Array 30 (2026) 100925

Contents lists available at ScienceDirect

Array

journal homepage: www.sciencedirect.com/journal/array

A lightweight 1D convolutional neural network for malicious network traffic detection on heterogeneous benchmark datasets

Raja Waseem Anwar [a,*], Mohammad Abrar [b], Abdu Salam [c], Faizan Ullah [d]

[a] Department of Computer Science, German University of Technology in Oman, P.O. Box: 1816, P.C, Muscat, 130, Oman
[b] Faculty of Computer Studies, Arab Open University, P.O., Box 1596, Muscat, 122, Oman
[c] Department of Computer Science, Abdul Wali Khan University, Mardan, 23200, Pakistan
[d] Institute of Neuroscience and Medicine 4 (INM-4), Forschungszentrum Julich, Germany

ARTICLE INFO

Keywords:
Intrusion detection
1D convolutional neural network
Malicious traffic detection
Class imbalance
CSE-CIC-IDS2018
UNSW-NB15
Lightweight deep learning
Threshold optimization

ABSTRACT

The rise of advanced cyber threats is creating a need for automated, computationally optimized systems to identify malicious traffic across the network at scale. The available deep learning intrusion detection platforms often incur significant parameter overhead, and few tackle both the challenges of high-dimensional tabular flow features and extreme class imbalance in a single platform. This study proposes a one-dimensional convolutional neural network (1D CNN) that operates directly on tabular features represented as vectors, transforming them into spatially formatted input. The network consists of two convolutional layers with 32 and 16 filters, respectively, global average pooling, and a single sigmoid output node, yielding a compact network suitable for deployment on resource-constrained devices. A strict preprocessing pipeline manages data heterogeneity using correlation-based feature filtering, MinMax scaling, and one-hot encoding. A training strategy that is imbalance-aware combines inverse-frequency weighting of classes, conditional synthetic minority oversampling, and validation-based optimization of the decision threshold. The recommended framework is tested on two standard datasets, CSE-CIC-IDS2018 and UNSW-NB15. Binary detection achieved ROC-AUC scores of 0.9782 and 0.9990 on CSE-CIC-IDS2018 and UNSW-NB15, respectively, with corresponding balanced accuracies of 0.9497 and 0.9967. The supplementary multiclass experiment on CSE-CIC-IDS2018 with a small multilayer perceptron achieves 95.2% accuracy across 13 traffic types, with a macro-ROC-AUC of 0.9857 and only 55,053 trainable parameters, further indicating that the proposed preprocessing and training strategy is generally applicable. The findings underscore the feasibility of lightweight convolutional networks in network intrusion detection.

1. Introduction

The traffic in modern enterprise networks is much faster than rule-based security systems can inspect. Handcrafted signature-based intrusion detection systems (IDSs) cannot be generalized to zero-day threats or polymorphic attack patterns, which has led to the need for long-term research on data-driven detection systems [1]. Machine learning and, more recently, deep learning models can automatically recognize statistical patterns in large-scale traffic logs, without explicit attack signatures [2].

Even though much has changed, there are still many challenges. To start with, network traffic data are inherently high-dimensional and tabular, with dozens of engineered flow-level features per connection record. Direct implementation of deep architectures for image or sequence data cannot be performed without proper reshaping and preprocessing [3]. Second, in practice, traffic distributions are grossly skewed: benign flows can easily outnumber malicious ones by orders of magnitude, and simple models will have high nominal accuracy but low accuracy for the minority group [4]. Third, real-world execution has severe constraints on computational budgets; architectures with millions of parameters do not fit edge or embedded security appliances [5]. Another nuance concerns the choice of the decision threshold. When the distribution of training and test classes is not similar, the default sigmoid threshold of 0.5 is hardly optimal. A principled method for calibrating sensitivity and specificity to the deployment context is threshold optimization on a held-out validation set with respect to balanced accuracy [6].

This paper aims to develop a computationally efficient and precise

* Corresponding author.
E-mail address: raja.anwar@gutech.edu.om (R.W. Anwar).

https://doi.org/10.1016/j.array.2026.100925
Received 15 April 2026; Received in revised form 16 May 2026; Accepted 18 May 2026
Available online 23 May 2026

framework for detecting malicious network traffic using a lightweight one-dimensional convolutional neural network (1D CNN). The goal is to develop a small-sized model that operates on high-dimensional tabular flow characteristics and achieves a high detection rate despite severe class imbalance. Also, the work aims to develop a single preprocessing and training pipeline, which generalizes across heterogeneous benchmark datasets. Another goal is to examine how threshold validation can be used to enhance balanced performance in detection. Lastly, the suggested framework is tested on various datasets to assess its robustness, scalability, and practicality for real-world use.

This study is not novel in terms of the development of a new convolutional operation. Instead, it presents a compact, repeatable, and imbalance-aware detection framework for tabular network traffic data that is heterogeneous. The main contributions are as follows:

- A lightweight 1D CNN framework is designed for binary malicious traffic detection using vectorized flow-level features reshaped as single-channel pseudo-sequences.
- A unified preprocessing pipeline is developed to handle heterogeneous benchmark datasets through duplicate removal, invalid-value cleaning, correlation-based feature reduction, MinMax scaling, and categorical encoding.
- An imbalance-aware training and calibration strategy is introduced by combining inverse-frequency class weighting, conditional SMOTE, and validation-guided threshold optimization.
- The framework is evaluated on CSE-CIC-IDS2018 and UNSW-NB15, with additional ablation, threshold sensitivity, and computational-efficiency analyses to support the claimed lightweight behavior.

The rest of this paper is structured as follows. Section 2 is a review of related work. Section 3 describes the datasets, the preprocessing pipeline, the proposed architecture, and the training strategy. Section 4 introduces and discusses experimental findings. Section 5 concludes with future directions for work.

2. Related work

The use of machine learning in intrusion detection dates back more than twenty years. The classical methods use shallow classifiers, decision trees, random forests, and support vector machines, using manually engineered flow statistics [7]. These approaches are interpretable and incur low inference costs; however, their discriminative performance degrades with more complex attack patterns and larger data sizes. Recurrent networks, especially the long short-term memory (LSTM) networks, are deep learning techniques that capture time-varying dependence in traffic sequences [8]. Hybrid CNN-LSTM models combine local feature extraction with a sequential context encoder and have been shown to be more effective than recurrent baselines on the CIC-IDS2017 and similar datasets [9]. Intrusion detection pipelines have also been incorporated with attention mechanisms to assign dynamic weights to the contributions of individual flow features [10].

A very similar line of inquiry is the use of one-dimensional convolutions on network traffic. Abdulhammed et al. [11] have shown that 1D CNNs, when applied at the packet level to sequences of individual bytes, can achieve competitive detectability. Later work has studied such architectures at the flow-feature level, where tabular flow vectors are transformed into pseudo-sequential representations that can be processed by convolutional methods [12]. The rationale behind this method is that neighboring elements in a sorted or grouped feature vector can share correlated local information, much like neighboring elements in an image exhibit spatial locality.

The benchmark data have been at the forefront of efforts to support reproducible assessment. The Canadian Institute of Cybersecurity released the CSE-CIC-IDS2018 dataset [13], consisting of approximately 8 million labeled network flow records across 14 attack types, generated on a simulated enterprise topology. The UNSW-NB15 dataset [14], produced by the Australian Center of Cyber Security, can also serve as a complementary benchmark, in which 9 types of attacks are considered, and feature engineering is performed differently. Both datasets exhibit high class imbalance and thus serve as suitable testbeds for imbalance-aware detection techniques. Ahmad et al, [15] used supervised machine learning with application and transport layer features from the UNSW NB15 dataset for IoT intrusion detection.

