A Hybrid Deep Learning Model with Feature Selection for Intrusion Detection in Software-Defined Networks

[1]Dr.R.Rajani, [2]Ch. Aswini, [3]K.Vishnuhansika, [4]D. Charanya, [5]M. Himagna
[1]Professor , [2,3,4,5]UG Students, [1,2,3,4,5]Department of Computer Science & Engineering, Geethanjali Institute Of Science And Technology, Nellore, India

Abstract
Software-Defined Networking (SDN) offers a flexible and centralized approach to manage modern network infrastructures. However, the centralized nature of SDN also introduces new vulnerabilities, including Distributed Denial-of-Service (DDoS), web- based attacks, and User to Root (U2R) intrusions. These threats can compromise the SDN controller, leading to severe network disruptions and potential system failure. To mitigate such risks, Network Intrusion Detection Systems (NIDS) are essential. In this project, we propose a hybrid deep learning model that combines Convolutional Neural Networks (CNN) and Bidirectional Long Short- Term Memory (BiLSTM) networks to enhance intrusion detection accuracy and performance in SDN environments. The CNN layers are employed for automated feature extraction, while the BiLSTM layers capture complex sequential attack patterns. Additionally, a hybrid feature selection method is used to eliminate redundant and irrelevant features, thereby optimizing input quality and reducing training time. Finally, the model is evaluated on UNSW-NB15, NSL-KDD, and InSDN datasets for both binary and multi-class classification tasks. Experimental results demonstrate improved accuracy, generalization capability, and detection efficiency. The proposed approach strengthens SDN security by providing a robust and scalable intrusion detection framework.
Keywords: Deep learning model, Intrusion detection, SDN, U2R, DDOS, NSL-KDD

Introduction

In recent years, Software-Defined Networking (SDN) has emerged as a transformative architecture in modern network management due to its programmability, centralized control, and dynamic network configuration capabilities. However, this same flexibility introduces new security challenges, making SDNs an attractive target for cyberattacks. Traditional intrusion detection systems (IDS) often fall short in handling the complexity and volume of data generated in SDN environments.
To address these limitations, this project proposes a Hybrid Deep Learning Model with Feature Selection for effective intrusion detection in SDNs. The system integrates the strengths of multiple deep learning models—such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks—to accurately learn both spatial and temporal features of network traffic. Additionally, feature selection techniques are employed to reduce redundancy, improve model efficiency, and enhance detection performance.
The goal is to build an intelligent, adaptive, and real-time intrusion detection system that can identify known and unknown threats with high accuracy while minimizing false alarms. The proposed solution is tested using benchmark intrusion datasets and evaluated through various performance metrics to ensure its practical applicability in real SDN environments.

Motivation

As network infrastructures evolve, Software-Defined Networking (SDN) has become a key technology for managing modern, scalable, and pr
This challenge motivates the development of a hybrid deep learning model that combines the strengths of multiple neural architectures (e.g., CNN for spatial feature extraction and LSTM for temporal analysis). To

further optimize performance, feature selection techniques are integrated to reduce noise, improve learning speed, and focus the model on the most relevant traffic attributes. ogrammable networks. While SDNs offer several advantages—such as centralized control, dynamic configuration, and simplified management—they also introduce new security vulnerabilities due to their programmable nature and separation of control and data planes. Moreover, the increasing volume and complexity of network traffic demand more advanced detection techniques that can process large datasets efficiently and accurately.

Objective

The primary objective of this project is to develop a robust and efficient intrusion detection system (IDS) tailored for Software Defined Networks (SDNs) by leveraging a hybrid deep learning model integrated with feature selection techniques. SDNs, due to their centralized control and programmability, have transformed traditional network architectures but have also introduced new security vulnerabilities. This project aims to enhance the security of SDNs by accurately identifying malicious traffic and cyber threats through intelligent data-driven methods.

The hybrid deep learning model combines the strengths of multiple deep learning architectures, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), to effectively capture both spatial and temporal patterns in network traffic data. This hybrid approach allows for more comprehensive detection of complex attack behaviors that might be overlooked by single-model systems. In addition, the integration of feature selection algorithms ensures that only the most relevant and informative features are used for training the model, thereby reducing computational complexity and improving detection accuracy.

