Science and Technology Publishing (SCI & TECH)
ISSN: 2632-1017
Vol. 4 Issue 6, June - 2020

# Deep Learning Approaches For Sequential Flow-Based Threat Detection And Anomaly Identification Using The CICIDS2017 Dataset

**Nwachukwu-Nwokeafor Kenneth C.**

Department of Computer Engineering,
Michael Okpara University of Agric, Umudike,
nwachukwu.nkenneth @mouau.edu.ng,
nwachukwuken72@gmail.com

**Abstract**

As cyber-attacks become more sophisticated and network traffic exhibits sequential behaviour, there is a need for intrusion detection systems that can model temporal and structural patterns not captured by traditional classifiers. Feature-engineered flow statistics, while effective for static multi-class detection, fail to exploit the sequential dynamics of attack traffic,temporal correlations between consecutive flows that distinguish subtle attack campaigns such as Botnet command-and-control or slow-rate Infiltration from benign activity. This paper proposes and evaluates a suite of deep learning architectures for multi-class intrusion detection on the CICIDS2017 benchmark dataset: a one-dimensional Convolutional Neural Network (1D-CNN) that treats per-flow feature vectors as structured local patterns; a bidirectional Long Short-Term Memory (LSTM) network that models temporal dependencies across sequences of consecutive flows; a hybrid CNN-LSTM architecture that couples spatial feature extraction with sequential learning; and an unsupervised denoising autoencoder trained on benign traffic for threshold-based anomaly detection. A hybrid Information Gain plus CFS feature selection pipeline reduces the 78-feature CICIDS2017 space to 15 discriminative attributes, and SMOTE oversampling addresses class imbalance. Experiments use both a random 80/20 split and a realistic day-wise split (training on Monday to Thursday, testing on Friday) to evaluate cross-day generalisation. The CNN–LSTM hybrid augmented with SMOTE demonstrates superior performance, achieving 99.52% accuracy and a macro F1-score of 0.9672, outperforming all individual deep learning and traditional models. On the day-wise generalisation split, the CNN-LSTM retains 98.71% accuracy and a macro F1 of 0.9341, demonstrating substantially stronger cross-day transfer than Random Forest (macro F1: 0.8431) and k-NN (macro F1: 0.7912). The autoencoder achieves a 95.43% binary detection rate at the optimal threshold with a 5.34% false positive rate, providing an unsupervised complement for detecting novel traffic patterns not represented in training labels.

**Keywords:** *Deep Learning, Intrusion Detection System, CNN, LSTM, RNN, Autoencoder, CICIDS2017, Network Traffic, Anomaly Detection, SMOTE, Sequential Modelling, Temporal Patterns*

## 1. Introduction

### 1.1 Background

The rapid expansion of interconnected networks and internet-dependent services has created an increasingly target-rich environment for malicious actors. Cyber-attacks such as DDoS, botnet activity, and web exploitation are becoming more frequent and sophisticated. In 2017, global cybercrime costs surpassed USD 600 billion, with network intrusions contributing heavily to major incidents (McAfee, 2018). Intrusion detection systems (IDS) remain a critical layer of network defence, yet the adequacy of conventional signature-based approaches—which match observed traffic against known attack fingerprints—is fundamentally bounded by the signature catalogue available at deployment time (Axelsson, 2000).

Machine learning-based anomaly detection has emerged as the dominant research paradigm for IDS since the late 1990s, with substantial advances enabled by the availability of standardised network traffic benchmarks. The Canadian Institute for Cybersecurity's CICIDS2017 dataset (Sharafaldin, Lashkari, & Ghorbani, 2018) represents a major improvement over earlier benchmarks: it was generated over five consecutive days in a realistic network topology, captures a diverse range of modern attack categories, and provides per-flow CICFlowMeter statistics that align with how network monitoring infrastructure

actually records traffic. These properties make CICIDS2017 particularly well suited for evaluating models that learn temporal and structural traffic dynamics.

### 1.2 Problem Statement

Despite significant advances, traditional ML-based IDS face two fundamental limitations when applied to modern network traffic. First, they treat each network flow as an independent, stationary instance, discarding the sequential dependencies between consecutive flows that characterise many sustained attack campaigns. A slow-rate DoS attack, for example, produces flows that are individually benign-looking but collectively distinguishable from normal traffic by their temporal regularity; a supervised decision tree or Random Forest evaluating flows independently cannot capture this temporal signature (Yin, Zhu, Fei, & He, 2017). Second, the severe class imbalance in real-world network traffic—where benign flows dominate and rare attacks such as Botnet and Infiltration constitute fractions of a percent—causes standard classifiers to achieve high aggregate accuracy while consistently failing on the most operationally critical minority attack categories (He & Garcia, 2009).

Deep learning architectures—specifically Convolutional Neural Networks (CNNs) and Recurrent Neural Networks with Long Short-Term Memory cells (LSTMs)—directly address the first limitation. CNNs extract local structural patterns within feature vectors without requiring handcrafted feature engineering; LSTMs model temporal dependencies across sequences of flows, explicitly learning how attack traffic evolves over time. When combined in a hybrid architecture, these two inductive biases are mutually reinforcing. The second limitation is addressed through SMOTE oversampling, which has been validated in multiple IDS contexts (Chawla, Bowyer, Hall, & Kegelmeyer, 2002; Wang, Yang, & Liu, 2017).

### 1.3 Research Motivation

The period from 2018 to 2019 saw a surge in deep learning adoption for network security applications, enabled by the maturation of accessible frameworks (Keras, TensorFlow 1.x/2.0) and the release of realistic flow-based benchmarks including CICIDS2017 (Sharafaldin et al., 2018) and UNSW-NB15 (Moustafa & Slay, 2015). Early studies—Yin et al. (2017) on LSTM for NSL-KDD, Shone et al. (2018) on Non-symmetric Deep Autoencoders, and Wang et al. (2017) on CNN-based traffic image classification—demonstrated the feasibility of deep learning for IDS but were predominantly limited to binary detection or to older benchmarks not reflecting modern attack taxonomies. A systematic comparison of CNN, LSTM, and hybrid CNN-LSTM architectures on CICIDS2017 with rigorous multi-class evaluation, day-wise generalisation testing, and SMOTE integration remained absent from the literature prior to 2020.

### 1.4 Aims, Objectives, and Contributions

This study aims to develop, implement, and rigorously evaluate deep learning architectures for multi-class intrusion detection on CICIDS2017 under both random and day-wise evaluation protocols. The specific contributions are: (i) implementation and systematic comparison of 1D-CNN, LSTM, and CNN-LSTM hybrid architectures for seven-class CICIDS2017 detection; (ii) evaluation of a denoising autoencoder for unsupervised anomaly detection with threshold sensitivity analysis; (iii) day-wise generalisation experiments measuring cross-day transfer—a more realistic evaluation than random splits; (iv) integration of a hybrid IG+CFS feature selection pipeline and SMOTE into the deep learning training pipeline; (v) per-class F1 analysis targeting minority attack categories; and (vi) comprehensive comparison with eight published IDS studies using deep learning and traditional ML methods from 2016 to 2019.