The class imbalance in intrusion detection datasets has been addressed using a number of approaches. The most popular oversampling method is SMOTE [4], which interpolates between existing examples to generate synthetic minority-class examples. Another solution, which does not involve data augmentation and is computationally less costly for large datasets, is cost-sensitive learning, using class-weighted loss functions [16]. In recent work, these strategies have been selectively applied, with oversampling used only when the imbalance ratio exceeds a threshold at which cost-sensitive learning is no longer adequate. High-dimensional flow data are preprocessed by feature selection and reduction. Correlation-based feature elimination is used to reduce redundancy and multicollinearity, and does not require any information about feature importance [17]. This method is orthogonal to variance-based filtering and has been demonstrated to enhance training performance and generalization in tabular deep learning pipelines. Optimizing a binary classifier's threshold has received relatively little attention in the IDS literature, even though it is crucial in an imbalanced environment. The default standard practice uses 0.5; however, a more principled calibration mechanism has been proposed through validation-informed threshold search with respect to balanced accuracy or F-measure, as advocated in the imbalanced learning literature [6]. Recent studies on intrusion detection have been on distributed and intelligent models of resource-constrained environments. As an example, intrusion detection in both IoT-enabled wireless sensor networks and industrial IoT, using federated LSTM-based techniques, has been investigated in terms of privacy-preserving learning and robustness across a variety of datasets [18,19]. There is also a suggestion that trust-aware anomaly detection is necessary to improve the reliability of IDS in IoT-enabled WSNs [20].

The current study combines the following strands: a lightweight 1D CNN model, a comprehensive and reproducible preprocessing pipeline, and an imbalance-aware training and calibration scheme, which is assessed on two benchmark datasets to provide a general and transparent characterization of performance.

3. Materials and methods

Fig. 1 shows the general approach of the proposed malicious traffic detection framework. Raw network traffic data is read into the pipeline and then processed through a sequence of preprocessing stages, including data cleaning, feature selection, normalization, and encoding, to create a single feature matrix. The processed data are then partitioned via stratified splitting, and class imbalance is addressed using inverse-frequency weighting and conditional SMOTE. The resulting data is then used to train a small one-dimensional convolutional neural network (1D CNN), with early stopping and adaptive learning-rate scheduling to guide training. Finally, threshold optimization with validations is performed to locate the optimal decision boundary, and the model is tested on a held-out test set with standard classification measures.

3.1. Datasets

This study uses two publicly available benchmark datasets, as shown in Table 1. The CSE-CIC-IDS2018 [21], developed by the Canadian Institute of Cybersecurity, was based on a network simulation that modeled realistic enterprise traffic. The dataset includes about 7.3 million flow records, each with 14 tags, including benign traffic and 13 attack types. Raw feature files include 80 designed flow attributes

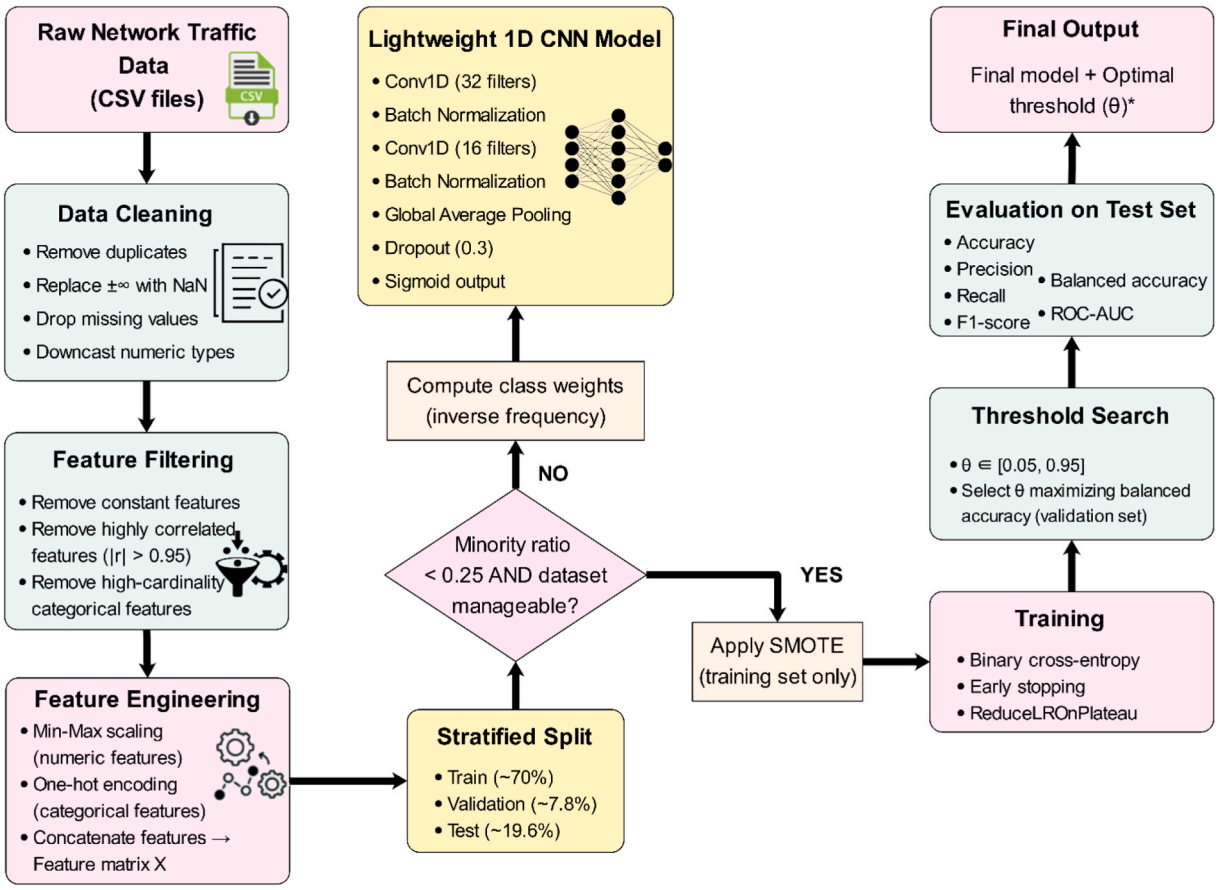

Fig. 1. Methodology flowchart of the proposed malicious traffic detection framework.

calculated by CICFlowMeter. The sample is highly imbalanced across classes, with benign flows occupying approximately 83 percent of the data. UNSW-NB15 [22] was developed by the Australian Center of Cyber Security using the IXIA PerfectStorm tool to create a mixture of realistic, modern normal, and attack operations, resulting in a normal-attack behavior mixture. It consists of nearly 2.54 million flow records in nine attack categories and normal traffic, characterized by 49 original features. The dataset is also highly skewed, with most records in the majority class. Binary labels are generated for both datasets, with all attack records set to 1 and all benign records set to 0. In the subsequent multiclass experiment on CSE-CIC-IDS2018, the 14-class label set (including benign) is reduced to 13 useful classes by removing empty classes during preprocessing.

3.2. Data preprocessing and feature engineering

The preprocessing and feature engineering phase aimed to generate a clean, condensed, and consistent representation of the input data for heterogeneous network intrusion datasets, while ensuring reproducibility. A common, but dataset-flexible, preprocessing pipeline was used prior to model training since the feature structure, categorical attributes, feature ranges, and label formats are different for both CSE CIC IDS2018 and UNSW NB15. The goal of this stage was to eliminate irrelevant and problematic data, reduce dimensionality, normalize numerical data, encode categorical data, and train the data to be imbalance-aware.

To minimize data leakage and avoid overfitting on the test set, duplicate records were first removed. The infinite values were replaced with missing values where they occurred in calculations based on the flow, such as byte rate and packet rate, when the denominator is zero. After that, records with any missing values were discarded, as the final feature matrix only included numerical and encoded categorical data. Where possible, numerical columns were downcast during processing to save memory.

The feature filtering was then carried out in three steps. First, features of the constant numbers have been discarded because they provide no discriminative information for classification. Secondly, high correlation numeric features were determined by the Pearson correlation coefficient. If the absolute correlation between two numeric features exceeded 0.95, one was dropped as redundant. Third, categorical data with more than 100 unique categories were not included in the dataset to prevent overfitting during one-hot encoding. This filtering process helped to reduce the dimensionality of the input while keeping the most informative flow characteristics.

