Multi-Class Network Intrusion Detection System

Sirijan Ra Na[1], J.C. Miraclin Joyce Pamila[2]
[1]*Department of Computer Science and Engineering, Coimbatore, India*
[2]*Ph.D, Department of Computer Science and Engineering, Coimbatore, India*

***Abstract*—In extensive growth of cyber world, there is also an increase in vulnerability and data theft. Deep Learning Based Network Attack Classification seeks to fill a crucial gap in modern network security by developing an effective Anomaly based Intrusion Detection System (IDS). Cyber threats are always changing from large Distributed Denial of Service (DDoS) attacks to less obvious, under represented intrusions like SQL-Injection. Traditional Signature based Intrusion Detection System (IDSs) often do not meet these challenges. This work presents a deep learning based Convolutional Neural Network model along with a Hybrid Contractive Autoencoder model for multiclass classification of network traffic into Benign and Multiple anomalous attack categories. The dataset used is CSE-CIC-IDS2018 which has been created for analyzing DDoS data and the data is divided into various files based on date. The model demonstrated proving the reliability of the Convolutional Neural Network in Multi-classifying the anomalous versus non-anomalous.**

***Index Terms*—Network Security, Anomaly based detection, Cybersecurity, Deep Learning, SQL-Injection.**

I. INTRODUCTION

Networks are constantly attacked by malware, phishing, DoS, etc. Network Intrusion Detection System (NIDS) helps to detect the malicious activities. The NIDS are signature based looks for known attack patterns and anomalies, which flags deviation from normal behavior. Hybrid system combine both methods to improve accuracy. Other types include host-based IDS(HIDS), which monitors a single host and protocol-based IDS, which analyzes network protocols.

For instance, there was a wave of major cyber incidents between April and July 2025, thus showing the different attack patterns and leading to massive data compromise across government and corporate sectors. A June 2025 the breach of a Chinese Surveillance Network was unparalleled in scale, exposing more than 4 billion records containing highly sensitive data. Such as personal identification and banking information. The mechanism was the exposure of an enormous unsecured database, which was easily accessible without a password. In July 2025, Microsoft had a 2.4TB data leak which wasn't a direct hack, but was due to misconfiguration of their systems meaning an error in so, setting up the security controls left this enormous volume of data exposed to unauthorized access. The VeriSource data breach in April 2025, which compromised personal information, including the social security numbers of a breach affecting about 4 million people, was attributed to a prior network intrusion detected in February of this previous year, for which the notification process was delayed until 2025 due to the prolonged investigation of the scope. Cisco the July 2025 Cyberattack was a high-profile, targeted intrusion which was performed by a coalition of advanced hacking groups, including UNC2447, Lapsus and Yanluowang by gaining the employees personal google account which synchronize corporate credentials then the attackers' bypasses security via MFA fatigue method. In July 2025 the Twitter user data leak revealed the information of more than 200 million users, which looks to be a repackaging and consolidation of previously scraped data initially gathered through an exploited vulnerability within the Twitter API back in 2022 that allowed external parties to match phone numbers or emails to Twitter IDs. On the mid of May 2025, Largest Multi Vector DDoS attack ever recorded was blocked by Cloudflare, 7.3 terabits per second where 99.96 percent of the attack traffic was categorized as UDP floods remaining 0.04 percent was identified as QOTD reflection attacks, NTP reflection attacks, Mirai UDP flood, echo reflection, Portman and RIPv1 amplification attack. These attacks were originated from 1,22,445 source IP address spanning 5,433 Autonomous system across 161 countries,

where half of the attacks were originated from Vietnam and Brazil approximately quarter each. Other one-third from US, Thailand, Saudi Arabia, China, Taiwan, Ukraine and Indonesia. The largest contributor of the DDoS attack traffic was Telefonica Brazil which was responsible for 10.5 percent of all attack traffic.

A NIDS and HIDS represent the two fundamental approaches to monitor malicious activity within an environment. NIDS act as the perimeter guard, monitoring all network packet traffic. It flows across an entire segment from one single point like a router. It offers a broad and efficient view of external threats but blind in encrypted data and activity happening within a compromised host. HIDS act as a local watchdog installed directly on a single machine (Host). It monitors internal system activities-log files, system calls and file changes. It provides an in depth and granular view of host behaviour and can identify potential threats that have already bypassed the network edge or that originated internally. Applying Convolutional Neural Network (CNN) to network data, involves transferring the raw input sequence of network packets into a structured, grid-like matrix. The CNN's Convolutional Filter slides across this matrix of layers, automatically based on stride learning complex, high-level spatial correlation among various feature extracted from the packets, such as byte sequence, packet lengths and flow statistics. The network learns to recognize the patterns and unusual feature relationships that states non-malicious or malicious traffic flows, like those characteristics of a SQL-injection or any other malicious attacks. Hybrid Contractive Autoencoder is a neural network model designed to combine the unsupervised learning of an autoencoder with supervised precision of a classifier. This multi-task learning framework that doesn't just compress data, but learns to extract features that are specifically useful for security or anomaly detection.

II. RELATED WORKS

Deep learning has become the dominant method in intrusion detection, Recent research has applied deep learning to intrusion detection in SDN (Software Defined Networking) environments, however significant limitations persist across existing models. Aktar et al. [1] employed a multilayer deep neural network trained on flow-based features for DDoS detection, but the DNN required extensive manual pre-processing and reduced demonstrate adaptability under highly dynamic SDN traffic fluctuations. Elsayed et al. [2] introduced DDoSNet, a stacked autoencoder combined with a softmax classifier trained on CIC-DDoS2019 traffic, although the model achieved high accuracy. The autoencoder based reconstruction process struggled to generalize to SDN controller level flow patterns and exhibited scalability issues under large distributed attacks. Mansoor et al. [3] utilized a deep feed-forward neural network optimized for SDN controller logs, yet the model imposed considerable computational overhead on the controller, significantly increasing flow rule installation delays during attack bursts. Halbouni et al. [4] proposed a CNN-LSTM hybrid architecture that first extract spatial correlations using Convolutional layers and then learned temporal dependencies with LSTM units, despite strong classification results the model's depth and the parameter count made real time deployment difficult on constrained SDN control plane resources. Ge et al. [5] implemented a Deep Brief Network (DBN) for IOT intrusion detection, leveraging hierarchical features learning; however, the unsupervised pretraining phase was highly sensitive to heterogeneous traffic and did not align well with SDN's structure flow statistics. Yuan et al. [6] designed a light weight distributed IDS using a shallow neural classifier and PCA-based feature reduction, which improved computational efficiency but showed limited capability against multi-vector DDoS attacks characteristic of modern SDN environments. Yuan et al. [7] proposed an adversarially robust convolutional model by integrating adversarial training techniques, yet the additional training complexity slowed convergence and still left the system vulnerable to real-time adaptive evasion attacks. Nakip et al. [8] introduced an online self-supervised learning framework based on contrastive learning for continuous IDS adaption, but its reliance on stable feature distributions caused unstable detection when traffic patterns shifted abruptly. Behal et al. [9] used a deep feed-forward classifier trained on NSL-KDD and custom DDoS datasets however, the model lacked validation on high-bandwidth SDN traffic and suffered performance drops under real-world throughput conditions. Salman et al. [10] combined gradient-boosting models with a

deep neural network in a distributed IDS architecture, but synchronization overhead across distributed nodes and delayed inference during high-volume DDoS bursts restricted its applicability to real-time SDN deployments.