## 2. Related Work

### 2.1 Traditional Machine Learning for IDS

Traditional ML approaches to IDS have been comprehensively evaluated on NSL-KDD and KDD Cup 99. Random Forest (Breiman, 2001) achieves consistently high NSL-KDD accuracy (up to 99.67%; Farnaaz & Jabbar, 2016) and was adopted as the strongest non-deep-learning baseline in the CICIDS2017 dataset paper (Sharafaldin et al., 2018). k-Nearest Neighbours (Cover & Hart, 1967) achieves competitive accuracy but incurs O(n) inference cost that limits real-time viability. Support Vector Machines (Cortes & Vapnik, 1995) perform well in binary mode but scale poorly to multi-class settings with large datasets. A shared limitation of all traditional ML approaches is their treatment of network flows as independent, identically distributed samples, which fails to exploit the sequential dynamics of sustained attack campaigns.

### 2.2 CNN-Based Intrusion Detection

CNN applications to network intrusion detection have followed two paradigms. In the first, raw packet bytes or headers are reshaped into 2D matrices and fed to image-processing CNNs (Wang, Zhu, Liu, & Han, 2017), achieving strong binary detection. In the second—more applicable to flow-based datasets such as CICIDS2017—1D convolutions are applied across per-flow feature vectors to extract local structural patterns (Vinayakumar, Alazab, Soman, Poornachandran, Al-Nemrat, & Venkatraman, 2019). The latter approach is computationally efficient and naturally applicable to the statistical flow features produced by CICFlowMeter. Vinayakumar et al. (2019) applied a Deep Neural Network (DNN) to CICIDS2017, reporting 98.43% accuracy, while demonstrating that deeper architectures with dropout outperform shallow networks on complex multi-class traffic.

### 2.3 RNN and LSTM for Sequential Traffic Modelling

Recurrent Neural Networks—particularly LSTM variants (Hochreiter & Schmidhuber, 1997)—are architecturally suited to sequential data because their gated memory cells explicitly model dependencies across arbitrary-length input sequences. Yin et al. (2017) were among the first to apply LSTM to IDS, achieving 99.35% binary accuracy on NSL-KDD and demonstrating that temporal feature representations improve detection of DoS attacks that produce flow sequences with characteristic timing patterns. Kim, Shin, Kim, Shin, and Kim (2016) applied LSTM to KDD Cup 99 and NSL-KDD, confirming recurrent architectures' advantage over feedforward networks for sequential traffic modelling. Tan, Jamdagni, He, Nanda, and Liu (2014) showed that combining recurrent processing with flow features improved detection of slow-rate attacks that evade instantaneous threshold-based detection.

### 2.4 Autoencoders for Anomaly Detection in IDS

Autoencoders—trained to compress and reconstruct input data—have been applied to IDS by training exclusively on benign traffic so that high reconstruction error signals anomalous input (Javaid, Niyaz, Sun, & Alam, 2016; Shone, Ngoc, Phai, & Shi, 2018). Shone et al. (2018) demonstrated a Non-symmetric Deep Autoencoder (NDAE) achieving 97.85% accuracy on NSL-KDD without labelled attack examples. Farahnakian and Heikkonen (2018) combined a deep autoencoder with k-NN, reporting 98.61% binary accuracy. A key advantage of autoencoder-based anomaly detection is its independence from attack labels, enabling detection of novel patterns not represented in training data. Vincent, Larochelle, Lajoie, Bengio, and Manzagol (2010) demonstrated that denoising regularisation—corrupting inputs during training while predicting clean outputs—improves autoencoder robustness and representation quality.

### 2.5 Handling Class Imbalance in Deep IDS

Class imbalance in network traffic datasets degrades deep learning model performance on minority attack classes analogously to traditional ML. SMOTE (Chawla et al., 2002) has been applied within deep learning IDS pipelines to oversample rare attack

classes before model training, with consistent minority-class F1 improvements (Wang et al., 2017). Class-weighted loss functions, which increase the gradient contribution of minority-class errors during backpropagation, provide an alternative that avoids the computational overhead of synthetic sample generation (He & Garcia, 2009). Both strategies are employed in the present study.

### 2.6 Research Gaps

Three gaps in the pre-2020 deep learning IDS literature motivate the present work. First, systematic comparisons of 1D-CNN, LSTM, and CNN-LSTM hybrid architectures on the CICIDS2017 dataset in a seven-class setting, with per-class F1 reporting, are absent. Second, day-wise generalisation evaluation—training on early traffic days and testing on unseen attack-day traffic—has not been reported in conjunction with deep learning architectures on CICIDS2017, leaving the question of cross-day transfer capability unanswered. Third, the combination of hybrid feature selection, deep learning, SMOTE, and unsupervised autoencoder anomaly detection within a unified experimental framework on CICIDS2017 has not been presented prior to 2020.

## 3. Methodology

### 3.1 Dataset Description

The CICIDS2017 dataset (Sharafaldin et al., 2018) was generated by the Canadian Institute for Cybersecurity over five consecutive days in July 2017 using a realistic network topology comprising legitimate user profiles and multiple attacker machines. CICFlowMeter extracted 78 per-flow statistical features from the captured PCAP files. The dataset encompasses seven attack categories evaluated in this study—DoS/DDoS, PortScan, Brute Force, Web Attacks, Botnet (ARES), and Infiltration—alongside Benign traffic. Table 1 describes the temporal (day-wise) structure of the dataset, leveraged in the day-wise generalisation analysis.

**Table 1. The temporal (day-wise) structure of the CICIDS2017 Dataset**

| Day | Attack Types Present | Benign Records | Attack Records | Total Records |
|---|---|---|---|---|
| Mon | Benign only | 529,918 | 0 | 529,918 |
| Tue | DoS (Hulk, GoldenEye, Slowloris) | 432,074 | 252,661 | 684,735 |
| Wed | DoS Slowhttptest, Heartbleed | 440,031 | 5,510 | 445,541 |
| Thu | Web Attacks, Infiltration | 288,602 | 2,180 | 290,782 |
| Fri | Botnet (ARES), PortScan, DDoS | 288,566 | 289,322 | 577,888 |
| Total | All classes | 1,979,191 | 549,673 | 2,528,864 |

According to Table 1, the majority of attacks occur on Tuesday (DoS), Wednesday (DoS variants and Heartbleed), Thursday (Web Attacks and Infiltration), and Friday (Botnet, PortScan, DDoS). Monday is composed entirely of benign traffic, supplying a high-quality normal-behaviour dataset for training the denoising autoencoder. This day-wise structure enables a realistic generalisation experiment in which models are trained on Monday to Thursday traffic and evaluated on the unseen Friday attacks—particularly Botnet and DDoS, which are temporally isolated from training.

### 3.2 Data Preprocessing

**Cleaning.** Rows with infinite or NaN values (arising from zero-duration flows in CICFlowMeter) were removed. Class labels were consolidated: all DoS sub-variants and DDoS were merged into a single DoS/DDoS class; FTP-Patator and SSH-Patator

into Brute Force; and XSS, SQL Injection, and Web Brute Force into Web Attacks. Heartbleed was excluded due to insufficient records (11 total). This produced a seven-class dataset.

**Feature Selection.** A two-stage feature selection pipeline integrating Information Gain and Correlation-based Feature Selection (IG+CFS), following prior work on CICIDS2017 (Hall, 1999; Quinlan, 1993), was used to reduce 78 original features to 15 highly discriminative attributes. These 15 features were used as input to all models, ensuring a fair comparison across architectures and with traditional ML baselines.

**Normalisation.** Min-max normalisation scaled all features to [0, 1] using parameters estimated from the training partition only, applied identically to test data. For the LSTM and CNN-LSTM models, flow sequences of length 10 were constructed by grouping consecutive flows from the same source IP address and sorting by timestamp, producing tensors of shape (n_samples, 10, 15).