All numeric features were minmax scaled to the interval [0, 1] after filtering. This step ensured that features with wide numeric intervals didn't dominate the optimization. Retained categorical features were transformed with one-hot encoding, and unknown categories were not encoded for inference. The scaled numeric matrix and the encoded categorical matrix were then horizontally concatenated to create a final input feature matrix. For the 1D CNN, this feature matrix was expanded (n, d) to $(n, d, 1)$ with n being the number of samples, and d being the number of features to keep after preprocessing. Processed data was divided into training, validation, and test sets using stratified sampling, so that the number of samples in each set retains the same class distribution as in the original set. 70% of the data was used for training, 7.8% for validation, and 19.6% for final testing. The validation set was exclusively for model monitoring and threshold selection, whereas the test set was left out until the final evaluation. Addressing class imbalance: inverse-frequency class weighting and conditional SMOTE. Class weights were calculated only from the training labels and applied during model training to make minority-class errors more costly. SMOTE was used only in the training partition, not in the validation or test

Table 1
Preprocessing and training Configuration of the proposed framework.

Configuration aspect	Details
Dataset cleaning	Duplicate records were removed. Infinite values were replaced with missing values, and all records containing missing values were removed. Numerical columns were downcast where possible to reduce memory usage.
Feature filtering	Constant numeric features were removed. Highly correlated numeric features were filtered using Pearson correlation, with one feature removed when absolute correlation exceeded 0.95. Categorical variables with more than 100 unique values were removed to avoid excessive one hot encoding expansion.
Numeric scaling	All retained numeric features were scaled to the range [0, 1] using MinMax normalization.
Categorical encoding	Retained categorical features were transformed using one hot encoding with handle_unknown = ignore to support unseen categories during inference.
Final feature matrix	Scaled numeric features and one hot encoded categorical features were concatenated to form the final feature matrix. For the 1D CNN, the matrix was reshaped from (n, d) to (n, d, 1).
Data splitting	Stratified splitting was used to preserve class distribution across subsets. Approximately 70 percent of the data was used for training, 7.8 percent for validation, and 19.6 percent for testing.
Imbalance handling	Inverse frequency class weights were computed from the training labels. SMOTE was applied only to the training set when the minority to majority ratio was below 0.25 and the cleaned training size was below 1,000,000 samples. Otherwise, SMOTE was skipped and class weighting was used.
Model input format	The processed feature vector was treated as a single channel pseudo sequence and reshaped into (d, 1) for each sample before being passed to the 1D CNN.
Optimizer and loss	Adam optimizer was used with an initial learning rate of 1×10^{-3}. Binary cross entropy was used as the loss function for binary detection.
Batch size and epochs	The model was trained with a batch size of 2048 for a maximum of 20 epochs.
Training callbacks	Early stopping monitored validation AUC with patience of 4 epochs and restored the best model weights. ReduceLROnPlateau monitored validation AUC with a reduction factor of 0.5, patience of 2 epochs, and minimum learning rate of 1×10^{-5}.
Threshold optimization	Candidate thresholds from 0.05 to 0.95 were evaluated with a step size of 0.01. The threshold with the highest validation balanced accuracy was selected and then fixed for final test evaluation.
Test evaluation	The final model was evaluated once on the held out test set using the selected validation threshold. Reported metrics included accuracy, precision, recall, F1 score, balanced accuracy, ROC AUC, confusion matrix, and precision recall behavior.

partitions. To make this step repeatable, for the training set only, SMOTE was applied if the minority-to-majority class ratio was less than 0.25 and the number of training samples was less than 1,000,000. If any of the conditions were not met, SMOTE was not applied; only class weighting was used. This rule is useful for avoiding excessive synthetic samples in very large datasets and for ensuring that the imbalance-handling strategy is neither overly costly nor non-reproducible.

The last pre-processing pipeline process is then finally implemented as a combination of data cleaning, data redundancy reduction, data normalization, categorical encoding, stratified splitting, and imbalance-aware training preparation. This design enables the same processing logic to be used for benchmark datasets of various types, without introducing dataset leakage and with a fixed evaluation protocol.

3.3. Proposed lightweight CNN architecture

The suggested binary detection model is a one-dimensional convolutional neural network (1D CNN). The reason for using 1D convolutions on tabular flow features is that they can capture local patterns across neighboring feature dimensions without incurring the large parameter count of fully connected layers. The feature vector of dimension d after MinMax scaling and concatenation is rearranged to (d, 1) to create a single-channel pseudo-sequence onto which convolutional filters are sliding.

The structure, as illustrated in Fig. 2, is the following. Tensors of size (d, 1) are accepted in the input layer. The initial convolutional layer uses 32 filters with a kernel size of 3, same padding, ReLU activation, and batch normalization. The second convolutional layer then reduces the number of filters to 16, using the same kernel size, padding, and activation, followed by a second batch normalization layer. Global average pooling then reduces the spatial dimension by averaging the activation across all feature positions for each filter, yielding a 16-dimensional vector. Regularization is achieved using a dropout layer with a rate of 0.30. This is because a single Dense neuron with a sigmoid activation is used to produce the output and is appropriate when binary detection is required.

This architecture is designed to be small. Global average pooling replaces the large fully connected layers that would otherwise precede a flattening operation, significantly reducing the number of parameters compared to standard CNN structures. The gradual decrease from 32 to 16 filters promotes a pyramidal structure of abstractions and helps regulate the model's complexity. The reported model contains only the layers shown in Fig. 2. No attention module is included in the final experimental architecture.

Algorithm 1 below formalizes the complete detection framework.

Algorithm 1. Proposed Lightweight Malicious Traffic Detection Framework

Input: Raw network traffic flow dataset D
Output: Trained model M, optimal threshold θ^*, evaluation metrics
1. $D \leftarrow$ load raw network traffic dataset; apply dataset-specific column mapping and label normalization
2. Remove duplicate records; replace $\pm\infty$ with NaN; drop rows with any NaN
3. Remove constant-valued numeric features
4. Remove numeric features with pairwise Pearson $|r| > 0.95$
5. Remove categorical features with unique cardinality > 100
6. Apply MinMax normalization to retained numeric features
7. Apply OneHotEncoder to categorical features with handle_unknown = "ignore"
8. Concatenate numeric and one-hot-encoded feature matrices $\rightarrow X$
9. Perform stratified split of (X, y) into $(X_{\text{train}}, y_{\text{train}})$, $(X_{\text{val}}, y_{\text{val}})$, and $(X_{\text{test}}, y_{\text{test}})$
10. Compute inverse-frequency class weights W from y_{train}
11. if minority-to-majority ratio < 0.25 and $|X_{\text{train}}|$ is manageable then $(X_{\text{train}}, y_{\text{train}}) \leftarrow \text{SMOTE}(X_{\text{train}}, y_{\text{train}})$
12. Reshape $X_{\text{train}} \rightarrow (|X_{\text{train}}|, d, 1)$; build lightweight 1D CNN model M
13. Train M on $(X_{\text{train}}, y_{\text{train}})$ using class weights W, Adam optimizer, early stopping, and ReduceLROnPlateau on validation AUC
14. for $\theta \in \{0.05, 0.06, \ldots, 0.95\}$ do
    Compute balanced accuracy on $(X_{\text{val}}, y_{\text{val}})$ using threshold θ
15. $\theta^* \leftarrow \arg\max_\theta$ balanced accuracy on the validation set
16. Evaluate M on X_{test} using θ^*; report Accuracy, Precision, Recall, F1-score, Balanced Accuracy, and ROC-AUC
17. Return M, θ^*, evaluation metrics