2.1 Research Gap

Based on the literature review and base paper work, identified that there has been very few research on multi-class classification about network anomalies detection and model have been developed for only selective attacks-dataset constrains. In this work those limitations has been overcome by multi-class classification using CNN model which classifies Non-anomalous(Benign) and Anomalous attacks such as SQL-Injection, SSH-Bruteforce, Infiltration, FTP-Bruteforce, DDoS attack Hulk, DDoS attack SlowHTTPTest, BOT, DDoS attack Hoic and Hybrid Contractive Autoencoder which classifies Non-anomalous(Benign) and Anomalous attacks such as SQL-Injection, SSH-Bruteforce, Infiltration, FTP-Bruteforce, DDoS attack Hulk, DDoS attack SlowHTTPTest, BOT. The Hybrid Contractive Autoencoder model tries to reconstruct the network data to multi-class between Benign and other attacks. The dataset constrains are been overcome by combining every individual day network data from CSE-CIC-IDS2018 dataset into a single file and then for attacks like SQL-Injection are very rare in the dataset which promotes data imbalance. Which has been overcome by data augmentation method by creating synthetic data using SMOTE (Synthetic Minority Over-Sampling Technique) method.

III. METHODOLOGY

This section outlines the experiment conducted in this research, which focused on multi-class classification involving various non-anomalous and anomalous attack using CSE-CIC-IDS2018 dataset.

3.1 Convolutional Neural Network

The proposed NIDS model uses CNN, based on supervised learning approach. The CNN was developed by Yann LeCunn in 1989, who applied backpropagation to train a network for handwritten zip code recognition. CNN in network data involves transforming the raw input sequence of network packets into a structured, grid-like matrix suitable for convolutional operations. The CNN's Convolutional filters then slide across the matrix of layers, automatically based on stride learning complex, high-level spatial correlations among various features extracted from the packets, such as byte sequences, packet lengths and flow statistics. The network learns to recognize the patterns and unusual feature relationships that states non-malicious or malicious traffic flows, like those characteristics of a DDoS attack or any other malicious attack. The CNN consists of different levels of layers, such as input, convolutional, pooling, normalization, dropout, flattening, dense (fully connected layers) and the output layer (produces the final predictions depends on the task whether binary, malicious or linear). Each convolutional layer comprises several filters, which are responsible for extracting features such as edges, shapes, or patterns from the input. The output size of a convolutional layer depends on the input dimensions, filter size, stride and padding.

The output width and height are computed as

$$Width2 = \frac{[W1 - F + 2P]}{S} + 1$$

$$Height2 = \frac{[H1 - F + 2P]}{S} + 1$$

Where W1 and H1 are the input width and Height, F is the filter size, S is stride and P padding. The output depth D2 is equal to the number of filters K used:

$$D2 = K$$

The core operation of a convolutional layer at a particular spatial position (i,j) is expressed as:

$$zij = \sum_{m=0}^{F-1} \sum_{n=0}^{F-1} I(i+m, j+n).K(m,n) + b$$

Where zij denotes the output feature map at position $(i,j), I(i+m, j+n)$ represents the input value at the corresponding location, $K(m,n)$ is the filter weight at position $(m,n)$and b is the bias term added to each output element. This operation performs a dot product between the filter and a local input patch, followed by the addition of a bias, allowing the network to detect specific patterns within the input.

The CNN training process aims to minimize the loss function(L) on dataset D_n by optimizing the filter weights and biases, $\theta = \{K, b\}$, such that the learned feature maps can effectively represent the input data for subsequent classification or detection tasks. For classification the loss function is cross-entropy and for regression is mean squared error.

3.2 Hybrid Contractive Autoencoder

In the realm of machine learning and neural networks, the evolution of autoencoders has been advancing in unsupervised learning. Among the various types of autoencoders, the Contractive Autoencoder (CAE) stands out due to its unique approach to feature learning. The NIDS model utilizes Hybrid Contractive Autoencoder model designed to learn robust and low-dimensional representations from high-dimensional network flows. Autoencoders (AE) are traditionally used to learn simple representations from data but are only used for reconstruction. HCAE adds a Contractive Regularization component as well as a Multi-Task Classification component to the autoencoder. The result is a "contractive" mapping that is insensitive to changes in the network traffic flows, making the model extremely good at capturing even small changes in malicious flows.

Encoder and Bottleneck Geometry,

The HCAE transforms raw input vectors $x \in \mathbb{R}^d$ through a series of non-linear transformations. The encoder $f\theta$ maps the input to a hidden latens space(bottleneck) $z\varepsilon\mathbb{R}^k$ $k < d$. The encoding process is defined as:

$$z = \sigma(W_e x + b_e)$$

Where $W_e$represents the weight matrix, b_e is the bias, and σ is the activation function. The HCAE utilizes a Contractive penalty by calculating the Frobenius norm of the Jacobian matrix $J_f(x)$ of the encoder activations with respect to the input. This forces the model to ignore noise and focus on the manifold of the network data:

$$\| J_f(x) \|_F^2 = \sum_{i,j} \left(\frac{\partial zi}{\partial xj}\right)^2$$

Task Architecture, unlike standard Auto Encoders, the HCAE architecture bifurcates at the bottleneck into two distinct functional branches:

1) The Decoder Branch: Reconstructs the input $\hat{x}$ to ensure the latent space preserves essential flow features, computed as $\hat{x} = \sigma'(W_d z + b_d)$.
2) The Classifier Branch: Maps the latent vector z directly to a label space y using a Softmax activation for multi-Class attack detection:

$$P(y = c|z|) = \frac{e^{wT}c^{z+b_c}}{\sum_{j=1}^{K} e^{w}T_c z + b_j}$$

Loss Function Optimization, The HCAE training process aims to minimize a composite loss function (L_{total}) that balances reconstruction integrity with classification precision. The objective function is expressed as:

$$L_{total} = L_{MSE}(x, \hat{x}) + \lambda . L_{CCE}(y, \hat{y}) + \gamma \left\| J_f(x) \right\|_F^2$$

Where $L_{MSE}$is the Mean Squared Error for reconstruction, L_{CCE} is the Categorical Cross-Entropy for attack classification, and λ is the hyperparameter (set to 15.0 in this implementation) the weights the importance of detection accuracy. The optimization of the parameter set $\theta = \{W, b\}$ is performed using the Adam Optimizer, allowing the network to detect patterns such as Dos attacks, Botnet communications and SQL-injections by identifying anomalies in the compressed feature relationships.

3.3 DATASET

The dataset has been used is CSE-CIC-IDS2018 dataset, a benchmark dataset widely used for network related works. This dataset contains real-world simulated network traffic data's, generated in a virtual environment, covering both anomalous and non-anomalous network patterns. This is highly imbalanced dataset where most of the attacks are comparatively very low in count than others. To overcome this situation 1lakh samples were taken from each network type to classify, but failed to extract exactly 1,00,000 samples for SQL-Injection because this class count is low in the dataset which has been taken for deploying the CNN model. Meanwhile for Hybrid Contractive Autoencoder around 10000 samples were taken for each class.

3.4 DATASET PRE-PROCESSING

Before feeding the data into the model, pre-processing is applied to ensure quality and consistency. Non-numeric and non-informative columns such as Flow ID, Timestamp, Source IP and Destination IP are excluded from the dataset, because they won't help the model to learn attack patterns which is not necessary. Flow ID-It's just a unique identifier for each network flow and doesn't contain any useful pattern or feature for detecting attacks. Timestamp-Shows the time when packet was captured, which doesn't directly relate to whether traffic is malicious or not. Source IP/Destination IP-These are just network addresses. Using them could make the model memorize specific

IPs instead of learning general attack behaviour. By removing these columns, the model focusses only on relevant numerical and statical features (like packet size, duration, bytes,) which represents actual network behaviour. All the infinite and missing values are handled by replacing them with zeros to maintain numerical stability. Data augmentation plays an important role in balancing the pre-processed data. In this system SMOTE has been used to generate synthetic data for underrepresented attack classes. By selecting a data point from minority class and identifying its k-nearest neighbours in the same class. Then it creates a new synthetic network patterns by interpolating between the feature values of original samples and one of its neighbours. SMOTE requires at least two samples to generate new synthetic data points. If the target class has fewer samples than the specified threshold, SMOTE is applied to generate synthetic samples by interpolating between existing feature vectors of the minority class. The function then prints the updated class distribution, confirming successful augmentation. After augmentation individual datasets were merged into single balanced corpus for model training.