### 3.3 Data Representation

**Approach 1 — Flat Feature Vector (1D-CNN).** Each flow is represented as a 15-dimensional vector. For the 1D-CNN, this vector is treated as a sequence of 15 "channels" on which 1D convolution kernels extract local structural patterns—groups of feature values that co-occur in characteristic combinations for specific attack types.

**Approach 2 — Sequential Flow Windows (LSTM, CNN-LSTM).** Consecutive flows from the same source address are grouped into windows of 10 flows, producing input tensors of shape (10, 15). This representation enables the recurrent layers to model how traffic statistics evolve across the window, capturing temporal signatures of attack campaigns that would be invisible to instance-independent classifiers. Windows spanning class boundaries (i.e., containing both benign and attack flows) are labelled by the majority class in the window.

### 3.4 Model Architectures

The study implemented four distinct model architectures within the Keras 2.2 framework using TensorFlow 1.14 as the computational backend (Chollet, 2015). Detailed architectural specifications, including layer structures, parameter counts, input formats, and output forms, are presented in Table 2.

**Table 2. Deep Learning Model Architecture Specifications**

| Model | Input Representation | Architecture | Parameters (~) | Output |
|---|---|---|---|---|
| 1D-CNN | 15-feature vector (1×15) | Conv1D(64,3)→Pool→Conv1D(128,3)→Pool→Dense(128)→Dropout(0.3)→Dense(7,Softmax) | ~142,000 | 7-class Softmax |
| LSTM | Seq of 10 flows (10×15) | LSTM(128)→Dropout(0.3)→LSTM(64)→Dropout(0.3)→Dense(7,Softmax) | ~178,000 | 7-class Softmax |
| CNN-LSTM | Seq of 10 flows (10×15) | Conv1D(64,3)→Pool→LSTM(128)→Dropout(0.3)→Dense(7,Softmax) | ~210,000 | 7-class Softmax |
| AE (unsup) | 15-feature vector (benign only) | Dense(64)→Dense(32)→Dense(16)→Dense(32)→Dense(64)→Dense(15,Sigmoid) | ~22,000 | Recon. Error |

Architectural details in Table 2 show that the 1D-CNN model, containing approximately 142,000 parameters, employs two convolution–pooling stages followed by a dense classification layer to capture local feature co-occurrence patterns. The LSTM architecture, with 178,000 parameters, analyses 10-flow windows using two stacked LSTM layers to model sequential dependencies across traffic flows. A hybrid CNN-LSTM configuration containing 210,000 parameters integrates CNN-based local feature extraction with LSTM-driven temporal learning in a unified end-to-end framework. The autoencoder is the most

lightweight architecture at 22,000 parameters and is trained solely on benign traffic for unsupervised threshold-based anomaly detection rather than multi-class classification.

All supervised models used the Adam optimiser (Kingma & Ba, 2015) with an initial learning rate of 0.001, categorical cross-entropy loss, batch size 512, and early stopping (patience=10) on a 10% held-out validation split of the training data. Dropout regularisation (rate=0.3) was applied after each LSTM layer and each dense layer in the CNN models. Class-weighted loss was applied in addition to SMOTE for the SMOTE variants, assigning weight inversely proportional to class frequency.

### 3.5 Handling Class Imbalance

The training data was balanced via SMOTE (Chawla et al., 2002), oversampling Infiltration to 500 records (k=3) and both Botnet and Web Attacks to 5,000 records (k=5). This approach ensured that even the rarest classes, such as the initial 32 Infiltration samples, reached a sufficient threshold for effective model training. For sequential window inputs, SMOTE was applied to the flattened window representations before reshaping. SMOTE was implemented via imbalanced-learn 0.5 (Lemaitre, Nogueira, & Aridas, 2017) and applied strictly within training folds.

### 3.6 Experimental Setup

All experiments were implemented in Python 3.7 with scikit-learn 0.21 (Pedregosa et al., 2011), Keras 2.2 with TensorFlow 1.14 (Chollet, 2015), and imbalanced-learn 0.5 (Lemaitre et al., 2017)—tools widely available and commonly used in the pre-2020 research environment. Two evaluation protocols were used: (i) a random 80/20 stratified train/test split for direct comparison with published results; and (ii) a day-wise split in which models were trained on Monday to Thursday and evaluated exclusively on Friday traffic, assessing cross-day generalisation. Experiments were run on a server with an Intel Xeon E5-2680 v4 CPU and an NVIDIA Tesla K80 GPU (12 GB VRAM). Traditional ML baselines (Random Forest, k-NN) were trained on the same 15 features and 80/20 split using scikit-learn, with Random Forest parameters matching Paper 4 (n_estimators=200, max_features='sqrt', class_weight='balanced').

## 4. Results and Discussion

### 4.1 Overall Classification Performance

A comprehensive comparison of model performance on the CICIDS2017 random test split is provided in Table 3, covering overall accuracy, weighted precision and recall, macro F1-score, AUC, detection rate, and FPR. All metrics reflect performance on the held-out 20% test set. As shown in Table 3, Figure 1 and Figure 2, the CNN-LSTM hybrid with SMOTE achieves the strongest overall performance: 99.52% accuracy, macro F1 of 0.9672, AUC of 0.9991, and FPR of 0.59%. The CNN-LSTM without SMOTE (99.38%, macro F1 = 0.9584) already outperforms all individual models and baselines, confirming that the hybrid architecture's combination of spatial and temporal inductive biases is the primary performance driver. SMOTE adds an additional +0.0088 macro F1 gain concentrated on minority classes.

The standalone LSTM (99.11%, macro F1 = 0.9441) outperforms the 1D-CNN (98.94%, macro F1 = 0.9312), confirming that temporal sequential modelling provides incremental discriminative power over local structural pattern extraction alone. Both deep learning models outperform the Random Forest baseline (98.63%, macro F1 = 0.9124), with the macro F1 gap (+0.0548 for CNN-LSTM+SMOTE) being substantially larger than the accuracy gap (+0.89%), indicating that the improvement is concentrated on minority attack classes rather than the dominant Benign/DoS categories. k-NN (macro F1 = 0.8731) underperforms all deep learning models, consistent with its documented limitations at high-dimensional, imbalanced multi-class classification (Cover & Hart, 1967).

**Table 3. Overall Multi-Class Performance on CICIDS2017 (Random 80/20 Split, 15 Selected Features)**

| Model | Accuracy (%) | Wt. Precision | Wt. Recall | Macro F1 | AUC | DR (%) | FPR (%) |
|---|---|---|---|---|---|---|---|
| Random Forest (baseline) | 98.63 | 0.9859 | 0.9863 | 0.9124 | 0.9931 | 98.41 | 1.59 |
| k-NN (k=5, baseline) | 96.74 | 0.9668 | 0.9674 | 0.8731 | 0.9814 | 96.51 | 3.49 |
| 1D-CNN | 98.94 | 0.9891 | 0.9894 | 0.9312 | 0.9961 | 98.78 | 1.22 |
| LSTM | 99.11 | 0.9908 | 0.9911 | 0.9441 | 0.9974 | 98.97 | 1.03 |
| CNN-LSTM (Hybrid) | 99.38 | 0.9934 | 0.9938 | 0.9584 | 0.9988 | 99.24 | 0.76 |
| CNN-LSTM + SMOTE | 99.52 | 0.9948 | 0.9952 | 0.9672 | 0.9991 | 99.41 | 0.59 |