Ultimately, the goal of this project is to deliver a scalable, accurate, and real-time intrusion detection framework for SDNs. By combining advanced deep learning techniques with intelligent feature selection, the system is expected to outperform traditional IDS models in terms of detection rate, false alarm rate, and processing efficiency. This work contributes to the growing field of intelligent network security by offering a practical and effective solution for mitigating threats in modern programmable network environments.

Literature Review

In this chapter, we delve into several significant research works to build a better understanding of the techniques that have been proposed for Network Intrusion Detection Systems (NIDS), particularly within Software-Defined Networking (SDN) environments. All these methods share a common goal - to enhance the detection of network intrusions efficiently and accurately. As the famous quote goes, "To know the road ahead, ask those coming back," it becomes clear that learning from previous research is not just beneficial, but essential. This literature survey focuses on the most impactful works that relate to deep learning-based intrusion detection, especially hybrid models like CNN and BI-LSTM, offering insights into how they have influenced the direction of our proposed model.

Intrusion Detection Model of CNN-BiLSTM Algorithm Based on Mean Control

Authors: L. Zhang, J. Huang

In 2020, Zhang et al. proposed an advanced intrusion detection model that combines Convolutional Neural Networks (CNN) with Bidirectional Long Short-Term Memory (BiLSTM) networks to enhance the detection of cyber threats. The CNN component is used for extracting local spatial features from network traffic data, while the BiLSTM component captures long- range temporal dependencies in the data stream. To further improve the model's performance and stability, a mean control mechanism is introduced, which helps regulate the training process and avoid overfitting. The model was presented at the IEEE 11th International Conference on Software Engineering and Service Science (ICSESS) and demonstrated superior accuracy and robustness in intrusion detection tasks.

Anomaly Detection in Encrypted Internet Traffic Using Hybrid Deep Learning

Authors: T. Bakhshi and B. Ghita

In 2021, Bakhshi and Ghita tackled the challenge of intrusion detection in encrypted traffic by introducing a hybrid deep learning approach. Published in Security and Communication Networks, their work addresses the limitations of traditional intrusion detection systems, which often struggle with analyzing encrypted data.Instead of relying on payload inspection, their model utilizes flow-based and statistical features extracted from encrypted network traffic. The hybrid model combines both convolutional and recurrent neural networks to effectively learn from the data's spatial and temporal characteristics. The results showed that their method can accurately detect anomalies in encrypted traffic while preserving data confidentiality.

Network Intrusion Detection Combined Hybrid Sampling with Deep Hierarchical Network

Authors: K. Jiang, W. Wang

In 2020, Jiang and colleagues introduced a novel approach to network intrusion detection by integrating hybrid sampling techniques with a deep hierarchical neural network. The study, published in IEEE Access, addresses the common problem of class imbalance in network datasets by using a combination of oversampling and undersampling methods to balance the training data. The deep hierarchical network then processes the balanced data, enabling multi-level feature extraction for more precise detection of both common and rare attack types. The proposed method showed significant improvements in detection accuracy, precision, and recall.

An Efficient Flow-Based Multi-Level Hybrid Intrusion Detection System for Software- Defined Networks

Authors: M. Latah and L. Toker

In their 2020 paper published in CCF Transactions on Networking, Latah and Toker proposed a multi-level hybrid intrusion detection system tailored for Software-Defined Networks (SDNs). Their system uses flow-based analysis, which focuses on high-level traffic patterns rather than packet content, making it lightweight and scalable. The multilevel design combines signature-based detection (to quickly identify known threats) and anomaly-based detection (to discover unknown or evolving attacks). The system was evaluated on a range of SDN scenarios and demonstrated improved detection performance with reduced false alarm rates, making it a practical solution for dynamic and large-scale network environments