3.4. Training strategy and threshold optimization

The model is trained using the Adam optimizer with an initial learning rate of 1×10^{-3}, binary cross-entropy loss, and binary accuracy and AUC as monitoring. The training continues until 20 epochs, with a batch size of 2048. The training is controlled by two callbacks. Early stopping checks the validation AUC with a patience of 4 epochs and restores the weights from the epoch with the optimal validation AUC. ReduceLROnPlateau is used with a factor of 0.5, a patience of 2 epochs, and a minimum learning rate floor of 1×10^{-5}, which enables fine-grained convergence during subsequent training epochs. After the training, a threshold search using validation is done to calibrate the sigmoid output. Thresholds were evaluated over the interval [0.05,

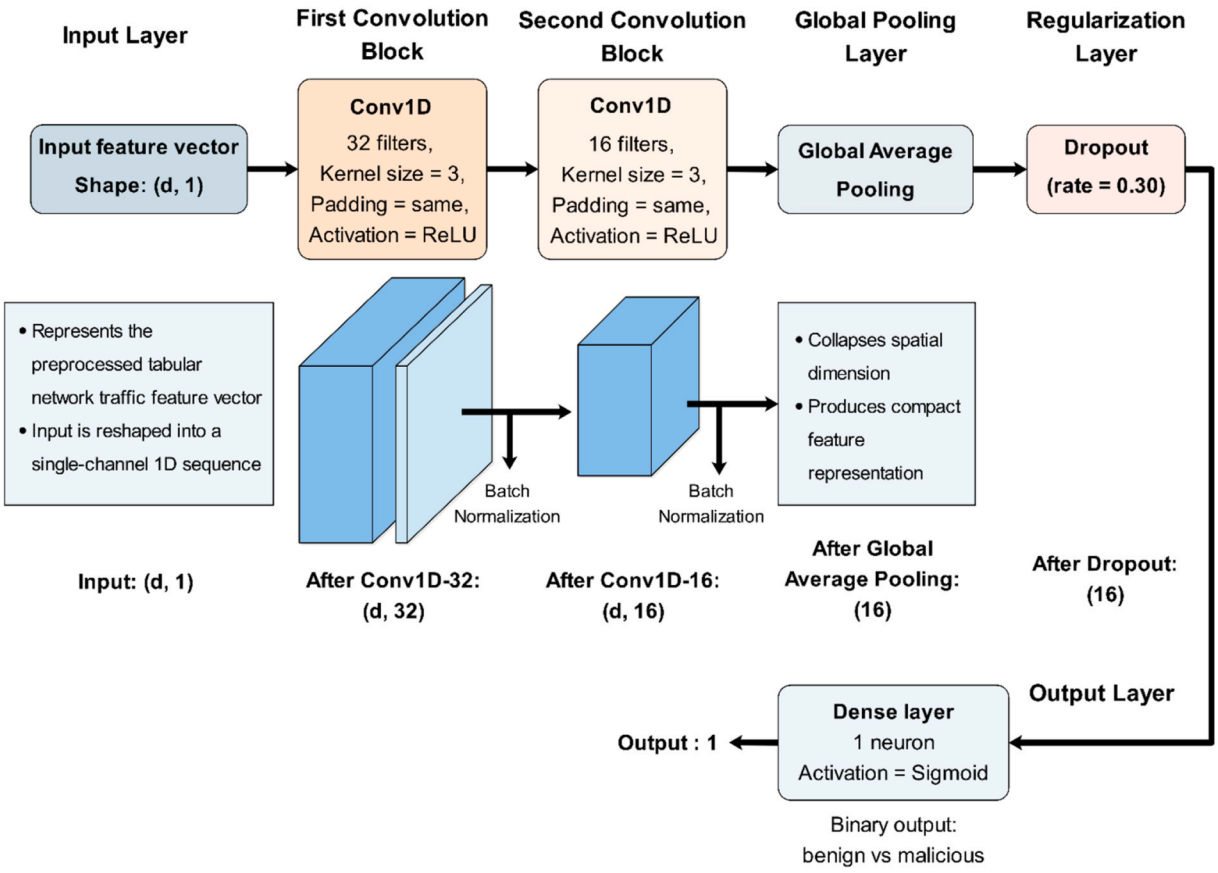

Fig. 2. Architecture of the proposed lightweight 1D CNN model for malicious traffic detection.

0.95] with a step size of 0.01. The threshold was selected only on the validation set by maximizing balanced accuracy and was then fixed before final test-set evaluation.

Predictions are binarized at every candidate threshold, and the balanced accuracy is calculated on the validation set. The threshold that yields the highest balanced accuracy is chosen as the operating threshold θ^* for test-set evaluation. The result of this process is that classifier training is not tied to the choice of decision boundary but instead generates a threshold optimized to the imbalance properties of each dataset.

3.5. *Supplementary multiclass generalizability experiment*

To determine whether the suggested preprocessing and training methodology can be generalized beyond binary detection, an additional experiment trains a compact multilayer perceptron (MLP) on the multiclass labels of the CSE-CIC-IDS2018 dataset. In this experiment, the same preprocessing pipeline is used except that the 1D CNN is substituted by a four-layer MLP: Dense (256) batch normalization and 0.30 dropout, Dense (128) batch normalization and 0.25 dropout, Dense (64) batch normalization and 0.20 dropout and a Dense (13) softmax output layer Multi-class imbalance structure is more complicated and focal loss is trained in the model [23]. This additional experiment is not the main contribution of the paper; it is a testimony to the fact that the preprocessing and training pipeline is universal.

4. Results and discussion

The experiments are all implemented in Python, using TensorFlow/Keras to build the models, scikit-learn for preprocessing and evaluation, and imbalanced-learn for SMOTE. Training models takes place on the GPU hardware. All stochastic operations are performed with fixed random seeds to ensure reproducibility. Accuracy, precision, recall, F1-score, balanced accuracy, and ROC-AUC-binary have been reported as evaluation metrics; precision, recall, and F1 have been reported in macro- and weighted variants, and accuracy with the top-3 and macro one-vs-rest ROC-AUC have been reported in the multiclass experiment.

Fig. 3 presents the learning behavior of the proposed model on the two benchmark datasets. As shown in Fig. 3(a), the loss curves of CSE-CIC-IDS2018 reveal that the training loss decreases more consistently across epochs, whereas the validation loss increases more sharply in the initial stages and then returns to a lower value. This trend indicates that the model is slowly learning to discriminate features and is finally reaching a stable point, with minimal variation between training and validation performance. Fig. 3(b) depicts the loss curves of UNSW-NB15 that decreased rapidly during the first few epochs and then level off at extremely low values. The close correspondence between the training and validation loss points to rapid convergence, strong generalization, and minimal signs of training instability. Fig. 3(c) shows that at epochs, both training and validation AUC curves of CSE-CIC-IDS2018 grow steadily. The validation AUC is slightly higher than the training AUC for most of the training process, indicating that the model has strong discriminative power and that no significant overfitting occurs. Fig. 3(d) shows that the AUC curves of UNSW-NB15 achieve near-perfect values at a rather early stage of training and then stay practically unchanged. The behavior indicates that the model is highly separable on this dataset and that the classifier is in a very stable state after a few epochs. In general, the four subplots demonstrate that the proposed lightweight 1D CNN can be effectively trained on both datasets and exhibits a smooth optimization process, high discriminative ability, and good validation performance throughout training.