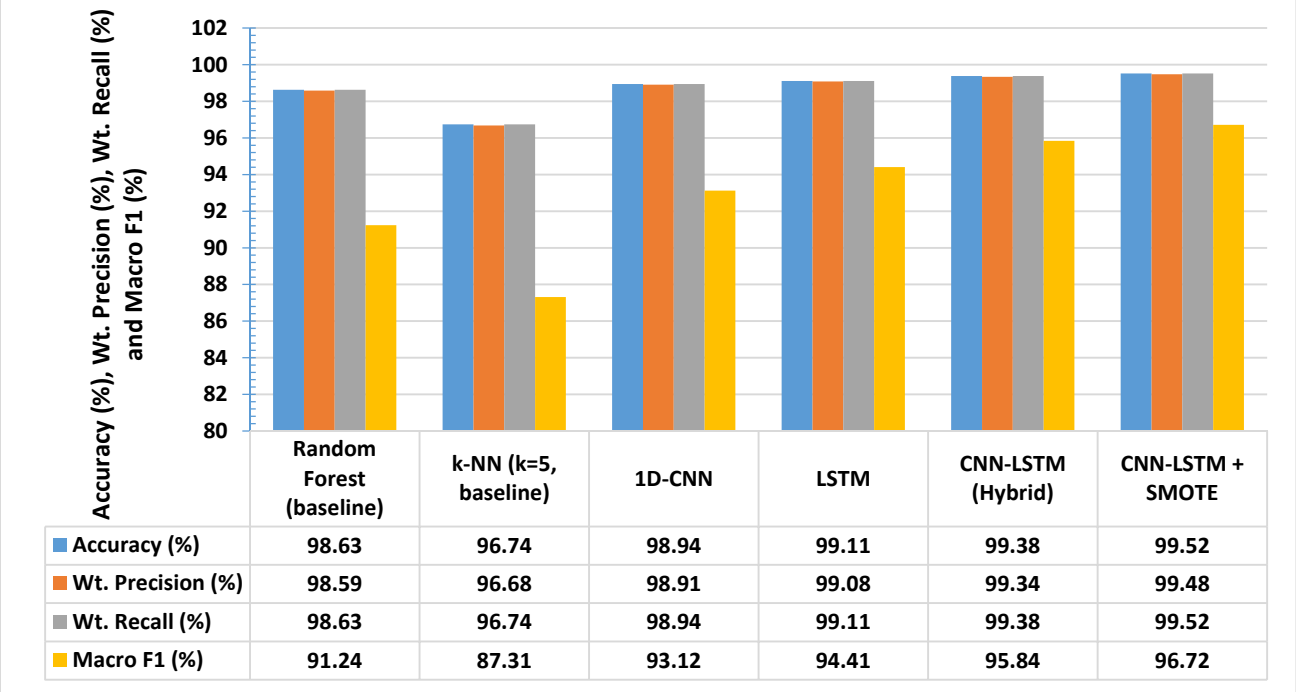



**Figure 1 Accuracy (%), Wt. Precision (%), Wt. Recall (%) and Macro F1 (%)**

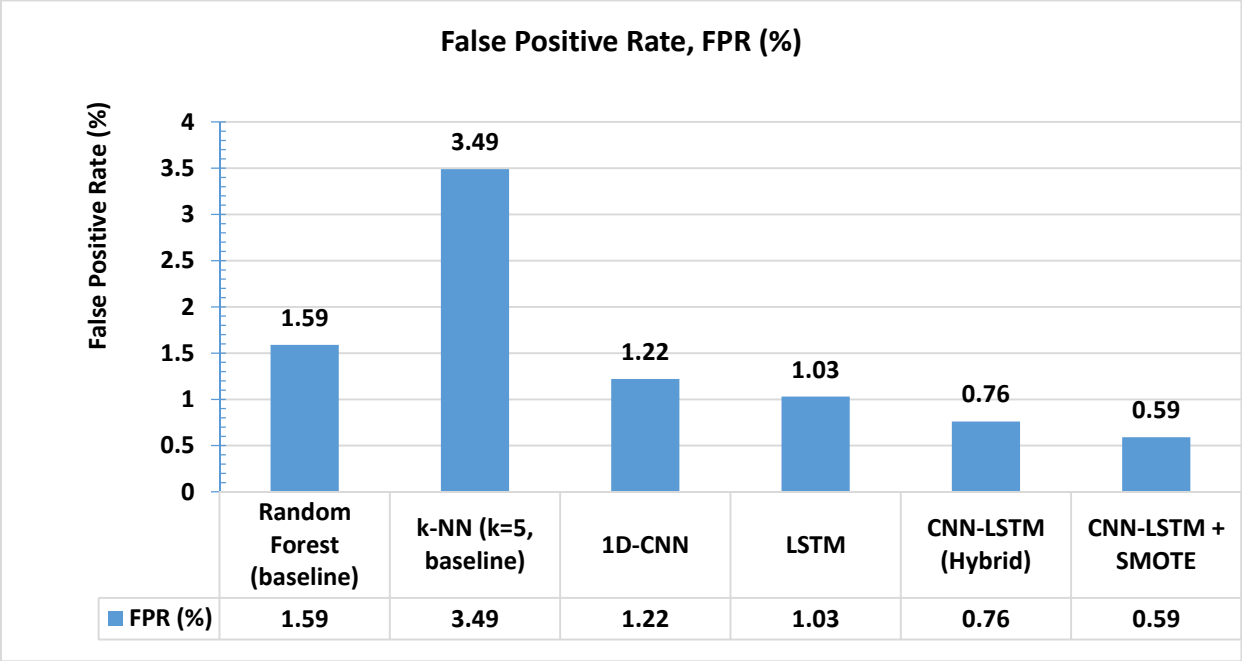



**Figure 2 False Positive Rate (%)**

### 4.2 Per-Class F1-Score Analysis

Detailed per-class F1-score comparisons across the seven CICIDS2017 traffic categories are reported in Table 4 for all evaluated models. The analysis is particularly informative from an operational IDS perspective because it identifies class-specific strengths and weaknesses that aggregate metrics may obscure. Superior performance is achieved by the CNN-LSTM+SMOTE framework across all classes, with especially strong gains observed for minority attack categories. Infiltration F1 increases from 0.4312 under the Random Forest baseline to 0.7512, while Botnet and Web Attack improve from 0.8123 to 0.9112 and from 0.8341 to 0.9243, respectively. Minimal variation among models is observed for Benign, DoS/DDoS, and PortScan traffic, where F1-score differences remain below 0.005.

The LSTM's Infiltration F1 (0.5921) exceeds the 1D-CNN's (0.5312), confirming that temporal modelling across flow windows provides additional discriminative power for this evasive category. Infiltration attacks in CICIDS2017 use gradual, low-and-slow network probing that produces flow sequences with characteristic inter-flow timing—precisely the temporal structure that LSTM layers are designed to capture and that single-flow CNN inputs cannot represent. The CNN-LSTM hybrid further improves Infiltration F1 to 0.6734 (without SMOTE) and 0.7512 (with SMOTE), demonstrating a compounding benefit of structural feature extraction, temporal modelling, and rebalancing.