3.5 Proposed Model

3.5.1 CNN

The NIDS is 1D CNN based, a deep learning model. The intrusion detection problem assigning a class label $y\epsilon\{1,2,\dots 9\}$ a network traffic sample x representing one of the nine categories, including Benign, DDoS, Brute Force, and SQL-Injection. The model aims to learn meaningful representation from the traffic data, capturing both local and global patterns through multiple convolutional and dense layers.

The model employs a multi-layer 1D CNN architecture, designed to process tabular flow data with 42 selected features reshaped into a format suitable for Conv1D layers. Each Convolutional layer applies Kearnable filters of size F over the input, producing feature maps z_i according to:

$$z_i == \sum_{m=0}^{F-1} I(i+m).K(m) + b,$$

$I(i+m)$ denotes input at position $i+m, K(m)$ represents filter weight and bias is b. The output length of each convolutional layer is determined by:

$$L_2 = \frac{[L_1 = F + 2P]}{S} + 1,$$

Where $L1$ is the input length, Padding (P) and Stride (S). Then feature maps then pass through a nonlinear activation function ReLU:

$$a_i = ReLU(z_i) = \max(o, z_i),$$

Followed by batch normalization:

$$\hat{x} = \frac{x - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}, \quad y = \gamma\hat{x} + \beta,$$

And max pooling to reduce the spatial dimension:

$$p_j = max(a_j S, a_j S + 1, \dots, a_j S + P - 1),$$

Dropout is applied to prevent overfitting:

$\overline{a_i} = a_i.\gamma_i, \gamma_i \sim Bernoulli(p)$,

Where p is the probability of keeping a unit active. Then feature maps passed through a fully connected dense layer, and in the final layer uses a Softmax activation function to perform multi-class classification:

$$\hat{y}k = \frac{e^{zk}}{\sum_{j=1}^{C} e^{zj}}, k = 1,2, \dots 9$$

Where $C = 9$ represents the classes.

The model is trained to minimize the categorical cross-entropy loss:

$$L = -\sum_{k=1}^{C} yk \log(\hat{y}k),$$

yk is the true label for class k and $\hat{y}k$ is the predicted probability. SMOTE is applied before training to handle class imbalance, which generates new samples x_{new} for minority classes especially used for SQL-Injection as:

$x_{new} = x_i + \lambda(x_{nn} - x_i), \lambda \sim U(0,1)$,

Minority class instance (x_i) and one of its k-nearest neighbours (x_{nn}).

The proposed intrusion detection system is developed on 1D CNN to multi-classify network traffic into nine distinct categories. The process begins with data cleaning, data augmentation using SMOTE for creating a balanced dataset. The balanced dataset is split into 70% for training, 15% for validation and 15% for testing. The CNN model employs multiple convolutional layers with ReLU activation, combined with Batch Normalization, Dropout and Max Pooling Layer to extract deep spatio-temporal features, with fully connected dense layer performing the classification.