The confusion matrices of binary classification on the two benchmark datasets are shown in Fig. 4. In Fig. 4(a), which represents CSE-CIC-IDS2018, the model accurately categorizes a very large number of

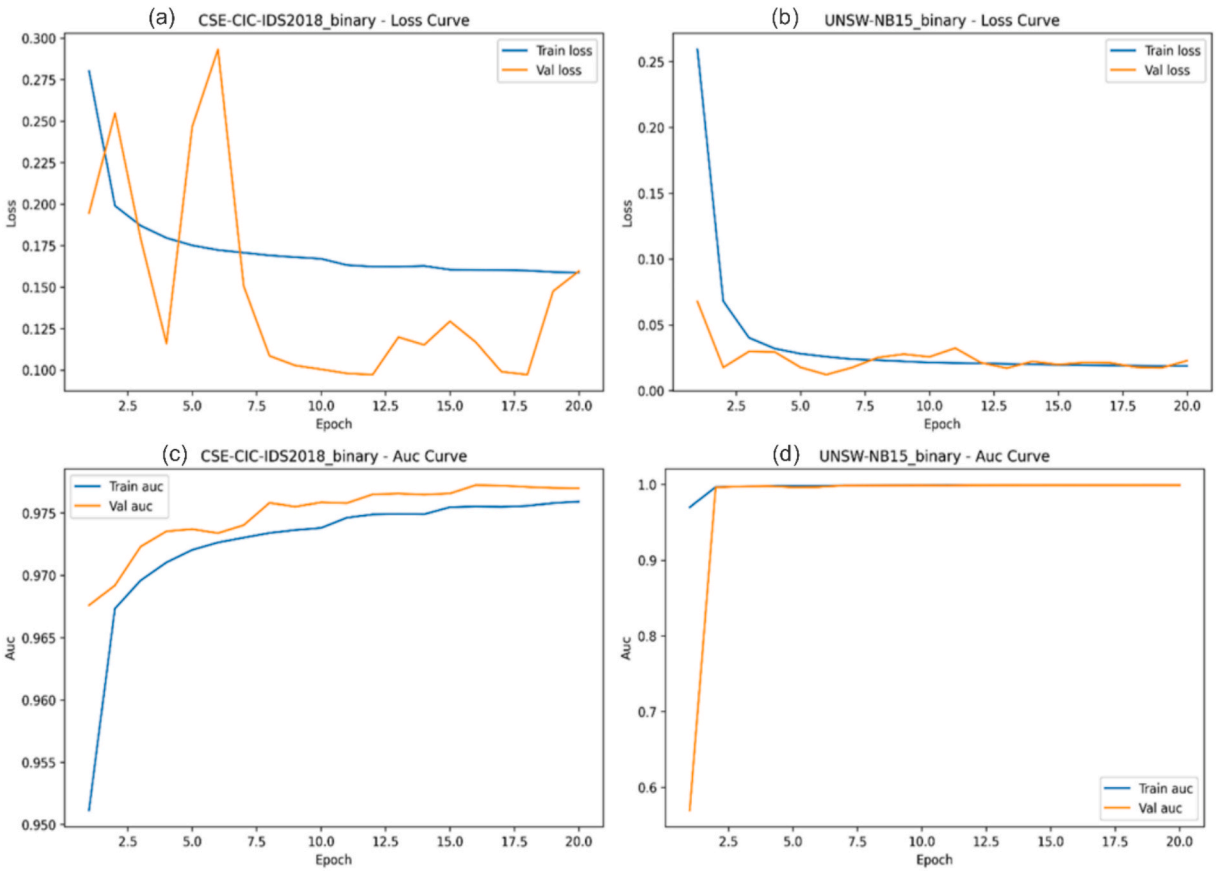

Fig. 3. Training dynamics of the proposed lightweight 1D CNN for binary intrusion detection on CSE-CIC-IDS2018 and UNSW-NB15. Subplots show training and validation loss and AUC across 20 epochs: **(a)** loss curve for CSE-CIC-IDS2018, **(b)** loss curve for UNSW-NB15, **(c)** AUC curve for CSE-CIC-IDS2018, and **(d)** AUC curve for UNSW-NB15.

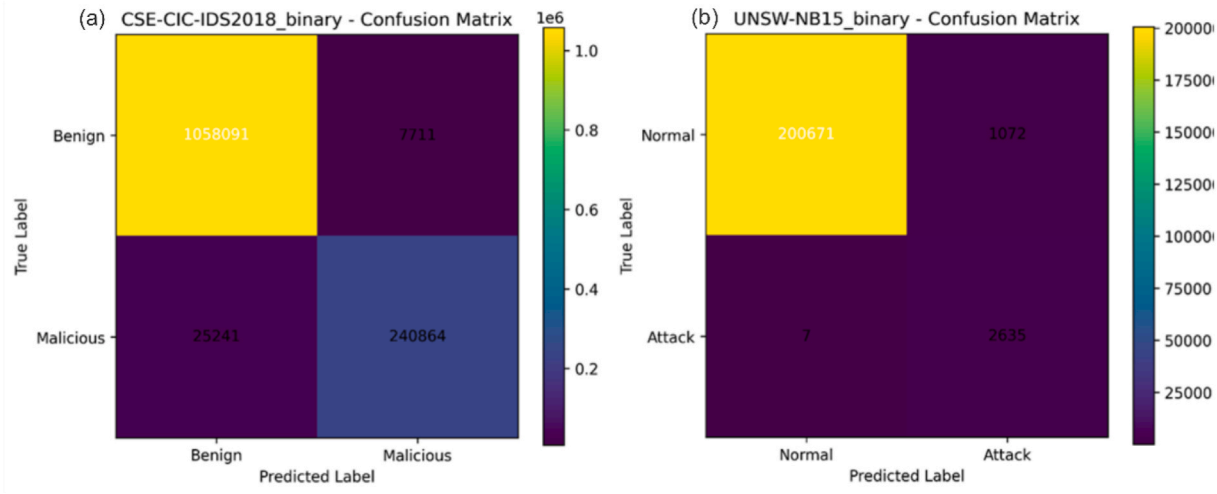

Fig. 4. Confusion matrices of the proposed lightweight 1D CNN for binary intrusion detection on the test sets of **(a)** CSE-CIC-IDS2018 and **(b)** UNSW-NB15. The matrices summarize the numbers of correctly and incorrectly classified benign/normal and malicious/attack flows.

benign and malicious flows, with 1,058,091 true negatives and 240,864 true positives. False positives are relatively low (7,711), and false negatives are higher (25,241), indicating that the classifier has a high overall detection rate but still misses some malicious traffic. This finding is consistent with the high and balanced accuracies reported and indicates that the chosen operating threshold favors a conservative trade-off with low false-alarm rates. Fig. 4(b), which corresponds to UNSW-NB15, indicates that the total number of true negatives is 200,671, and the total number of true positives is 2,635, with only 1072 false positives and 7 false negatives. This distribution shows that the attack-detection ability is extremely strong, particularly with respect to recall, as virtually all attack samples are correctly detected. The minimal false-negative rates indicate that the chosen threshold makes the model highly sensitive to malicious traffic, which is a desirable attribute in a

security-oriented deployment environment where unidentified attacks are costly.

The discriminative ability of the proposed model is depicted using ROC and precision-recall analysis on the two benchmark datasets, as shown in Fig. 5. Fig. 5(a) shows that the ROC curve of CSE-CIC-IDS2018 increases steeply toward the upper-left corner and has an AUC of 0.9780, indicating that benign and malicious traffic are largely separated across a broad range of decision thresholds. This finding confirms that the model has high ranking quality prior to the implementation of the ultimate operating threshold. This means that the ROC curve of UNSW-NB15 is even nearer to the optimal top-left corner in Fig. 5(b), with an AUC of 0.9989, and can divide normal and attack traffic almost perfectly. This indicates that the classifier assigns completely separable prediction scores to the two classes in this dataset. Fig. 5(c) shows that the precision-recall curve of CSE-CIC-IDS2018 is in the upper region for most recall values, with an average precision (AP) of 0.9600. The level of precision remains quite high until the recall approaches its upper threshold, after which a more pronounced decrease is observed. The behavior demonstrates that the model maintains a desirable trade-off between false alarms and detection of malicious traffic within a wide operating range. Fig. 5(d) shows that the UNSW-NB15 precision-recall curve has an AP of 0.8848. Despite the near-perfect ROC score, the precision-recall curve shows a smoother drop in recall, reflecting greater class imbalance and the sensitivity of precision to false positives in the positive class. Nevertheless, the curve remains high across most of the recall range, validating strong attack-detection capabilities.