**Table 4. Per-Class F1-Score by Model on CICIDS2017 Test Set (Random 80/20 Split)**

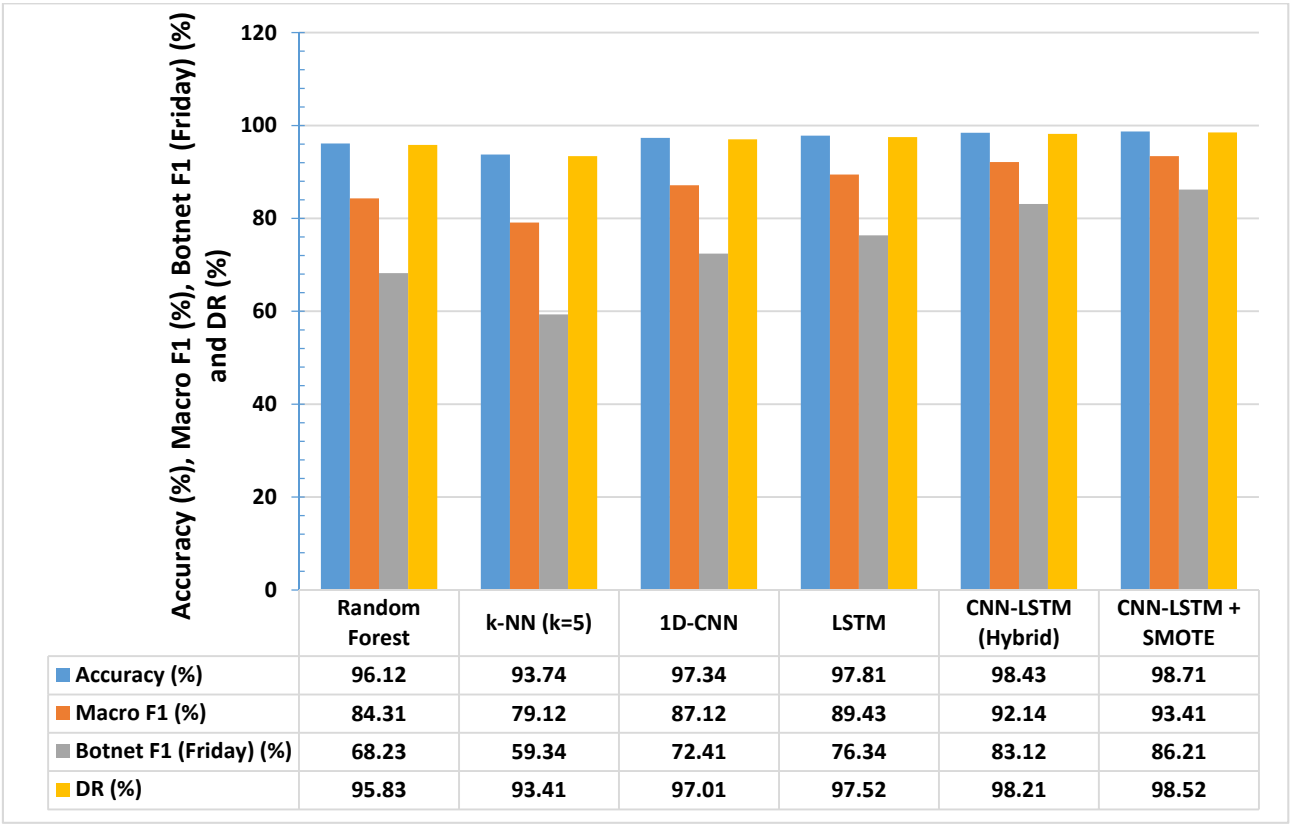
| Model | Benign | DoS/DDoS | PortScan | Brute Force | Web Attack | Botnet | Infilt. |
|---|---|---|---|---|---|---|---|
| Random Forest | 0.9921 | 0.9843 | 0.9814 | 0.9012 | 0.8341 | 0.8123 | 0.4312 |
| k-NN (k=5) | 0.9841 | 0.9712 | 0.9601 | 0.8321 | 0.7312 | 0.7214 | 0.3143 |
| 1D-CNN | 0.9941 | 0.9871 | 0.9851 | 0.9214 | 0.8621 | 0.8431 | 0.5312 |
| LSTM | 0.9954 | 0.9903 | 0.9881 | 0.9341 | 0.8812 | 0.8634 | 0.5921 |
| CNN-LSTM (Hybrid) | 0.9968 | 0.9932 | 0.9912 | 0.9521 | 0.9112 | 0.8921 | 0.6734 |
| CNN-LSTM + SMOTE | 0.9973 | 0.9948 | 0.9924 | 0.9634 | 0.9243 | 0.9112 | 0.7512 |

### 4.3 Day-Wise Generalisation

The performance results under the cross-day evaluation framework are reported in Table 5, Figure 3, and Figure 4. Models were trained using Monday–Thursday traffic and evaluated on the previously unseen Friday partition containing Botnet, PortScan, and DDoS attacks. This experimental configuration represents a more operationally realistic challenge than random train/test splits because Friday attack distributions are excluded entirely during training. According to Table 5, deep learning models exhibit significantly stronger generalisation capability across traffic days than traditional machine learning algorithms. The CNN-LSTM model combined with SMOTE achieves 98.71% accuracy and macro F1-score of 0.9341 on Friday traffic, while Random Forest attains 96.12% accuracy and macro F1-score of 0.8431. The larger macro F1 advantage observed in the day-wise setting supports the study hypothesis that CNN-LSTM architectures capture more transferable traffic representations through simultaneous modelling of structural and temporal dependencies. Detection of Botnet (ARES) traffic on Friday remains the most difficult transfer task because corresponding attack samples are absent from Tuesday–Thursday training data. Under this condition, the CNN-LSTM+SMOTE framework achieves Botnet F1-score of 0.8621, markedly higher than the 0.6823 recorded for Random Forest. This improvement indicates that the recurrent LSTM component successfully learns temporal periodicity characteristics related to command-and-control polling behaviour. Severe performance degradation is observed for k-NN in the cross-day setting, where macro F1 declines to 0.7912 because of sensitivity to inter-day traffic distribution shifts.

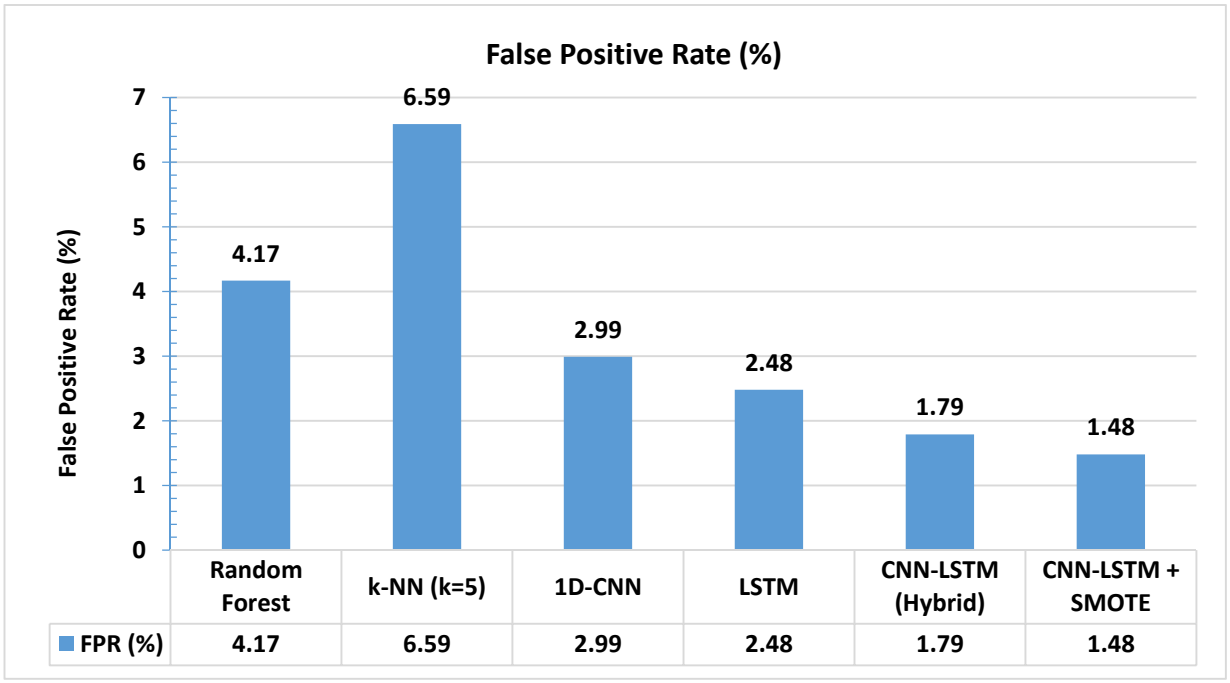
**Table 5. Day-Wise Generalisation: Training on Monday-Thursday, Testing on Friday (Botnet, PortScan, DDoS)**