Proposed Model

In this project, we propose a hybrid deep learning model that leverages the strengths of Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) networks to build an intelligent and adaptive Network Intrusion Detection System (NIDS) for Software-Defined Networking (SDN) environments. CNNs are renowned for their powerful feature extraction capabilities, especially in capturing spatial correlations in input data. When applied to network traffic data, CNN can detect subtle patterns or anomalies in the feature space that may indicate malicious behavior. On the other hand, BiLSTM is highly effective at capturing temporal dependencies and bidirectional sequence information, which is vital for understanding the chronological flow of network activities. By integrating CNN and BiLSTM, the proposed system not only extracts complex spatial features from raw traffic data but also learns temporal dynamics to accurately detect both known and unknown attacks. To enhance the model's performance, a hybrid feature selection approach is employed reduce dimensionality, eliminate irrelevant features, and retain the most informative attributes.

This step ensures that the system remains efficient and robust, improving detection speed without compromising accuracy

This study examines the economic and technical impact of implementing a CNN - BiLSTM-based Network Intrusion Detection System (NIDS) in a Software-Defined Networking (SDN) environment. As organizations have limited budgets for research and development, careful planning is required to ensure that the project remains cost- effective and within allocated resources. The system was designed with economic efficiency in mind. Most of the tools and frameworks used such as TensorFlow, Keras, and Scikit-learn—are open-source, significantly reducing the software acquisition costs. Only hardware and some cloud-based services, if needed for scalable model training and deployment, incur additional expenses. The integration of CNN-BiLSTM hybrid architecture improves intrusion detection accuracy, reducing false positives and improving network security-ultimately justifying the system's development cost through enhanced threat mitigation.

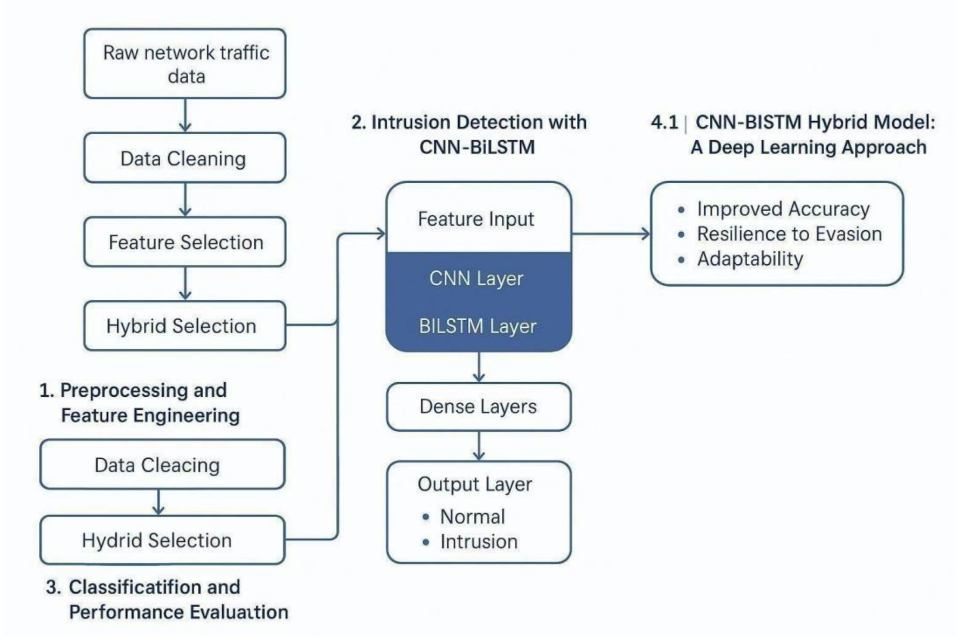


Fig.1. System Architecture

Preprocessing and Feature Engineering:

Input: Raw network traffic data collected from an SDN (Software-Defined Networking) environment using standard datasets (e.g., NSL-KDD, CICIDS2017).

Data Cleaning: Handles missing values, removes duplicates, and normalizes the features to ensure consistent data representation.

Feature Selection:

Filter-based Techniques: Employs statistical methods to identify relevant features based on correlation, information gain, etc.

Wrapper-based Techniques: Uses machine learning models to evaluate feature subsets and their impact on prediction performance.