Fig. 6 investigates the distribution of model prediction scores and how the choice of a threshold influences classification performance. Fig. 6(a) indicates that the distribution of predicted scores of CSE-CIC-IDS2018 is mostly centered around low predicted-probabilities as most benign samples, and around high predicted-probabilities near 1.0 as most malicious samples. Though the two classes are mostly distinct, there is still a middle ground, which is why there are false positives and false negatives in the final classification results. Fig. 6(b) shows the distribution of prediction scores for UNSW-NB15, with even more distinct class separation. Normal samples are tightly clustered around 0, while attack samples are clustered around 1, with little overlap between the two classes. This distribution is consistent with a near-perfect ROC-AUC and very high recall on this dataset. Fig. 6(c) presents the threshold-tuning curves for CSE-CIC-IDS2018. As the threshold increases, precision improves while recall gradually decreases, reflecting the expected trade-off between reducing false positives and avoiding missed attacks. The selected threshold, $\theta = 0.68$, was obtained exclusively from the validation set by maximizing balanced accuracy and was then fixed for test-set evaluation. Fig. 6(d) shows the corresponding threshold-tuning behavior for UNSW-NB15, where the selected threshold is lower, $\theta = 0.29$. This lower threshold prioritizes attack sensitivity and explains the very high recall observed on UNSW-NB15, although it also contributes to a lower precision value.

Table 2 indicates that the proposed model provides a balance in terms of detection performance and architectural simplicity. The results of the CSE CIC IDS2018 indicate the high binary classification performance i.e. the high value of precision, which implies that the model can efficiently control the false alarms on this data set. The UNSW NB15

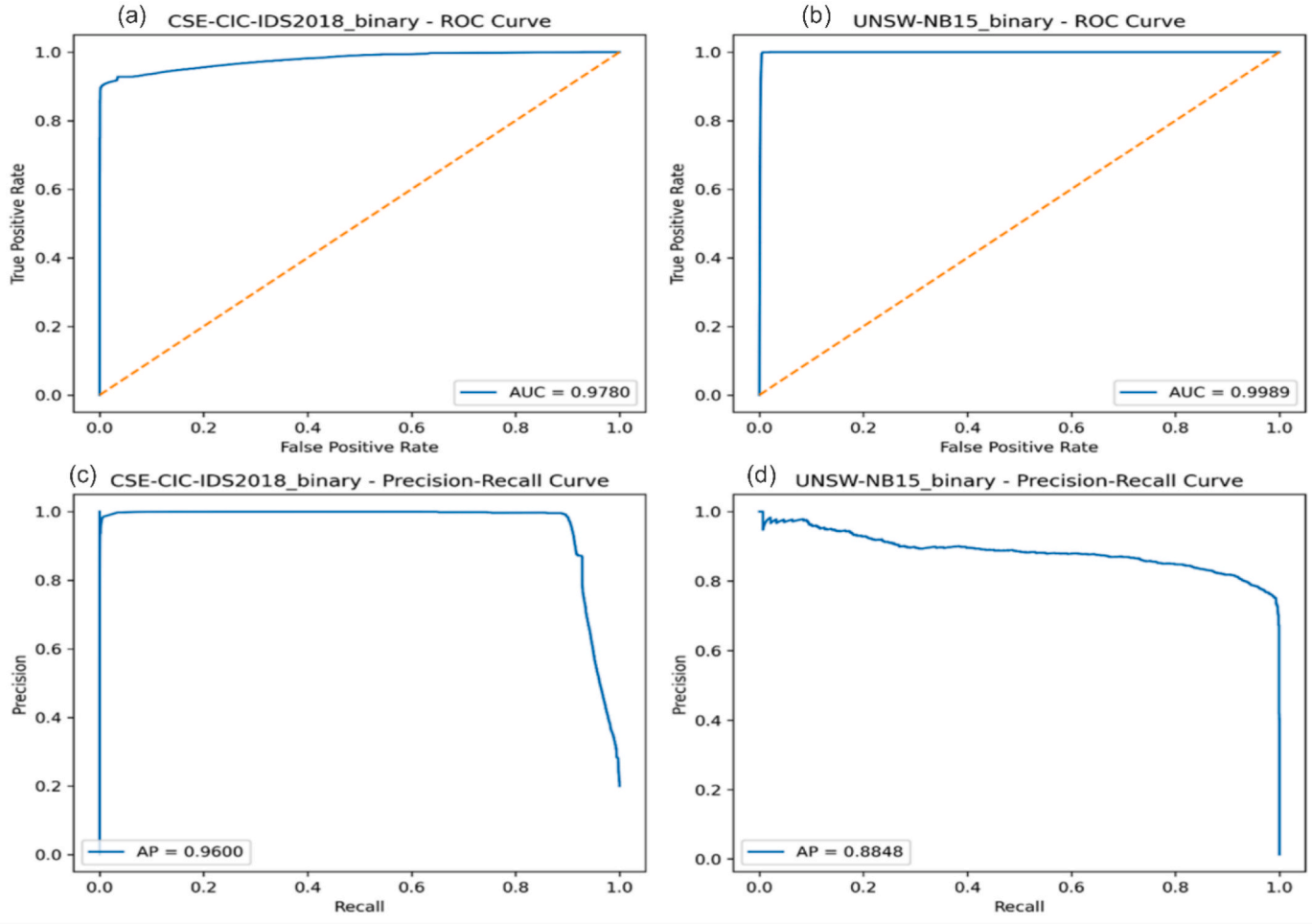

Fig. 5. Discriminative performance of the proposed lightweight 1D CNN for binary intrusion detection on CSE-CIC-IDS2018 and UNSW-NB15. Subplots show **(a)** ROC curve for CSE-CIC-IDS2018, **(b)** ROC curve for UNSW-NB15, **(c)** precision–recall curve for CSE-CIC-IDS2018, and **(d)** precision–recall curve for UNSW-NB15.