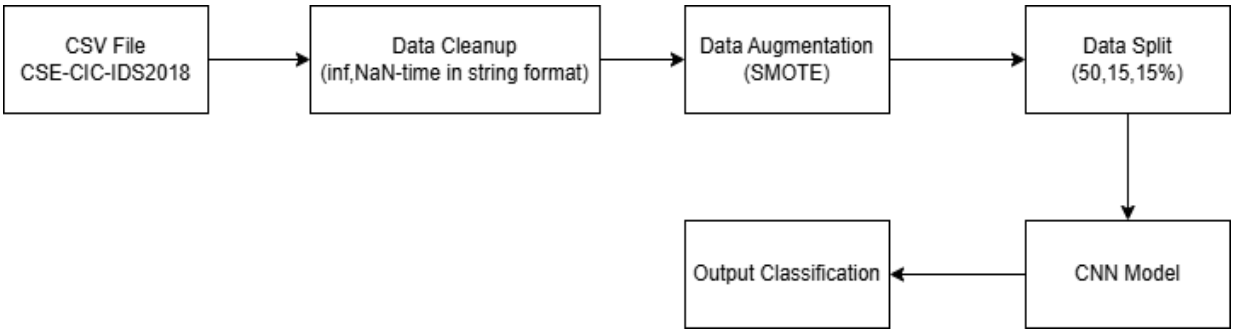

Figure 1. System Architecture

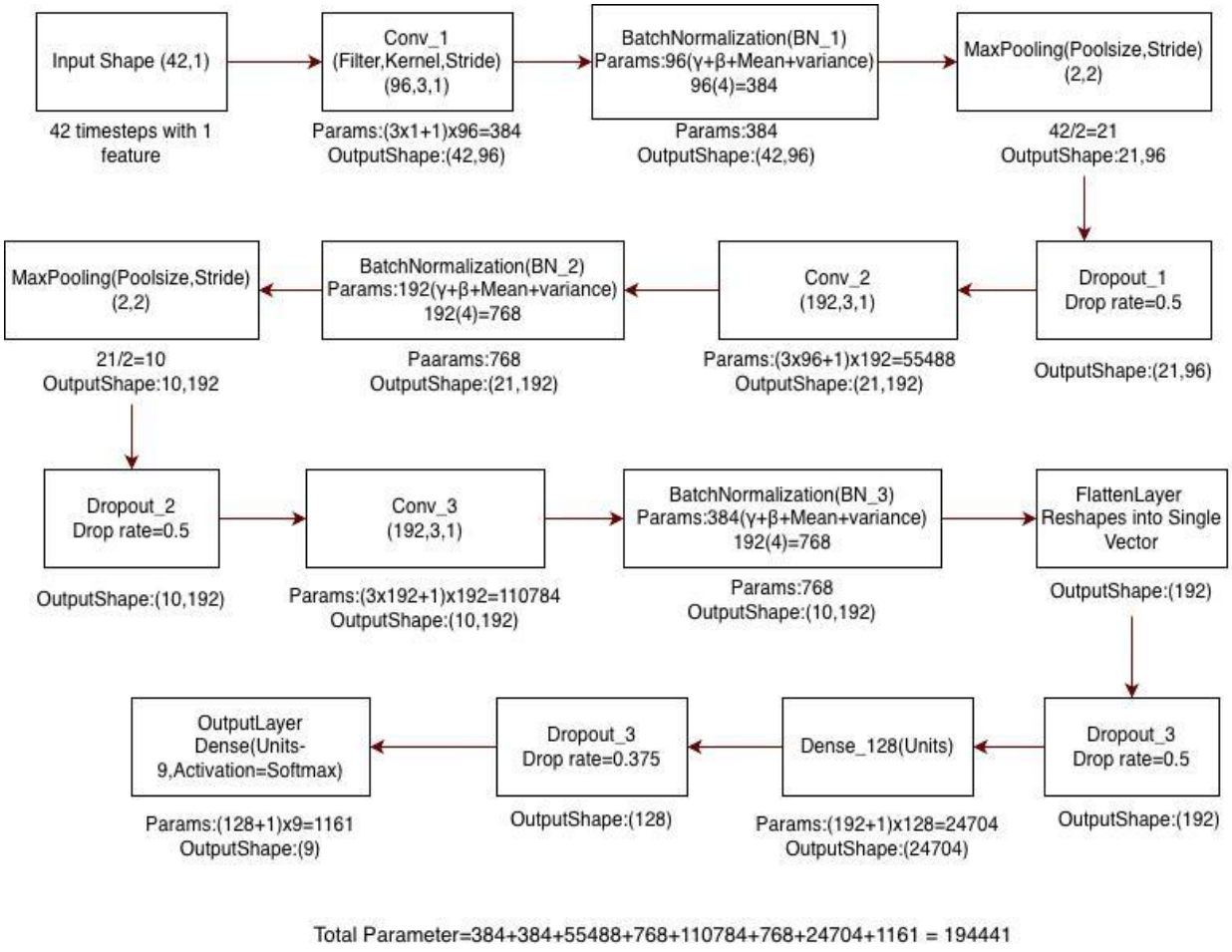

Figure 2. CNN Model Architecture

The model begins with an input layer corresponding to 42 selected features, reshaped to fit the Conv1D format. The first convolutional layer employs 96 filters with a kernel size of 3, producing feature maps that capture low-level spatial dependencies within the data. This is followed by batch normalization to stabilize learning, a Max Pooling layer to reduce dimensionality, and Dropout to prevent overfitting. The second convolutional block expands to 192 filters, allowing the network to extract more complex attack patterns, again accompanied by normalization, pooling and dropout layers for enhanced generalization. A third convolutional layer with 192 filters further refines the learned representations and a Global average pooling layer is applied to condense the learned spatial information into a compact feature

vector. This output is passed through a dense layer of 128 neurons with ReLU activation for non-linear transformation, followed by a dropout layer to minimize overfitting. Finally, a Dense output layer with 9 neurons and a softmax activation performs multi-class classification predicting one of the nine traffic types such as Benign, BOT, SQL-Injection. The complete model comprises approximately 1,94,441 parameters of which 1,93,481 are trainable and 960 are non-trainable, providing a balance between representational power and computational efficiency. This architecture effectively captures both local and global dependencies within network traffic, enabling accurate detection of subtle intrusion behaviour across multiple attack categories.

3.5.2 HYBRID CONTRACTIVE AUTOENCODER

The proposed Network Intrusion Detection System is based on a Hybrid Contractive Autoencoder with a Supervised Classification Head, a deep learning architecture designed for multi-class network traffic classification. Intrusion detection then becomes a class labelling task. $y \in \{0,1 \dots 8\}$

to a network traffic sample denoted as x, which belongs to one of eight categories of network traffics including Benign, Bot, DoS-Hulk, DoS-SlowHTTPTest, FTP-BruteForce, SSH-Bruteforce, Infiltration and SQL-Injection. Allowing the model to jointly optimize the reconstruction of network features and classification of attacks enables it to build robust and informative representations of network flows, both generally and discriminatively.

Encoder

The encoder transforms the input feature vector $x \in \mathbb{R}^d$ into a compact latent representation z :

$$h_1 = LeakyReLU\big(BN(W_1 x + b_1)\big)$$
$$h_2 = LeakyReLU\big(BN(W_2 h_1 + b_2)\big)$$
$$z = \tanh(W_3 h_2 + b_3)$$

Where,

W_i, b_i are trainable weights and biases,

BN denotes Batch Normalization and

LeakyReLU introduces non-linearity.

To improve robustness, Gaussian noise is injected into the latent representation during training:

$$\tilde{z} = z + \in, \quad \in \sim N(0, \sigma^2)$$

This encourages the model to learn stable, noise-resistant embeddings.

DECODER

The decoder attempts to reconstruct the original input from the latent vector:

$$\hat{x} = W_5 ReLU(W_{4\tilde{z}} + b_4) + b_5$$

The reconstruction loss is computed using Mean Squared Error (MSE):

$$L_{rec} = \big|\big|x - \hat{x}\big|\big|_2^2$$

This forces the encoder to preserve essential structural information about network traffic.

CLASSIFIER HEAD (ATTACK PREDICTION)

The latent representation is also passed to a supervised classifier:

$$h_c = LeakyReLU\big(BN(W_c z + b_c)\big)$$
$$\hat{y} = Soft\max(W_o h_c + b_o)$$

The Softmax output gives class probabilities:

$$\hat{y}_k = \frac{e^{zk}}{\sum_{j=1}^{C} e^{zj}},\ k = 1,2, \dots, 8$$

Where C=8 represents the number of traffic classes.

LOSS FUNCTION

The model is trained using a joint loss function combining reconstruction and classification objectives:

$$L_{total} = L_{rec} + \lambda L_{clf}$$

Where $\lambda = 15$ balances the two tasks and classification loss is Sparse Categorical Cross-Entropy:

$$L_{clf} = -\sum_{k=1}^{C} yk \ \log\big(\widehat{yk}\big)$$

Class weights are incorporated to address residual imbalance:

$$L_{clf}^{weighted} = w_k . L_{clf}$$

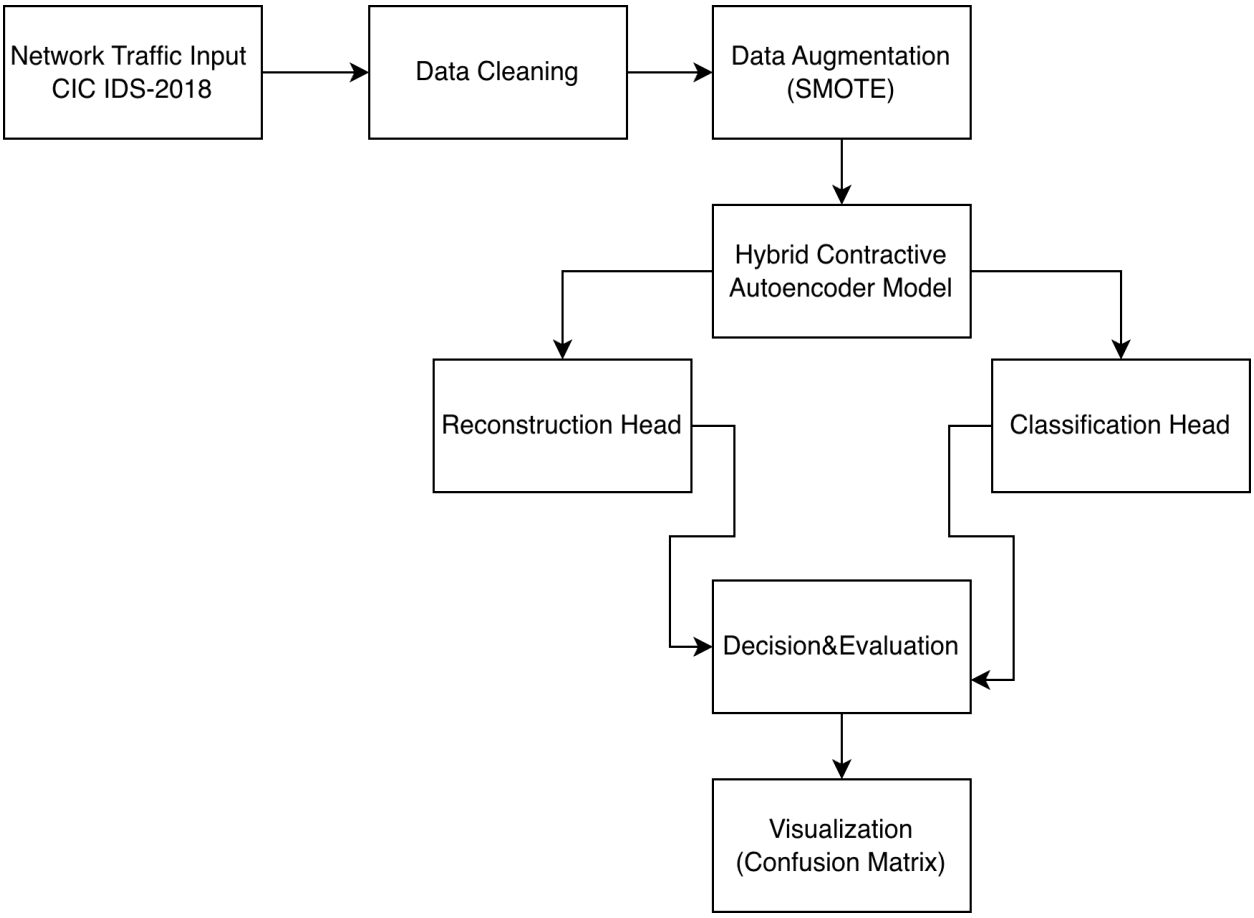

Figure 3. System Architecture

Initially the model cleans the data, then normalizes the labels and performs deterministic sampling in order to have an equal number of samples for each class. It then imputes the missing values with the median, followed by feature scaling. Finally, it adds some standard normal noise to the training samples to simulate real-world variability. The model multi-classify Benign Normal, Bot, DoS attacks-Hulk, Dos attacks-SlowHTTPTest, FTP-BruteForce, Infilteration, SQL Injection, and SSH-Bruteforce network datas from the CIC IDS2018 dataset. Unlike Standard Autoencoders, The HCAE model applies Contractive penalty(The Frobenius norm of the Jacobian matrix) which forces the model to be resistant to small perturbations in the input data by effectively teaching it to ignore noise and focus only on the underlying structure of the data. The model splits at the Bottleneck layer as decoder(Focus on reconstruction ensuring no vital information is lost during compression) and classifier(Focuses on categorizing the data as distinguishing between normal traffic versus attacks). Weighted loss function L_{total} allows to tune the balance between how well the model reconstructs data versus how accurately it classifies it using hyperparameters like λ.

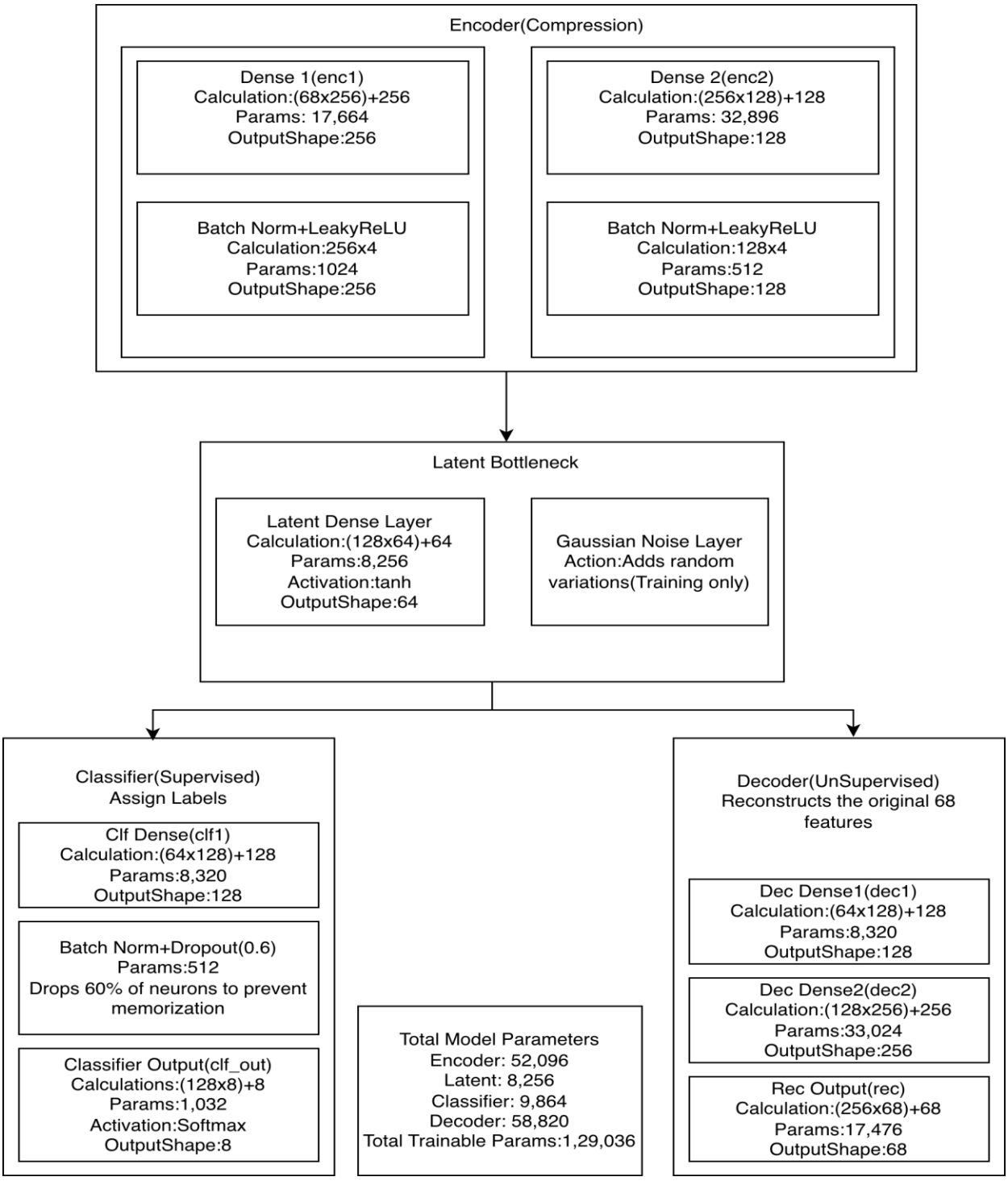

Figure 4. Hybrid Contractive Autoencoder Model Architecture

The process starts with an Input Layer comprising 68 selected features after standardization and data cleaning of the CICIDS 2018 dataset. The encoder architecture consists of a dense layer comprising 256 neurons to project the features to a higher-dimensional space to explore complex feature correlations. It follows with a batch normalization layer to regularize training, Leaky-ReLUs activation function with alpha set to 0.2 to preserve gradient flow during backpropagation, and dropout with 0.5 probability to

prevent overuse of certain features. Another higher-dimensional space encoding layer comprising 128 neurons follows to refine the features before projecting them to a bottleneck layer comprising 64 neurons with a tanh activation function. This "core signature" of the features represents the latent space of network traffic. To prevent overuse of certain patterns by the network, Gaussian noise with 0.2 probability is injected during training. From the 64-dimensional latent vector, the model splits into two functional heads: The

Decoder Branch (Reconstruction): This branch seeks to achieve the reconstruction of all 68 features to ensure the latent representation maintains the highest integrity of data. It involves the following steps: passing the latent vector through a dense layer of 128 neurons (dec1), then passing it through another dense layer of 256 neurons (dec2), and finally passing it through the Reconstruction Output layer of 68 neurons with linear activation. This branch computes the MSE between the input and the output to influence the encoding.

The Classifier Branch (Supervision): The latent vector is also passed through a classification branch that begins with a dense layer of 128 neurons (clf1). Batch Normalization is followed by Leaky ReLU and an aggressive dropout rate of 0.6 to prevent overfitting of the synthetic/upsampled attack classes. The Dense Output layer of 8 neurons with Softmax activation is utilized for multi-class classification of the attack classes such as Benign, Bot, or SQL Injection.
The complete model consists of exactly 129,036 parameters in total, with 127,996 being trainable and 1,040 being non-trainable (used for the Batch Normalization Stats). In utilizing the Joint Loss Function, which weighs classification error 15 times more than reconstruction error, the model essentially uses the architecture to provide a context wherein the encoder emphasizes "attack discriminating" features and maintains a wholesome latent space for accurate intrusion detection.

