-order moment estimates to facilitate stable and efficient convergence during training.

V. IMPLEMENTATION STRATEGY AND EXPERIMENTAL SETUP

A. Implementation Framework

The proposed hybrid CNN–LSTM model was implemented using the TensorFlow deep learning framework with the Keras high-level API. This environment provides computational efficiency, scalability, and architectural flexibility, thereby supporting systematic experimentation and reproducible model development for intrusion detection tasks involving sequential log data.

B. Training Configuration

The training process was configured to ensure stable convergence and optimal generalization.

1. **Number of Epochs:**

The model was initially trained for 20 epochs. Experimental analysis indicated rapid convergence with stable validation performance in early epochs. To prevent potential overfitting and reduce computational overhead, the final number of epochs was adjusted based on validation trends.

2. **Batch Size:**

A batch size of 32 was adopted to optimize memory utilization and support consistent gradient estimation throughout the learning process.

3. **Loss Function:**

Sparse Categorical Cross-Entropy was adopted, as defined in (6), due to the multi-class nature of the problem and integer-encoded labels.

4. **Optimizer:**

The Adam optimizer was employed for parameter updates due to its adaptive learning rate mechanism and robust convergence properties.

C. Handling Class Imbalance

To mitigate skewed class distribution, a two-stage imbalance handling strategy was implemented:

1. **Synthetic Minority Oversampling Technique (SMOTE):**

SMOTE was applied exclusively to the training dataset to generate synthetic minority-class samples.

2. **Class Weighting:**

Class weights were incorporated during model training to penalize minority-class misclassification more heavily.

This combined strategy enhanced minority-class detection performance while reducing bias toward majority classes.

D. Evaluation Metrics

This section describes the evaluation metrics used to assess the performance of the proposed security attack detection framework. Multiple metrics are employed to provide a comprehensive evaluation, particularly under class imbalance conditions commonly observed in cybersecurity datasets

1. Confusion Matrix

The performance of the classifier is summarized using a confusion matrix comprising true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). These elements form the basis for standard evaluation metrics .

2. Accuracy

Accuracy evaluates the model's overall predictive performance by determining the proportion of correctly classified observations among all test instances.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \qquad (7)$$

Although widely used, accuracy alone may be misleading in imbalanced datasets .

3. Precision

Precision evaluates the reliability of positive predictions:

$$\text{Precision} = \frac{TP}{TP + FP} \qquad (8)$$

4. Recall

Also termed the detection rate, recall assesses how effectively the model captures positive (attack) instances from the total set of true attack events.

$$\text{Recall} = \frac{TP}{TP + FN} \qquad (9)$$

5. F1-Score

The F1-score provides a harmonic mean of precision and recall:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \qquad (10)$$

It is particularly effective for evaluating imbalanced cybersecurity datasets.

VI. Results and Performance Analysis

A. Confusion Matrix Analysis

The confusion matrix constructed from test predictions demonstrates strong diagonal dominance, indicating that the majority of instances were correctly classified across all severity levels as shown in Figure .Key observations include High true positive rates across dominant classes,Limited inter-class misclassification and Enhanced minority-class detection due to SMOTE and class weighting.The results confirm that the proposed hybrid CNN–LSTM architecture achieves robust multi-class classification performance while maintaining balanced detection capability across varying class distributions.