Hybrid Selection: Combines filter and wrapper methods to select the most informative features and reduce dimensionality.

Intrusion Detection with CNN-BiLSTM:

Feature Input: The selected and preprocessed feature vectors are used as input.

CNN Layer (Convolutional Neural Network): Extracts local spatial patterns and correlations among features.

Captures the structure and hierarchy in the input data, acting as an effective feature extractor.

BiLSTM Layer (Bidirectional Long Short-Term Memory): Processes the output of the CNN layer in both forward and backward directions.

Captures temporal dependencies in network traffic patterns, improving detection of complex and evolving attacks.

Dense Layers: Used for final classification based on features learned from CNN and BiLSTM layers.

Classification and Performance Evaluation:

Output Layer: A softmax layer is used to classify traffic as normal or intrusion (multi-class for various attack types).

Evaluation Metrics:

1. **Accuracy:** Measures the percentage of correct predictions.
2. **Precision,Recall,F1-Score:**Used to evaluate model, performance especially in imbalanced classes.
3. **Confusion Matrix:** Visual tool to understand misclassifications

The hybrid CNN-BiLSTM model outperforms traditional machine learning and deep learning models in detecting various types of network
intrusions

CNN-BiLSTM Hybrid Model: A Deep Learning Approach:

CNN-BiLSTM is a hybrid deep learning architecture combining:

CNN (Convolutional Neural Network):

i. Efficiently captures spatial features from structured input data (feature vectors).
ii. Learns high-level representations of traffic features.

b. BiLSTM (Bidirectional Long Short-Term Memory):

i. Enhances the model's ability to capture temporal dynamics in both directions (past and future).
ii. Crucial for detecting time-dependent anomalies in network behavior

CNN-BiLSTM

CNN-BiLSTM:A Hybrid Deep Learning Model for Network Intrusion Detection

CNN-BiLSTM is a hybrid architecture that combines Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory networks (BiLSTM) for effective intrusion detection in Software-Defined Networks (SDNs).

CNN (Convolutional Neural Network): Used for automatic feature extraction from network traffic data. It captures local spatial patterns in input sequences, such as sudden spikes or anomalies.

BiLSTM (Bidirectional Long Short-Term Memory): Enhances temporal analysis by learning from both past and future time steps, capturing long-term dependencies in network traffic. This hybrid model improves detection accuracy by combining the strength of spatial and emporal feature learning.

Data Flow Diagram :

This diagram shows the basic steps of a machine learning classification process. First, data preprocessing is done using techniques like one-hot encoding and minmax scaling to prepare the data. Then, the data is split into 80% for training and 20% for testing. Various classification models like Multilayer Perceptron, Support Vector Machine (SVM), and K-Nearest Neighbors are used to train the data. After training, the model's performance is evaluated using metrics such as accuracy, precision, recall, and F1 score. Finally, the results are obtained based on these evaluations.