IV. EVALUATION METRICS

The following performance metrics are used to evaluate the model: Accuracy, Precision, Recall and F1-Score.

These are calculated using measures such as, true positive (TP), true negative (TN), false positive (FP) and false negative (FN).

1. True Positive: Total instances that were correctly predicted as positive.
2. True Negative: Total instances that were correctly predicted as negative.
3. False Positive: Total instances that were incorrectly predicted as positive.
4. False Negative: Total instances that were incorrectly predicted as negative.

Precision: Number of true positives among all positive predictions. Ratio of the number of correctly classified anomalous and non-anomalous attack patterns to the total number of classified attack patterns.

$$Precision = \frac{TP}{TP + FP}$$

Recall: The amount of true positive to all actual positive instances. Ratio of the number of correctly classified anomalous instances to the number of actual anomalous instances.

$$Recall = \frac{TP}{TP + FN}$$

F1-Score: A single metric that balances precision and recall. It is the harmonic mean of the two, meaning it gives more weight to values closer to the average and penalizes extreme values in either precision or recall.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Accuracy: The total number of predictions correct predictions.

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

For evaluation confusion matrix was also used along with the above-mentioned metrics.

Table 1. Confusion Matrix

		Predicted Class	
		Anomaly	Normal
Actual Class	Anomaly	TP	FN
	Normal	FP	TN

4.1 Result Analysis

The evaluation metric for CSE-CIC-IDS2018 dataset, calculated Precision, Recall, F1-Score and accuracy fir CNN model.

Table 2. Evaluation Metrix

CLASS	PRECISION	RECALL	F1-SCORE	SUPPORT
Benign	0.6938	0.7984	0.7424	15,000
BOT	0.9608	0.9991	0.9795	15,000
DDOS attack-HOIC	0.9991	1.000	0.9995	15,000
DDOS attack-Hulk	0.9998	0.9998	0.9998	15,000
DDOS attack Slow HTTP Test	0.7870	0.4720	0.5901	15,000
FTP-Brute Force	0.6232	0.8733	0.7274	15,000
In filteration	0.7769	0.6215	0.6905	15,000
SQL Injection	0.9813	0.9867	0.9840	15,000
SSH-Brute Force	0.9978	0.9997	0.9988	15,000
Accuracy			0.8612	1,35,000

The evaluation metric for CSE-CIC-IDS2018 dataset, calculated Precision, Recall, F1-Score and accuracy fir Hybrid Contractive Autoencoder model.

Table 3. Evaluation Metrix

CLASS	PRECISION	RECALL	F1-SCORE	SUPPORT
Benign	0.7442	0.5013	0.5991	3000
BOT	0.8337	0.9990	0.9089	3000
DDOS attack-Hulk	0.9973	0.9940	0.9957	3000
DDOS attack Slow HTTP Test	0.5986	0.9933	0.7471	3000
FTP-BruteForce	0.9797	0.3377	0.5022	3000
Infilteration	0.6500	0.7107	0.6790	3000
SQL Injection	0.9321	0.9610	0.9463	3000
SSH-Brute Force	0.9967	0.9997	0.9982	3000
Accuracy			0.8121	24,000