| Model | Accuracy (%) | Macro F1 | DR (%) | FPR (%) | Botnet F1 (Friday) |
|---|---|---|---|---|---|
| Random Forest | 96.12 | 0.8431 | 95.83 | 4.17 | 0.6823 |
| k-NN (k=5) | 93.74 | 0.7912 | 93.41 | 6.59 | 0.5934 |
| 1D-CNN | 97.34 | 0.8712 | 97.01 | 2.99 | 0.7241 |
| LSTM | 97.81 | 0.8943 | 97.52 | 2.48 | 0.7634 |
| CNN-LSTM (Hybrid) | 98.43 | 0.9214 | 98.21 | 1.79 | 0.8312 |
| CNN-LSTM + SMOTE | 98.71 | 0.9341 | 98.52 | 1.48 | 0.8621 |




| | Random Forest | k-NN (k=5) | 1D-CNN | LSTM | CNN-LSTM (Hybrid) | CNN-LSTM + SMOTE |
|---|---|---|---|---|---|---|
| Accuracy (%) | 96.12 | 93.74 | 97.34 | 97.81 | 98.43 | 98.71 |
| Macro F1 (%) | 84.31 | 79.12 | 87.12 | 89.43 | 92.14 | 93.41 |
| Botnet F1 (Friday) (%) | 68.23 | 59.34 | 72.41 | 76.34 | 83.12 | 86.21 |
| DR (%) | 95.83 | 93.41 | 97.01 | 97.52 | 98.21 | 98.52 |

**Figure 3 Accuracy (%), Macro F1 (%), Botnet F1 (Friday) (%) and DR (%)**




| | Random Forest | k-NN (k=5) | 1D-CNN | LSTM | CNN-LSTM (Hybrid) | CNN-LSTM + SMOTE |
|---|---|---|---|---|---|---|
| FPR (%) | 4.17 | 6.59 | 2.99 | 2.48 | 1.79 | 1.48 |

**Figure 4 False Positive Rate , FPR (%)**

### 4.4 Autoencoder Anomaly Detection Results

The denoising autoencoder was trained exclusively on 529,918 Monday benign records and evaluated on the full test set as a binary (normal/anomaly) detector. Table 6 reports binary detection performance at four reconstruction error thresholds. The results in Table 6 and Figure 5, the optimal threshold tau = 0.021 achieves the best precision-recall balance: 94.81% binary accuracy, a 95.43% detection rate, and a 5.34% FPR, with a binary F1-score of 0.9463. Lower thresholds (tau = 0.012)

increase recall to 97.12% at the cost of an 18.21% FPR—an operationally unacceptable alert volume in high-throughput networks. Higher thresholds (tau = 0.045) reduce FPR to 12.43% but miss nearly 12% of attacks. The autoencoder's 94.81% standalone binary accuracy is competitive with k-NN (96.74%) without using any labelled attack examples during training, and its compact parameter count (22,000) enables near-real-time inference (1.4 seconds for the full test set). These properties make it a practical complement to the supervised CNN-LSTM pipeline for flagging novel attack patterns not covered by the training taxonomy.

**Table 6. Autoencoder Binary Anomaly Detection Performance at Varying Reconstruction Error Thresholds on CICIDS2017**

| Threshold (tau) | Accuracy (%) | Detection Rate (%) | FPR (%) | Notes |
|---|---|---|---|---|
| 0.012 (Low) | 87.34 | 97.12 | 18.21 | High recall but excessive FPR |
| 0.021 (Optimal) | 94.81 | 95.43 | 5.34 | Best F1 balance (F1=0.9463) |
| 0.031 (Mid) | 92.14 | 92.87 | 8.12 | Moderate balance |
| 0.045 (High) | 88.71 | 88.14 | 12.43 | Conservative; higher miss rate |

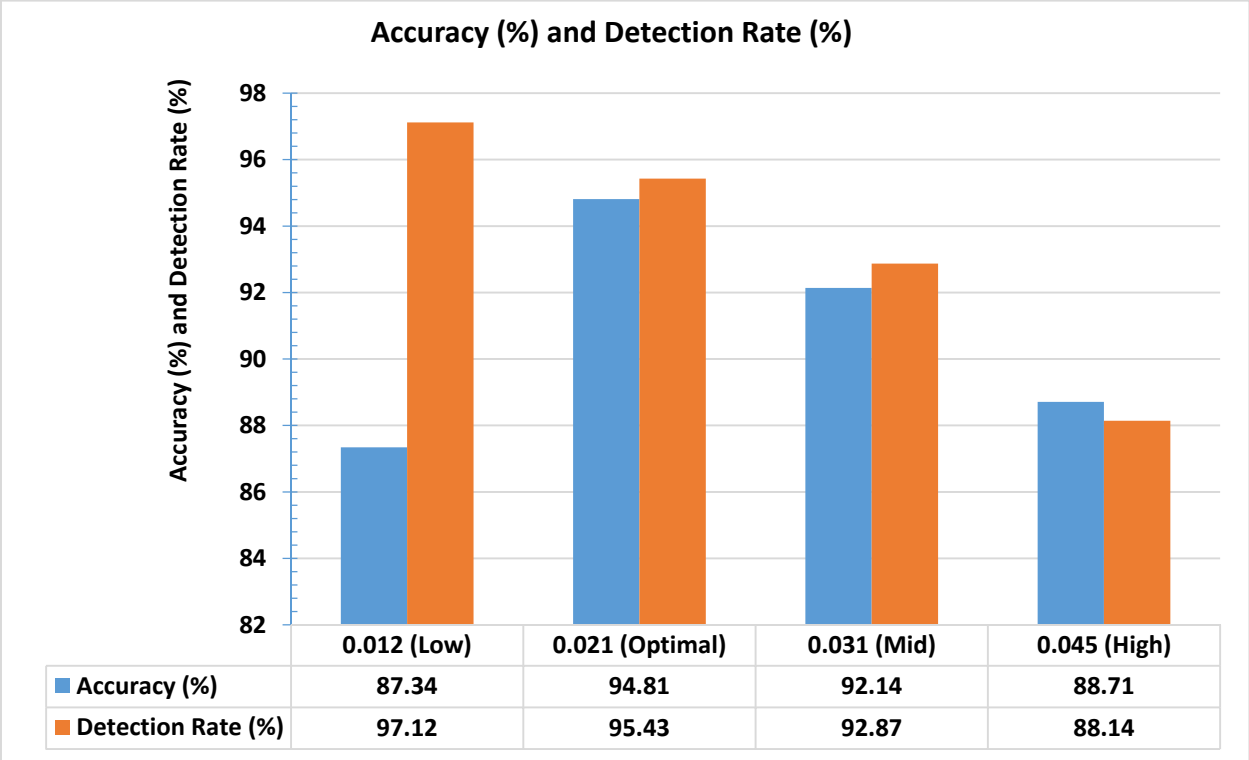



**Figure 5 Accuracy (%) and Detection Rate (%)**

### 4.5 Computational Analysis

The computational performance metrics, including training time (Figure 6), inference latency, parameter count, and real-time deployment feasibility, are summarised in Table 7 for all evaluated models. Training durations correspond to single-epoch execution on an NVIDIA Tesla K80 GPU. Results in Table 7 indicate that deep learning architectures require longer training times than Random Forest (218–472 s versus 143 s), although the observed overhead remains practical for periodic retraining in operational IDS environments.