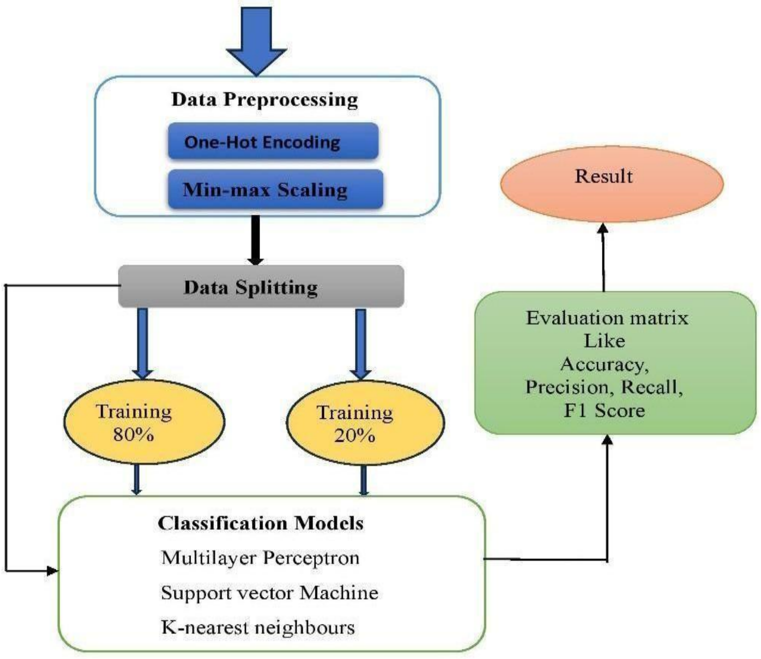


Fig.2. Data Flow Diagram

SYSTEM IMPLEMENTATION

7.1 SYSTEM MODULES

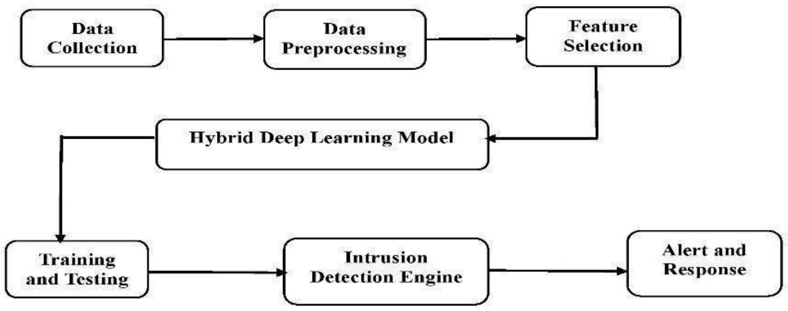


Fig.3. System Modules

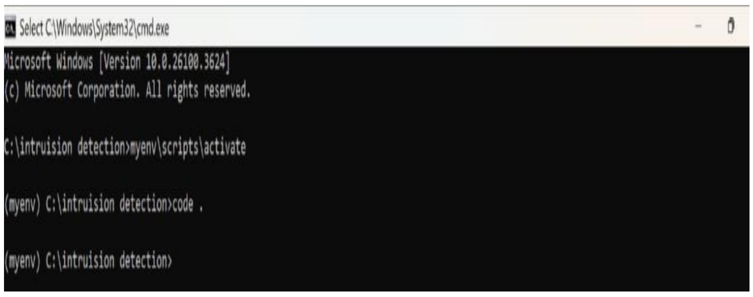
Data Collection Module:

Collects network traffic data (e.g., flow records) from SDN Environments. Can use sources like KDD Cup, NSL-KDD, CICIDS, or real-time SDN controller logs.

Results & Analysis

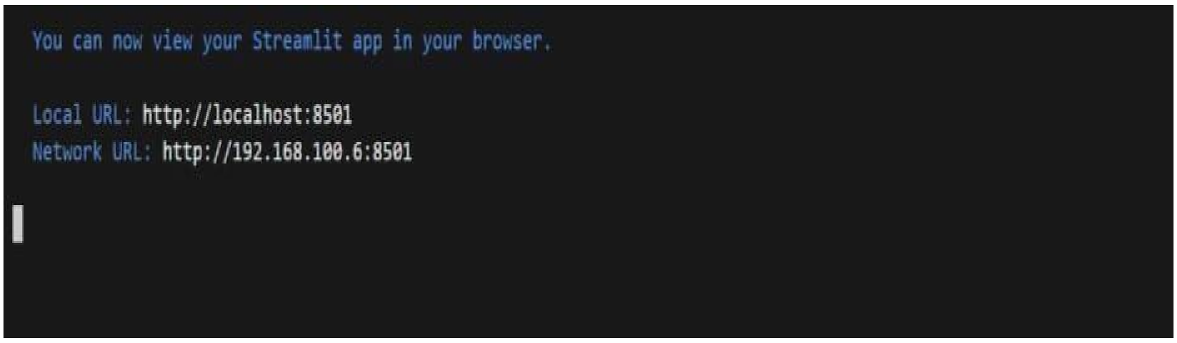
The execution process of the project is shown below Step–