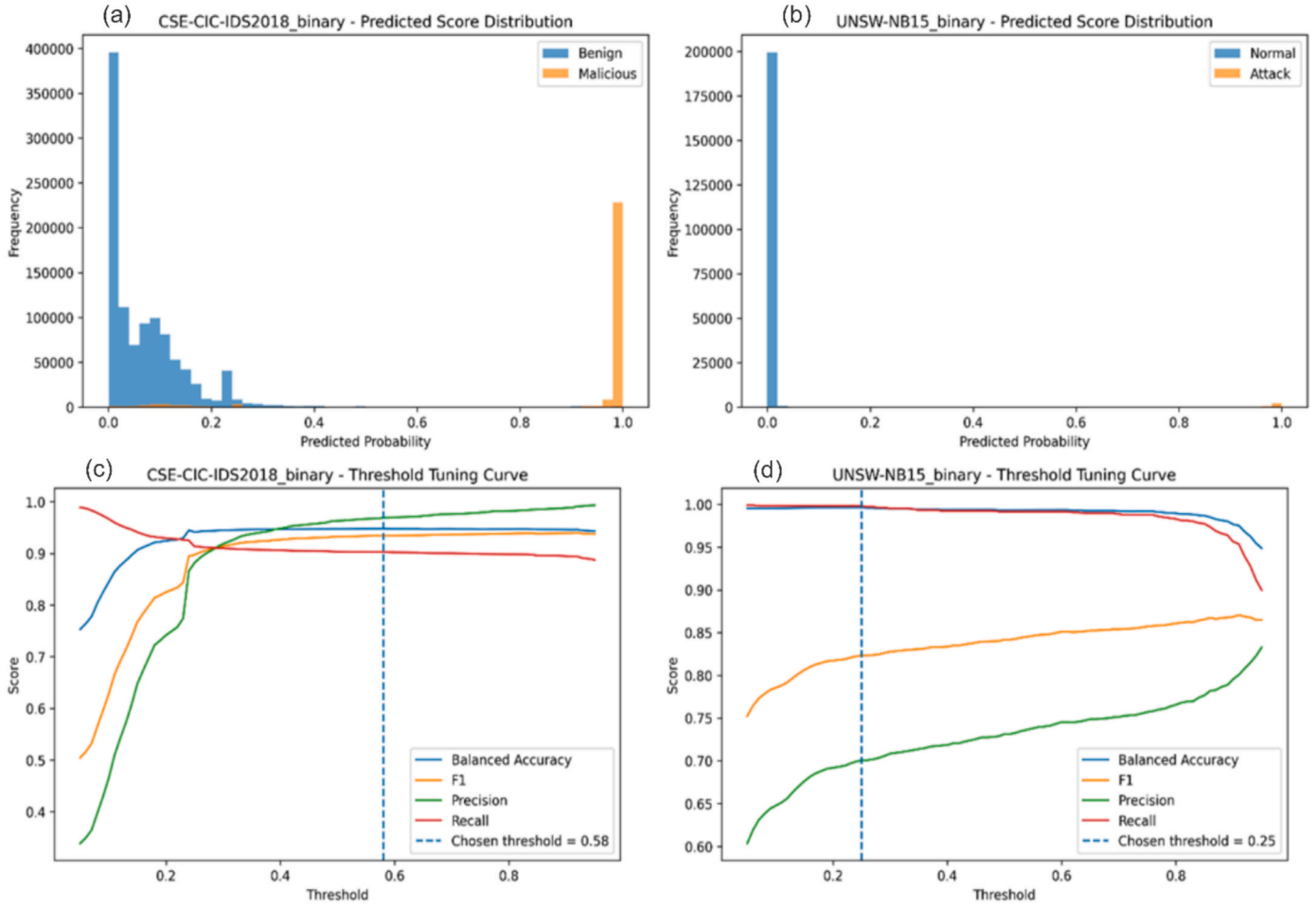

Fig. 6. Predicted-score distributions and threshold-tuning behavior of the proposed lightweight 1D CNN for binary intrusion detection on CSE-CIC-IDS2018 and UNSW-NB15. Subplots show **(a)** predicted probability distributions for CSE-CIC-IDS2018, **(b)** predicted probability distributions for UNSW-NB15, **(c)** threshold-dependent performance curves for CSE-CIC-IDS2018, and **(d)** threshold-dependent performance curves for UNSW-NB15.

result has very high recall and ROC AUC, whereas lower precision reflects the price of choosing the threshold which maximizes sensitivity. Thus, the proposed model cannot be said to be better than all of the previous models. Its major strength lies in its ability to achieve high detection performance with a compact 1D CNN pipeline with explicit pre-processing, imbalance handling, and threshold calibration based on validation.

The comparison also helps to explain why some previous methods cannot be compared directly as they are based on different data sets, detection criteria, or assumptions regarding the architecture. Because of their modeling of feature interactions and contextual representations, the transformer-based methods are relevant, but they have different hybrid architectures and evaluation protocols than the present work.

The cross-dataset applicability of the preprocessing pipeline is also impressive: the same correlation filtering, encoding, and scaling steps can be applied to both CIC-IDS2018 and UNSW-NB15 without dataset-specific tuning, achieving ROC-AUC scores above 0.97 on both datasets. This indicates that the pipeline is resistant to heterogeneity in feature-engineering conventions and traffic-generation methodologies across datasets.

The difference between the two sets at the thresholds $\theta^* = 0.68$ for CIC and $\theta^* = 0.29$ for UNSW indicates how dangerous it can be to use a fixed threshold in deployed IDS systems. The validation-guided search model used in this case is computationally inexpensive and can be restarted either after model retraining or due to data drift, which can be useful in operational practice.

The observed trade-off between precision and recall at $\theta^* = 0.29$ for UNSW-NB15 demonstrates the fundamental tension in intrusion detection design. The more a high-security deployment focuses on zero missed attacks, the more false positives it will accept; the less bandwidth an analyst has, the higher the threshold can be. The optimization framework for the threshold proposed here is transparent and can be adjusted to meet either requirement.

The 1D CNN architecture offers an acceptable tradeoff of model capacity and inference efficiency. Global average pooling removes the dense layers that are usually parameter-intensive and occur after convolutional blocks in image classification networks, ensuring the overall number of parameters remains low. This small code size is directly proportional to shorter inference times and lower memory usage, which is useful in both software-defined networking and edge deployments.

Interpretations of comparisons with published results should be done with care since differences in experimental protocol, especially those introduced by train-test split methodology, near-duplicate records, and normalization of class labels, can create large discrepancies in reported values even on the same nominal dataset. The current assessment is based on a strict stratified splitting algorithm that avoids data spillage and guarantees a representative test-set composition.

5. Conclusion and future work

In this paper, a 1D CNN, a lightweight binary detector for malicious network traffic, is proposed as part of a reproducible preprocessing and imbalance-aware training system. The proposed model achieved ROC-AUC scores of 0.9782 on CSE-CIC-IDS2018 and 0.9990 on UNSW-

Table 2
Comparison of the proposed method with recent state-of-the-art intrusion detection studies.

Method	Dataset or datasets	Detection setting	Accuracy	Precision	Recall	F1 score	ROC AUC	Comparability level	Key interpretation
Federated learning with LSTM [24]	WSN DS, CIC IDS2017, UNSW NB15	Multi dataset IDS	0.9578	0.9550	0.9620	0.9580	NR	Partial comparison	Includes UNSW NB15, but the federated and multi dataset protocol differs from the proposed centralized binary setting.
Federated CNN BiLSTM [25]	X IIoTID, WUSTL IIoT, Edge IIoTset	IIoT intrusion detection	0.9780	0.9690	0.9720	0.9700	0.9780	Contextual only	Strong hybrid deep learning IDS, but evaluated on different IIoT datasets; not directly comparable with CSE CIC IDS2018 or UNSW NB15 results.
TEAD trust enhanced anomaly detection [20]	NSL KDD, CIC IDS2017	Anomaly based IDS	0.9600	NR	NR	NR	NR	Contextual only	Relevant to trust based anomaly detection, but uses different benchmarks and evaluation settings.
IDS INT Transformer based transfer learning with SMOTE, CNN, and CNN LSTM [26]	UNSW NB15, CIC IDS2017, NSL KDD	Imbalanced network intrusion detection	NR	NR	NR	NR	NR	Partial comparison	Relevant Transformer based IDS for imbalanced traffic; uses some overlapping datasets but a different hybrid architecture and evaluation protocol.
Trans IDS Transformer based intrusion detection system [27]	UNSW NB15, NSL KDD	Binary intrusion detection	0.8914 on UNSW NB15	NR	NR	NR	NR	Partial comparison	Uses UNSW NB15 and binary classification, but differs in preprocessing, train validation design, and model structure.
RAGN robust adaptive graph neural network [28]	Malicious network traffic graph setting	Unknown malicious traffic detection	NR	NR	NR	NR	NR	Contextual only	Graph based method for unknown malicious traffic detection; useful for state of the art positioning but not directly comparable with tabular 1D CNN results.
TIPSO GAN optimized generative adversarial network [29]	CIC IDS2018, CICAPT IIoT2024, CIC DDoS2019	Malicious traffic detection and zero day evaluation	NR	NR	NR	0.991 on CIC IDS2018	NR	Contextual only	GAN based imbalance and zero day detection method; relevant to advanced IDS comparison but uses a different generative architecture and broader evaluation protocol.
Proposed Lightweight 1D CNN with unified preprocessing and validation threshold tuning	CSE CIC IDS2018	Binary malicious traffic detection	0.9775	0.9826	0.9033	0.9413	0.9782	Proposed result	Strong binary detection with high precision and high ROC AUC under the proposed stratified evaluation protocol.
Proposed Lightweight 1D CNN with unified preprocessing and validation threshold tuning	UNSW NB15	Binary malicious traffic detection	0.9949	0.7176	0.9985	0.8351	0.9990	Proposed result	Very high recall and near perfect ROC AUC, but lower precision indicates a non negligible false alarm burden.