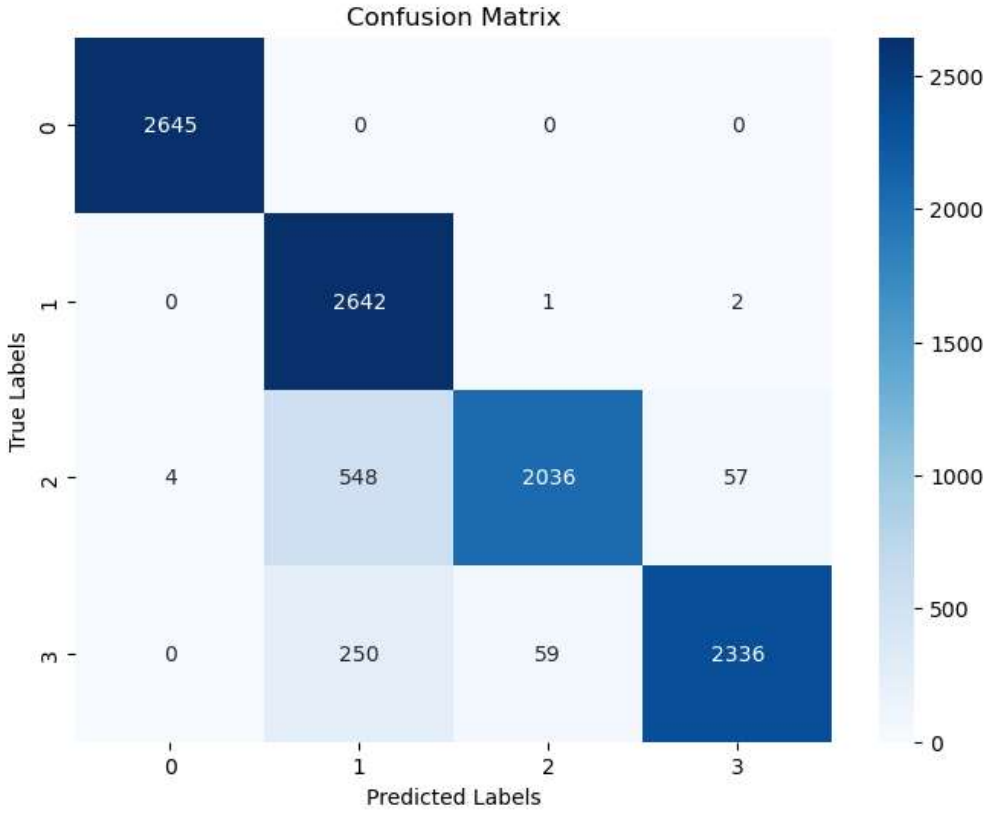

Fig.4 Confusion Matrix

B. Classification Metrics

Table 2. Performance Metrics of the Supervised CNN–LSTM Model

Metric	Value (%)
Accuracy	91.20%
Precision(Weighted Avg)	93%
Recall (Weighted Avg)	91%
F1-Score	91%
Macro F1-Score	91%
Test Samples	10,580

To further evaluate the effectiveness of the proposed supervised CNN–LSTM model, multiple classification metrics were analyzed on the independent test dataset comprising 10,580 samples. The results demonstrate consistent and balanced performance across different evaluation measures.The model achieved an overall accuracy of 91.20%, indicating that the majority of event instances were correctly classified into their respective severity categories. While accuracy provides a general overview of performance, it may not fully reflect model behavior under imbalanced class distributions. Therefore, additional metrics were examined.The weighted precision of 93% suggests that when the model predicts a particular class, the prediction is highly reliable. This indicates a low rate of false positive classifications across event categories. The weighted recall of 91% demonstrates that the model successfully identifies most of the true instances belonging to each class, reflecting strong detection capability.The weighted F1-score of 91% confirms a balanced trade-off between precision and recall, reinforcing the robustness of the classifier. Importantly, the macro F1-score of 91% indicates that performance remains consistent across all classes, including minority categories. This balanced outcome highlights the effectiveness of the implemented class imbalance mitigation strategies, namely SMOTE-based oversampling and class weighting.Overall, the classification metrics analysis validates that the hybrid CNN–LSTM architecture generalizes well to unseen data and maintains stable performance across varying class distributions. The consistency among accuracy, precision, recall, and F1-scores demonstrates that the model does not favor majority classes and achieves reliable multi-class intrusion severity classification.