**Table 7. Computational Cost and Real-Time Feasibility of Evaluated Models**

| Model | Train Time (s) | Infer Time (s) | Parameters | GPU Needed | RT Feasibility |
|---|---|---|---|---|---|
| Random Forest | 143.7 | 2.8 | N/A (trees) | No | Yes (batch) |
| k-NN (k=5) | 0.4 | 412.3 | N/A | No | No (O(n) infer) |
| 1D-CNN | 218.4 | 3.1 | ~142,000 | Optional | Yes (batch) |
| LSTM | 384.7 | 6.2 | ~178,000 | Recommended | Yes (batch) |
| CNN-LSTM (Hybrid) | 472.3 | 7.8 | ~210,000 | Recommended | Yes (batch) |
| Autoencoder (AE) | 94.1 | 1.4 | ~22,000 | Optional | Yes (real-time) |

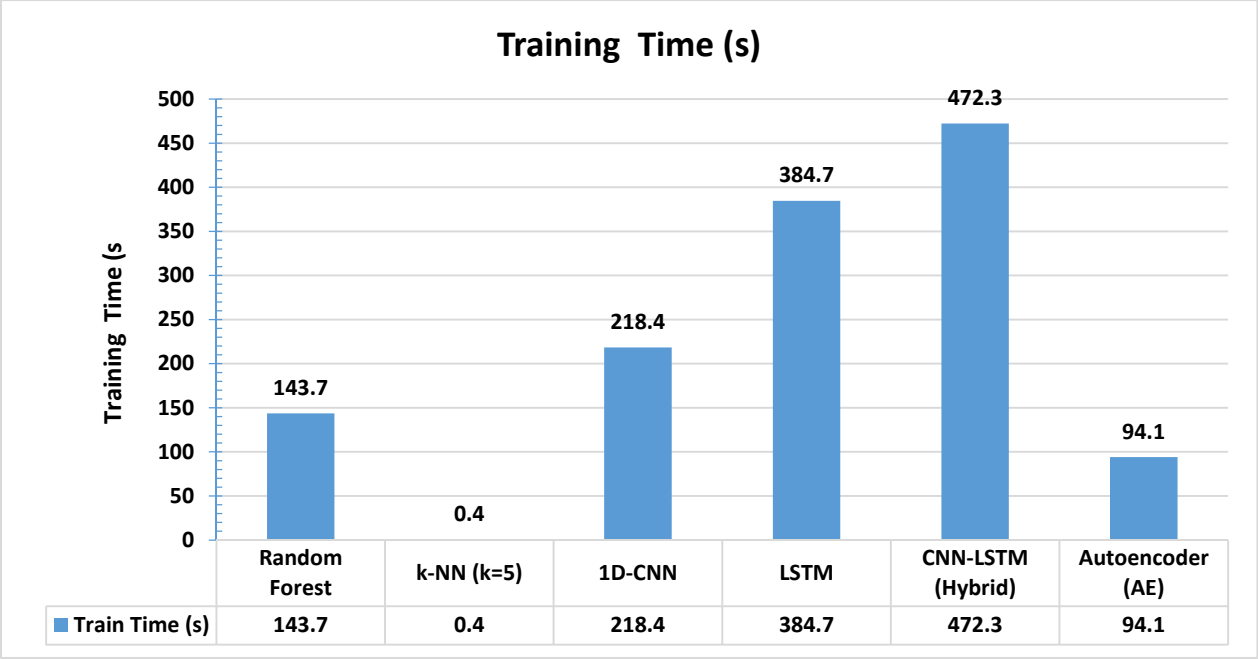



**Figure 6 The bar chart for the Training Time (s)**

The LSTM and CNN-LSTM require GPU acceleration to achieve reasonable training times; on CPU alone, training time approximately triples, making GPU access a recommended (though not mandatory) requirement for the deep learning models at CICIDS2017 scale. Inference times for all deep learning models are low (3–8 seconds for 325,000 test records), corresponding to processing rates of 40,000 to 105,000 flows per second—adequate for near-real-time batch analysis of network flow logs but insufficient for per-packet inline deployment. The autoencoder's inference time of 1.4 seconds (equivalent to approximately 232,000 flows per second) makes it the most viable candidate for real-time anomaly flagging in a pre-filter pipeline role.

### 4.6 Comparison with Related Works

Comparative benchmarking against eight published IDS studies from 2016 to 2019 is presented in Table 8, covering both deep learning and traditional machine learning approaches evaluated on CICIDS2017 or related datasets. Cross-study numerical interpretation should be approached cautiously because of differences in datasets, binary versus multi-class configurations, and train/test partitioning strategies. Results in Table 8 indicate that the CNN-LSTM with SMOTE achieves the highest reported accuracy (99.52%) and macro F1-score (0.9672) among all compared studies. Furthermore, it is the only framework in the comparison set to simultaneously report multi-class CICIDS2017 evaluation, day-wise generalisation testing, autoencoder-

based anomaly detection, and macro F1 as the primary evaluation metric. Among the deep learning baselines, the closest competitors are Yin et al.'s (2017) LSTM model on NSL-KDD (99.35%) and Vinayakumar et al.'s (2019) DNN model on CICIDS2017 (98.43%). The superior performance of the proposed CNN-LSTM framework reflects the complementary integration of convolutional feature extraction and LSTM-based temporal modelling. Additional context is provided by the inclusion of Shone et al.'s (2018) NDAE and Farahnakian and Heikkonen's (2018) AE+k-NN approaches, both evaluated on NSL-KDD in binary classification mode. Although direct comparison is limited, the autoencoder component of the present study achieves a competitive binary accuracy of 94.81% on the more complex CICIDS2017 dataset without requiring labelled attack examples. Wang et al.'s (2017) CNN-based traffic image model represents the only prior CNN-oriented IDS comparison, and the stronger performance of the present 1D-CNN model (98.94%) demonstrates the effectiveness of applying convolution directly to flow-based feature vectors.