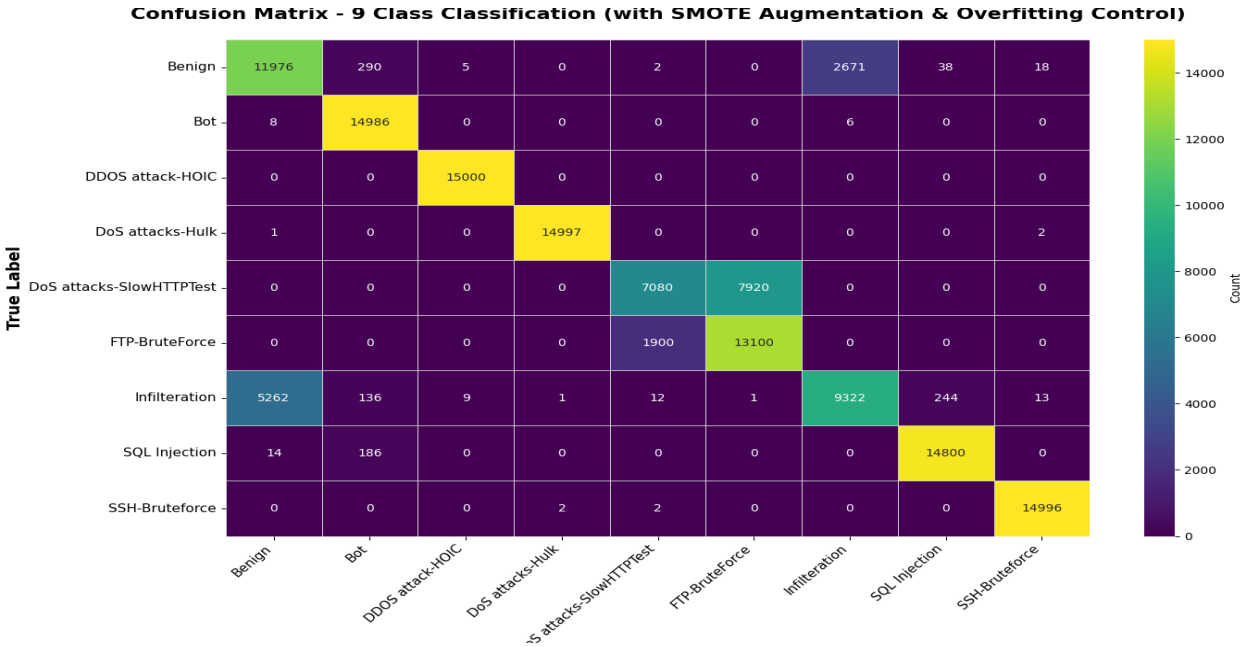

Figure 3. Confusion Matrix for CNN

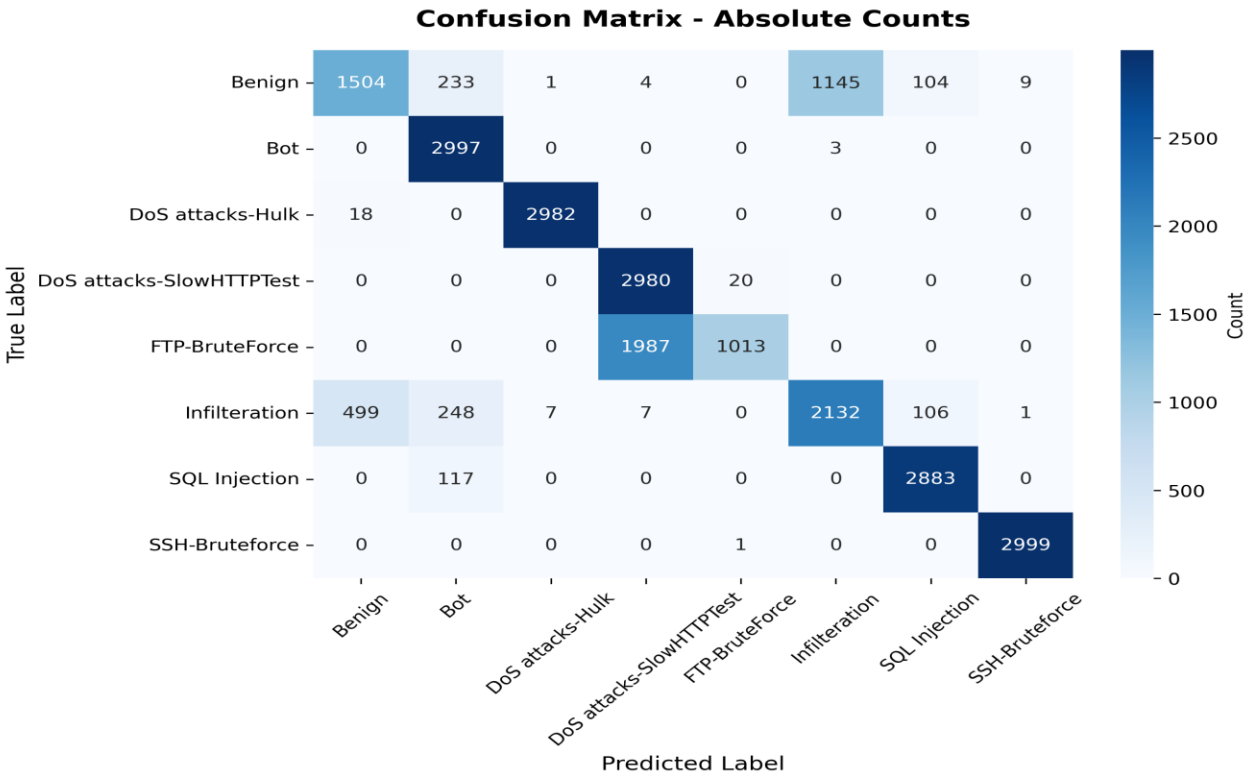

Figure 4. Confusion Matrix for Hybrid Contractive Autoencoder

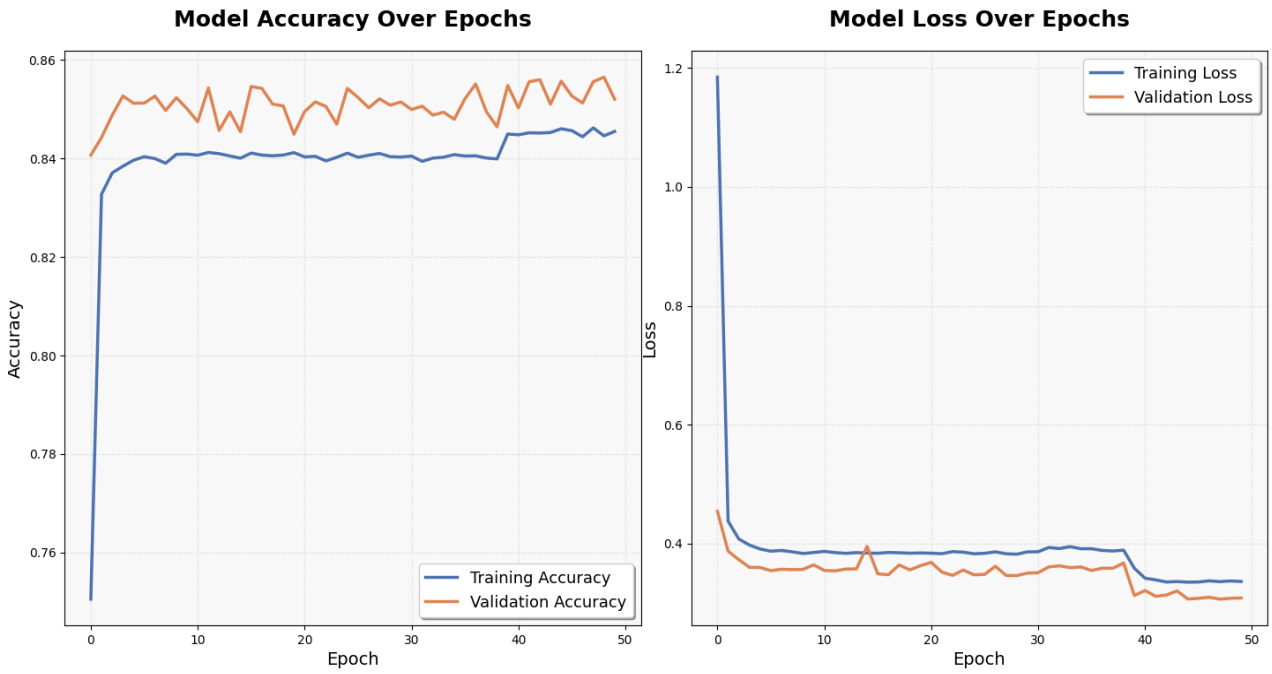

Figure 4. Training Performance graph for CNN

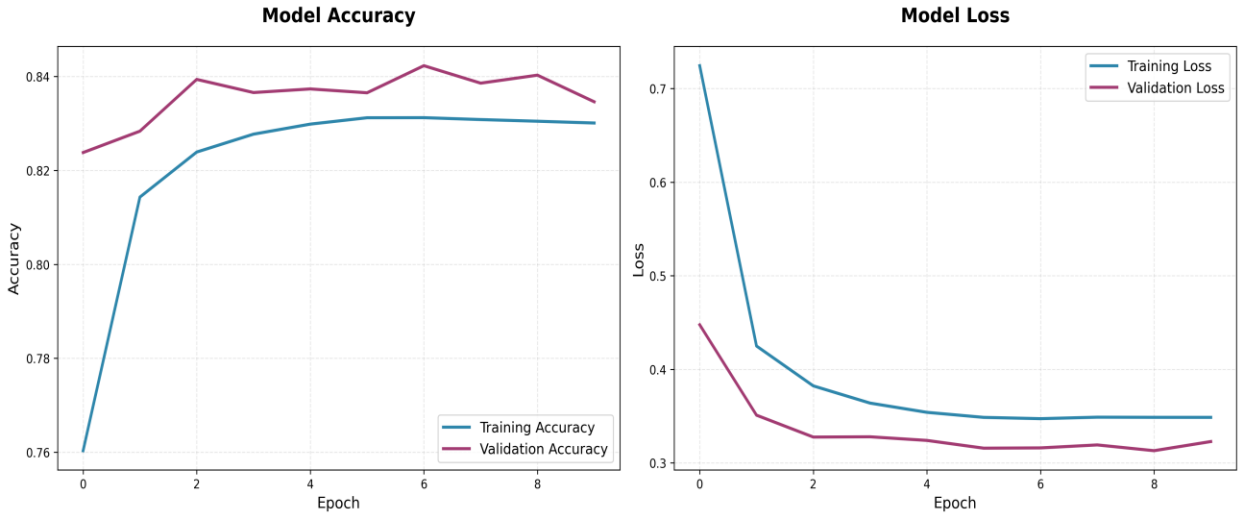

Figure 5. Training Performance graph for Hybrid Contractive Autoencoder

According to the scores of the evaluation metrics from the CSE-CIC-IDS2018 dataset, most attack categories show a strong classification capability for the CNN-based intrusion detection model. As can be observed from Table 2, for the main attack types like Bot, DDoS-HOIC, DDoS-Hulk, SQL Injection, and SSH-Bruteforce, the precision and recall are all above 0.97 in F1-score, which means the detection is highly reliable. The F1-score values of the Benign and Infiltration classes are relatively competitive at 0.74 and 0.69, respectively.

Compared to traditional machine learning baselines in previous studies, the proposed CNN model provides improved performance, especially in precision and F1-score. The overall accuracy attained is 0.8612, which reflects a notable boost in multiclass intrusion detection under imbalanced network traffic conditions. Moreover, the confusion matrix in Figure 3 visually confirms that most classes achieve near-perfect true positive rates with limited misclassification between closely related attack types.