VII. DISCUSSION AND PRACTICAL IMPLICATIONS

The superior performance of the proposed hybrid CNN–LSTM model can be attributed to the complementary strengths of its architectural components. The convolutional layer effectively extracts structural and local feature representations from Windows Event Logs, while the LSTM layers capture temporal dependencies across sequential events. This combined spatial–temporal learning capability enhances discriminative power for multi-class severity classification.Furthermore, the integration of SMOTE-based oversampling and class-weight optimization mitigates class imbalance bias, contributing to consistent macro-level performance across minority and majority classes.From a practical perspective, the relatively lightweight 1D CNN structure and moderate LSTM depth ensure computational efficiency and low inference latency. These characteristics make the model suitable for real-time deployment scenarios, including streaming log ingestion, sliding-window detection mechanisms, Security Information and Event Management (SIEM) integration, and Security Operations Center (SOC) automation workflows.Overall, the

architecture demonstrates both strong predictive capability and operational feasibility for enterprise-level cybersecurity environments.

VIII. CONCLUSION

This study presented a hybrid CNN–LSTM-based deep learning framework for multi-class classification of structured Windows Event Logs. By integrating convolutional layers for local feature extraction with LSTM layers for temporal dependency modeling, the proposed architecture effectively captured both spatial and sequential characteristics of log data.A comprehensive preprocessing pipeline, including timestamp transformation, data cleaning, categorical encoding, feature normalization, SMOTE-based oversampling, and stratified data splitting, was implemented to enhance data quality and address class imbalance challenges. The combination of synthetic minority oversampling and class weighting significantly improved minority-class detection performance.Experimental results demonstrated that the proposed model achieved robust classification performance, with an overall accuracy exceeding 91% and consistent precision, recall, and F1-scores across classes. The strong macro F1-score further indicates balanced performance even under imbalanced data conditions.Overall, the findings confirm that hybrid deep learning architectures offer a reliable and scalable solution for security event severity classification in intrusion detection systems. Future work may focus on real-time deployment, attention-based mechanisms, or transformer-based architectures to further enhance detection accuracy and adaptability in dynamic cybersecurity environments.

IX. FUTURE WORK

Although the proposed CNN–LSTM model achieved strong classification performance, several directions may further enhance its effectiveness.First, incorporating attention mechanisms or transformer-based architectures could improve the modeling of long-range dependencies within log sequences. Second, extending the framework to real-time streaming environments would increase its practical applicability in operational intrusion detection systems.Additionally, integrating explainable AI techniques may enhance model transparency and support security analysts in understanding classification decisions. Finally, evaluating the model on diverse cybersecurity datasets and applying automated hyperparameter optimization could further improve robustness and generalization capability.These extensions provide promising avenues for strengthening deep learning-based intrusion detection in dynamic security environments.

REFERENCES

[1] A. M. Alashjaee, “Deep learning for network security: An Attention-CNN-LSTM model for accurate intrusion detection,” *Scientific Reports*, vol. 15, Art. no. 21856, 2025.

[2] Y. Wu, B. Zou, and Y. Cao, “Current status, challenges, and future trends of deep learning-based intrusion detection models,” *Journal of Imaging*, vol. 10, no. 10, Art. no. 254, 2024.

[3] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, “Hybrid CNN–BiLSTM-based intrusion detection system,” *Future Generation Computer Systems*, vol. 128, pp. 447–460, 2022.

[4] M. Du, F. Li, G. Zheng, and V. Srikumar, “DeepLog: Anomaly detection and diagnosis from system logs through deep learning,” in *Proc. ACM Conf. Computer and Communications Security (CCS)*, 2017, pp. 1285–1298.

[5] C. Wang, D. Xu, Z. Li, and D. Niyato, "Effective intrusion detection in highly imbalanced IoT networks with lightweight S2CGAN-IDS," *arXiv preprint*, arXiv:2306.03707, 2023.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[7] C. Zhang, J. Li, N. Wang, and D. Zhang, "Intrusion detection based on Transformer and CNN-BiLSTM in IoT," *Sensors*, vol. 25, no. 9, Art. no. 2725, 2025.

[8] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[9] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[10] A. Neto, J. M. R. S. Tavares, and M. I. G. Santos, "Deep learning for intrusion detection in emerging technologies: A comprehensive survey and new perspectives," *Artificial Intelligence Review*, vol. 58, 2025.