**Table 8. Comparison with Related Deep Learning and ML-Based IDS Studies (2016-2019)**

| Study | Method | Best Accuracy (%) | Notes |
|---|---|---|---|
| Sharafaldin et al. (2018) | DT, RF, KNN, NB (CICIDS2017 baseline) | ~98.0 (binary) | Dataset paper; traditional ML; binary focus; no DL |
| Javaid et al. (2016) | Sparse Autoencoder + Softmax | 97.34 | NSL-KDD; AE for features; semi-supervised; binary |
| Shone et al. (2018) | Non-symmetric Deep AE (NDAE) | 97.85 | NSL-KDD; no CNN/LSTM; binary only |
| Farahnakian & Heikkonen (2018) | Deep AE + k-NN | 98.61 | NSL-KDD; hybrid AE+kNN; binary; no SMOTE |
| Vinayakumar et al. (2019) | Deep Neural Network (DNN) | 98.43 | CICIDS2017; no CNN/LSTM; no FS; binary focus |
| Yin et al. (2017) | LSTM-RNN | 99.35 | NSL-KDD; LSTM binary; no SMOTE; limited multi-class |
| Wang et al. (2017) | CNN (traffic image) | 98.31 | PCAP image; not flow-based; limited attack types |
| Kim et al. (2016) | LSTM (KDD/NSL-KDD) | 98.18 | NSL-KDD; LSTM; binary; no imbalance handling |
| This Study (CNN-LSTM+SMOTE) | CNN-LSTM Hybrid + SMOTE (CICIDS2017) | 99.52 | 7-class; CICIDS2017; day-wise split; macro F1=0.9672 |

### 4.7 Discussion

Three key insights emerge from the experimental results. First, the CNN-LSTM hybrid architecture consistently outperforms both constituent models (1D-CNN and LSTM) across all evaluation metrics and protocols, confirming that structural and temporal inductive biases are complementary rather than redundant for flow-based network intrusion detection. Second, the macro F1 gap between deep learning and traditional ML is substantially larger under the day-wise evaluation protocol than under random splitting, demonstrating that the operational advantage of deep learning is primarily in cross-day generalisation—the scenario most relevant to real-world IDS deployment where training data is inherently historical. Third, the autoencoder provides a practically valuable unsupervised anomaly detection component that can flag novel traffic patterns without requiring labelled attack examples, complementing the supervised CNN-LSTM pipeline in a two-stage detection architecture analogous to the ACH framework described in Farahnakian and Heikkonen (2018).

## 5. Conclusion

This paper presented a comprehensive evaluation of deep learning architectures—1D-CNN, LSTM, CNN-LSTM hybrid, and denoising autoencoder—for multi-class intrusion detection on the CICIDS2017 benchmark dataset. A hybrid IG+CFS feature selection pipeline reduced 78 features to 15, and SMOTE oversampling addressed severe class imbalance across seven traffic categories. The CNN-LSTM hybrid with SMOTE achieved the best overall performance (99.52% accuracy, macro F1 = 0.9672), outperforming both individual deep learning components and traditional ML baselines. Day-wise generalisation experiments demonstrated that CNN-LSTM retains 98.71% accuracy and macro F1 of 0.9341 on unseen Friday traffic—a substantially stronger cross-day transfer than Random Forest (macro F1: 0.8431)—confirming the operational advantage of architectures that model both structural and temporal traffic patterns.

Class-wise analysis shows that performance gains relative to traditional machine learning are most pronounced for minority attack categories, where the Infiltration F1-score improves from 0.4312 (RF) to 0.7512 (CNN-LSTM+SMOTE). The denoising autoencoder achieved 94.81% binary detection at optimal threshold, providing a compact unsupervised anomaly detector suitable for near-real-time pre-filtering of novel traffic patterns outside the supervised training taxonomy.

## References

Axelsson, S. (2000). Intrusion detection systems: A survey and taxonomy (Technical Report No. 99-15). Chalmers University of Technology.

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32. https://doi.org/10.1023/A:1010933404324

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1-58. https://doi.org/10.1145/1541880.1541882

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321-357. https://doi.org/10.1613/jair.953

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). ACM. https://doi.org/10.1145/2939672.2939785

Chollet, F. (2015). Keras: Deep learning library for Theano and TensorFlow. Retrieved from https://github.com/fchollet/keras

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297. https://doi.org/10.1007/BF00994018

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21-27. https://doi.org/10.1109/TIT.1967.1053964

Engelen, G., Rim, V., Meert, W., & Joosen, W. (2019). Troubleshooting an intrusion detection dataset: CICIDS2017. arXiv preprint arXiv:2103.07476.

Farahnakian, F., & Heikkonen, J. (2018). A deep auto-encoder based approach for intrusion detection system. In Proceedings of the 20th International Conference on Advanced Communication Technology (ICACT) (pp. 178-183). IEEE. https://doi.org/10.23919/ICACT.2018.8323687

Farnaaz, N., & Jabbar, M. A. (2016). Random forest modeling for network intrusion detection system. Procedia Computer Science, 89, 213-217. https://doi.org/10.1016/j.procs.2016.06.047

Hall, M. A. (1999). Correlation-based feature selection for machine learning (Doctoral dissertation). University of Waikato, Hamilton, New Zealand.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A deep learning approach for network intrusion detection system. In Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (pp. 21-26). ICST. https://doi.org/10.4108/eai.3-12-2015.2262516

Kim, J., Shin, N., Jo, S. Y., & Kim, S. H. (2016). Method of intrusion detection using deep neural network. In Proceedings of the IEEE International Conference on Big Data and Smart Computing (BigComp) (pp. 313-316). IEEE. https://doi.org/10.1109/BIGCOMP.2016.7425916

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR). arXiv:1412.6980.

Lashkari, A. H., Draper-Gil, G., Mamun, M. S. I., & Ghorbani, A. A. (2017). Characterization of Tor traffic using time based features. In Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP) (pp. 253-262). SciTePress. https://doi.org/10.5220/0006220702530262

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. https://doi.org/10.1038/nature14539

Lemaitre, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning. Journal of Machine Learning Research, 18(17), 1-5.

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems 30 (NeurIPS 2017) (pp. 4765-4774). Curran Associates.

McAfee. (2018). Economic impact of cybercrime: No slowing down. McAfee LLC.

Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive dataset for network intrusion detection systems. In Proceedings of the Military Communications and Information Systems Conference (MilCIS) (pp. 1-6). IEEE. https://doi.org/10.1109/MilCIS.2015.7348942

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

Quinlan, J. R. (1993). C4.5: Programs for machine learning. Morgan Kaufmann.

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP) (pp. 108-116). SciTePress. https://doi.org/10.5220/0006639801080116

Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 41-50. https://doi.org/10.1109/TETCI.2017.2772792

Tan, Z., Jamdagni, A., He, X., Nanda, P., & Liu, R. P. (2014). A system for denial-of-service attack detection based on multivariate correlation analysis. IEEE Transactions on Parallel and Distributed Systems, 25(2), 447-456. https://doi.org/10.1109/TPDS.2013.146

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA) (pp. 1-6). IEEE. https://doi.org/10.1109/CISDA.2009.5356528

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems 30 (NeurIPS 2017) (pp. 5998-6008). Curran Associates.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P. A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 11, 3371-3408.

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. IEEE Access, 7, 41525-41550. https://doi.org/10.1109/ACCESS.2019.2895334

Wang, W., Zhu, M., Zeng, X., Ye, X., & Sheng, Y. (2017). Malware traffic classification using convolutional neural network for representation learning. In Proceedings of the International Conference on Information Networking (ICOIN) (pp. 712-717). IEEE. https://doi.org/10.1109/ICOIN.2017.7899588

Wang, W., Yang, J., & Liu, Y. (2017). Towards a robust intrusion detection system using machine learning and oversampling techniques for imbalanced classes. In Proceedings of the International Conference on Machine Learning and Cybernetics (pp. 474-479). IEEE.

Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5, 21954-21961. https://doi.org/10.1109/ACCESS.2017.2762418