The CNN model provides a higher overall accuracy (86.12%) compared to the Hybrid Contractive Autoencoder (81.21%). This is likely due to the CNN's ability to treat network traffic data as a structured grid, effectively capturing local correlations between packet headers. The CNN shows much higher stability across varied datasets. Its performance

Comparing the experimental results from the models proves that the CNN model functions quite efficiently and can be deployed for intrusion detection in real-time situations. With a high precision, strong recall, and stable convergence, the present model is suitable for environments based on SDN and for large-scale network monitoring applications.

V. CONCLUSION

The 1D-CNN multi-class intrusion detection system described here reliably separates the benign traffic from a wide range of attacks, such as DoS variants, brute-force, bot, infiltration, and SQL injection, by combining systematic feature preparation and SQL-injection augmentation with Conv1D modeling and class-weighting for imbalance. Performance on most classes was observed when evaluating the test set, with accuracy of 86.12%, precision, recall, and F1, while the per-class breakdown pinpoints weaker spots, notably SlowHTTPTest and FTP-BruteForce. As the model is trained on a large, balanced dataset and validated using standard metrics and visualizations, it will also be suited for practical deployment but still will allow room for improvement through targeted data collection, feature engineering, or ensemble methods. CNN is the superior model for general network intrusion detection due to its higher accuracy and better handling of benign traffic. While the Hyb rid Contractive Autoencoder demonstrates specialized value in detecting low-rate attacks like SlowHTTP. An ensemble-based strategy would be more beneficial. CNN can be used as the primary detector for fast and large attacks, while the Hybrid CAE can be used as secondary detector for subtle attacks and slow-rate anomalies.

REFERENCES

[1] Aktar, S. and Nur, A.Y., 2023, Towards DDoS Attack Detection Using Deep Learning Approach, Computers & Security, Vol. 131, pp. 102531.

[2] Elsayed, M.S., Le-Khac, N.-A., Dev, S., and Jurcut, A.D., 2020, DDoSNet: A Deep-Learning Model for Detecting Network Attacks, IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), Vol. 1, pp. 391-396.

[3] Mansoor, Amran, Mohammed Anbar, Abdullah Ahmed Bahashwan, Basim Ahmad Alabsi, and Shaza Dawood Ahmed Rihan, 2023, Deep Learning-Based Approach for Detecting DDoS Attack on Software-Defined Networking Controller, Systems, Vol. 11, pp. 296.

[4] Halbouni, A., Gunawan, T.S., Habaebi, M.H., Halbouni, M., Kartiwi, M., and Ahmad, R., 2022, CNN-LSTM: Hybrid Deep Neural Network for Network Intrusion Detection System, IEEE Access, Vol. 10, pp. 99837-99849.

[5] Ge, M., Fu, X., Syed, N., Baig, Z., Teo, G., and Robles-Kelly, A., 2019, Deep Learning-Based Intrusion Detection for IoT Networks, IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC), Vol. 1, pp. 256-265.

[6] Yuan, S., Li, H., Zhang, R., Hao, M., Li, Y., and Lu, R., 2021, Towards Lightweight and Efficient Distributed Intrusion Detection Framework, IEEE Global Communications Conference (GLOBECOM), Vol. 1, pp. 1-6.

[7] Yuan, X., Han, S., Huang, W., Ye, H., Kong, X., and Zhang, F., 2024, A Simple Framework to Enhance the Adversarial Robustness of Deep Learning-Based Intrusion Detection System, Computers & Security, Vol. 137, pp. 103644.

[8] Nakıp, M. and Gelenbe, E., 2024, Online Self-Supervised Deep Learning for Intrusion Detection Systems, IEEE Transactions on Information Forensics and Security, Vol. 19, pp. 5668-5683.

[9] Behal, S., Saluja, K., and Mittal, M., 2021, Distributed Denial of Service Attack Detection Using Deep Learning Approaches, IndiaCom 2021, Vol. 1, pp. 1-7.

[10] Salman, A.H., and Huah, C.Y., 2025, An Efficient Distributed Intrusion Detection System that Combines Traditional Machine Learning Techniques with Advanced Deep Learning, Mesopotamian Journal of CyberSecurity, Vol. 5, pp. 721–734.

[11] J. Wang L.P. Wang "SDN-Defend: A Lightweight Online Attack Detection and Mitigation System for DDoS Attacks in SDN," Sensors (Basel, Switzerland), vol. 22, issue 21, pp. 8287, 2022.

[12] M. Idhammad, K. Afdel, and M. Beoouch "Detection System of HTTP DDoS Attacks in a Cloud Environment Based on Information Theoretic Entropy and Random Forest" Security and Communication Networks, vol. 2018, pp. 1263123, 2018.

[13] Y.H. Yang H.R. Wang "DDoS attack detection method in SDN environment based on Rényi entropy and BiGRU algorithm," Computer Science, vol. 49(Suppl), pp. 555-561, 2022.

[14] K.S. Sahoo, B. Sahoo, and M. Vankayala, "Detection of Control Layer DDoS Attack using Entropy metrics in SDN: An Empirical Investigation" IEEE Internet Computeng pp. 281-286, 2017.

[15] D. Ferdaus R. Munade and Y. Pur anto "DDoS Attack Detection in Software Defined Network using Ensemble K-means++ and Random Forest," IEEE Internet Computing, pp. 164-169, 2020.

[16] V. Itage M. Javaoe H. Madhukesh ar "DDoS Attack Detection in SDN Environment using Bi-directional Recurrent Neural Network" IEEE Internet Computing, pp. 123-128, 2021.

[17] Elsayed, M. S., Le-Khac, N. A., and Jurcut, A. D., "InSDN: A novel SDN intrusion dataset," IEEE Access, vol. 8, pp. 165263-165284, 2020.

[18] Nguyen, T. T., & Kim, D. H. (2020). An Efficient Flow-Based Multi-Level Network Intrusion Detection System Based on PCA and Machine Learning. Computers, Materials & Continua, 63(3), 1291-1308.

[19] Raja, D. J. S., Sriranjani, R., Parvathy, A., and Hemavathi, N., "A review on distributed denial of service attack in smart grid," in Proceedings of the 2022 7th International Conference on Communication and Electronics Systems (ICCES), Chennai, pp. 812-819, June 2022.

[20] Setitra, M. A., Madoune, S. A., Bensalem, Z. E. A., & Fan, M. Toward Delegating the Detection of DDOS Attacks to the SDN Data Plane: A Security Perspective. 20th International Computer Conference on Wavelet Active Media Technology

and Information Processing (ICCWAMTIP) (pp. 1-5). IEEE. December 2023.

[21] Thirupathi, V., Sandeep, C. H., Kumar, N., and Kumar, P. P., "A comprehensive review on SDN architecture, applications and major benefits of SDN," International Journal of Advanced Science and Technology, vol. 28, no. 20, pp. 607-614, 2019.

AUTHORS

Sirijan Ra Na is currently pursuing a Master's degree in Computer Science and Engineering at the Government College of Technology, Coimbatore, Tamil Nadu, India. He completed his Bachelor's degree in Computer Science and Engineering at K.S.R. College of Engineering, Tiruchengode. His academic interests include Data Science, Intelligent Systems, and Machine Learning.

Miraclin Joyce Pamila J.C., is working as Professor in the Department of Computer Science and Engineering of Government College of Technology, Coimbatore, Tamil Nadu, India. She is a Life member of ISTE. She is a passionate Teacher and believes in life-long learning Her Research areas include Data Science, Machine Learning, and Artificial Intelligence. She has completed AICTE funded project on Intelligent Transportation System.