NB15, with corresponding balanced accuracies of 0.9497 and 0.9967. An additional multiclass experiment on CSE-CIC-IDS2018 using a small MLP further indicates the generalizability of the preprocessing and training strategy, achieving 95.2 percent accuracy and a macro ROC-AUC of 0.9857 with only 55,053 parameters. The most important practical contributions of this work are two. In architecture, the global average pooling layer enables high detection performance without large fully connected layers, resulting in a small model that can be deployed on resource-constrained security appliances. The unified preprocessing pipeline and threshold optimization via validation offer a methodological advantage: a reproducible, dataset-free framework that can be applied to new traffic datasets with minimal adaptation. There are a few limitations. First, the input to the proposed 1D CNN is the processed feature order, and the ablation study with a compact MLP is meant to evaluate the usefulness of the convolutional filtering, the order of tabular features is not as natural as spatial or temporal order. Secondly, the experiments are performed independently on CSE-CIC-IDS2018 and UNSW-NB15, thus the study provides proof of cross-benchmark applicability of the same pipeline but is not a strict cross-dataset transfer evaluation. Third, the high ROC-AUC value on UNSW-NB15 could be due to separability in the dataset, which should be tested with other live or recent traffic traces. Fourth, while the model is compact, it should be reported how big the model is, how long it takes to make an inference, and how much memory it consumes when deployed. Finally, the supplementary multiclass experiment is performed with an MLP rather than the proposed 1D CNN and is thus considered only as support for the preprocessing pipeline.

Several directions will be investigated in the future. To assess whether the additional complexity can consistently improve performance on datasets with more feature interactions, the optional squeeze-and-excitation attention branch will be tested. Flow features that play the most significant role in the detection decision will be explained using SHAP or Integrated Gradients techniques, which, in turn, will help build an analyst's trust and support regulatory compliance. The extension of online learning will be explored to overcome the distribution shift of live traffic streams. Federated IDS architectures, in which lightweight models are trained locally within distributed segments of the network and centrally, are an interesting deployment paradigm that fits the small scale of the proposed architecture.

Research involving human and/or animals

Not Applicable.

CRediT authorship contribution statement

Raja Waseem Anwar: Conceptualization, Formal analysis, Funding acquisition, Methodology, Validation, Writing – original draft, Writing – review & editing. **Mohammad Abrar:** Conceptualization, Formal analysis, Software, Writing – review & editing. **Abdu Salam:** Data curation, Software, Validation, Visualization. **Faizan Ullah:** Data curation, Formal analysis, Software, Validation, Visualization.

Declaration of competing interest

The authors declare no conflict of interest.

Acknowledgment

This work was supported by the German University of Technology (GUtech) – Muscat, Sultanate of Oman.

Data availability

The dataset proposed in this study is publicly available on Kaggle at: www.kaggle.com/code/muhammadabrar78/a-lightweight-deep-learning-model-for-malicious?scriptVersionId=311336309. The repository includes the complete dataset and supporting information to enable reproducibility and further research in cybersecurity.

References

[1] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. "A detailed analysis of the KDD CUP 99 data set," in 2009 IEEE symposium on computational intelligence for security and defense applications. IEEE 2009:1–6.
[2] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521(7553):436–44.
[3] Ring M, Wunderlich S, Scheuring D, Landes D, Hotho A. A survey of network-based intrusion detection data sets. Comput Secur 2019;86:147–67.
[4] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 2002;16:321–57.
[5] Siddiqui AJ, Boukerche A. Adaptive ensembles of autoencoders for unsupervised IoT network intrusion detection. Computing 2021;103(6):1209–32.
[6] LemaÃŽtre G, Nogueira F, Aridas CK. Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. J Mach Learn Res 2017;18 (17):1–5.
[7] Li W, Yi P, Wu Y, Pan L, Li J. A new intrusion detection system based on KNN classification algorithm in wireless sensor network. Journal of Electrical and Computer Engineering 2014;2014(1):240217.
[8] Mahdavifar S, Ghorbani AA. Application of deep learning to cybersecurity: a survey. Neurocomputing 2019;347:149–76.
[9] Tang TA, Mhamdi L, McLernon D, Zaidi SAR, Ghogho M. Deep recurrent neural network for intrusion detection in sdn-based networks. In: 2018 4th IEEE Conference on network softwarization and workshops (NetSoft). IEEE; 2018. p. 202–6.
[10] Wang Z. The applications of deep learning on traffic identification. BlackHat USA 2015;24(11):1–10.
[11] Abdulhammed R, Musafer H, Alessa A, Faezipour M, Abuzneid A. Features dimensionality reduction approaches for machine learning based network intrusion detection. Electronics 2019;8(3):322.
[12] Lopez-Martin M, Carro B, Sanchez-Esguevillas A, Lloret J. Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot. Sensors 2017;17(9):1967.
[13] Sharafaldin I, Lashkari AH, Hakak S, Ghorbani AA. Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In: 2019 international carnahan conference on security technology (ICCST). IEEE; 2019. p. 1–8.
[14] Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 military communications and information systems conference (MilCIS). IEEE; 2015. p. 1–6.
[15] Ahmad M, Riaz Q, Zeeshan M, Tahir H, Haider SA, Khan MS. Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using UNSW-NB15 data-set. EURASIP J Wirel Commun Netw 2021;2021(1):10.
[16] Elkan C. The foundations of cost-sensitive learning. Int Joint Conf Artif Intell 2001; 17(1):973–8. Lawrence Erlbaum Associates Ltd.
[17] Guyon I, Elisseeff A. An introduction to variable and feature selection. J Mach Learn Res 2003;3(Mar):1157–82.
[18] Anwar RW, Abrar M, Salam A, Ullah F. Federated learning with LSTM for intrusion detection in IoT-based wireless sensor networks: a multi-dataset analysis. PeerJ Comput Sci 2025;11:e2751.
[19] Anwer RW, Abrar M, Ullah M, Salam A, Ullah F. Advanced intrusion detection in the industrial internet of things using federated learning and LSTM models. Ad Hoc Netw 2025:103991.
[20] Anwer RW, Abrar M, Salam A, Ullah F. TEAD: trust-enhanced anomaly detection framework for intrusion detection in IoT-enabled wireless sensor networks (WSNs). Wirel Netw 2025;31(6):4179–97.
[21] Mainframe S. IDS 2018 Intrusion CSVs (CSE-CIC-IDS2018). https://www.kaggle.com/datasets/solarmainframe/ids-intrusion-csv.
[22] Moustafa N. UNSW-NB15 DataSet for network intrusion detection systems. https://research.unsw.edu.au/projects/unsw-nb15-dataset.
[23] Lin T-Y, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision; 2017. p. 2980–8.
[24] Alzughaibi S, El Khediri S. A cloud intrusion detection systems based on dnn using backpropagation and pso on the cse-cic-ids2018 dataset. Appl Sci 2023;13(4): 2276.
[25] Altunay HC, Albayrak Z. A hybrid CNN+ LSTM-based intrusion detection system for industrial IoT networks. Eng Sci Technol Int J 2023;38:101322.
[26] Ullah F, Ullah S, Srivastava G, Lin JC-W. IDS-INT: intrusion detection system using transformer-based transfer learning for imbalanced network traffic. Digital Communications and Networks 2024;10(1):190–204.
[27] Mercha EM, Chakir EM, Erradi M. Trans-IDS: a transformer-based intrusion detection system. In: SECRYPT; 2023. p. 402–9.
[28] Akpaku E, Chen J, Ahmed M, Agbenyegah FK, Brown-Acquaye WL. RAGN: detecting unknown malicious network traffic using a robust adaptive graph neural network. Comput Netw 2025;262:111184.
[29] Akpaku E, Chen J, Ofoeda J. TIPSO-GAN: malicious network traffic detection using a novel optimized generative adversarial network. In: Network and distributed System security (NDSS) symposium; 2026. https://doi.org/10.14722/ndss.2026.243241. 23-27. San Diego, CA, USA ISBN 979-8-9919276-8-0.