1: Setting up the environment



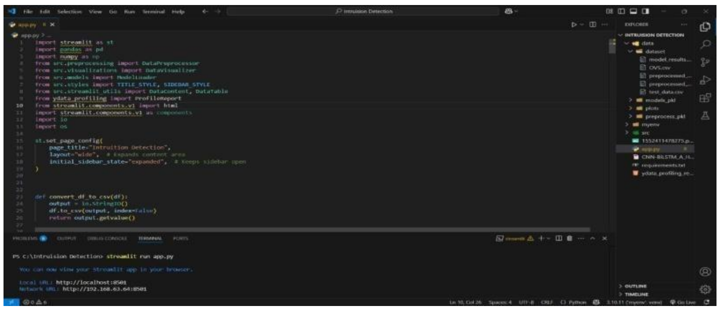
```
Select C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.26100.3624]
(c) Microsoft Corporation. All rights reserved.

C:\intruision detection>myenv\scripts\activate

(myenv) C:\intruision detection>code .

(myenv) C:\intruision detection>
```

Step–2: Running the application and accessing the Web Interface using the URL



```
You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.100.6:8501
```

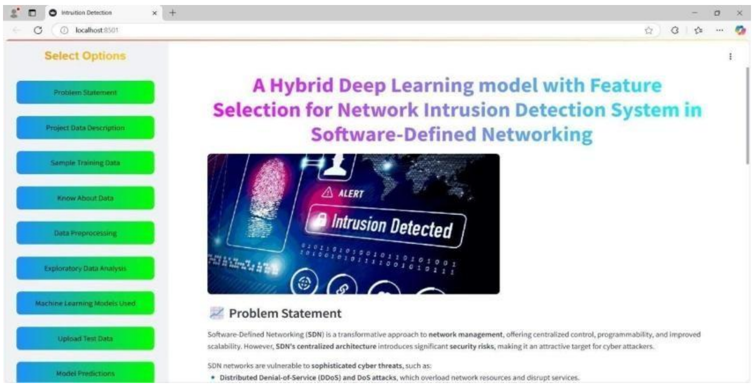
Commands that are used for running the project(incmdprompt): To Activate virtual environment:
Myenv\scripts\activate To Run the application:
Streamlitrunapp.py To stop the application:
Ctrl + C
Step–3:Runstream lit



```
import streamlit as st
import pandas as pd
import numpy as np
from src.preprocessing import DataPreprocessor
from src.visualizations import DataVisualizer
from src.models import ModelLoader
from src.styles import TITLE_STYLE, SIDEBAR_STYLE
from src.streamlit_utils import DataContent, DataTable
from ydata_profiling import ProfileReport
from streamlit.components.v1 import html
import streamlit.components.v1 as components
import io
import os

st.set_page_config(
    page_title="Intruition Detection",
    layout="wide",  # Expands content area
    initial_sidebar_state="expanded",  # Keeps sidebar open
)


def convert_df_to_csv(df):
    output = io.StringIO()
    df.to_csv(output, index=False)
    return output.getvalue()
```

```
PS C:\Intrulsion Detection> streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.63.64:8501
```

Step 4 : Home page of the project



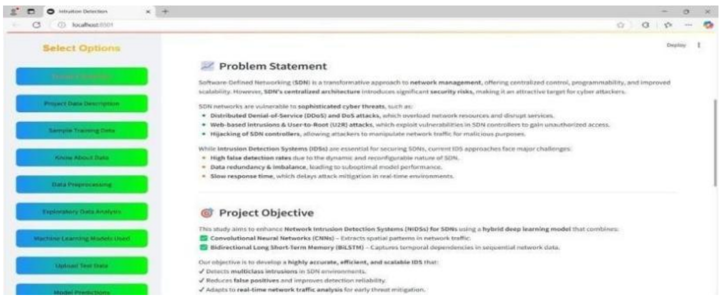
A Hybrid Deep Learning model with Feature Selection for Network Intrusion Detection System in Software-Defined Networking

Problem Statement

Software-Defined Networking (**SDN**) is a transformative approach to **network management**, offering centralized control, programmability, and improved scalability. However, **SDN's centralized architecture** introduces significant **security risks**, making it an attractive target for cyber attackers.

SDN networks are vulnerable to **sophisticated cyber threats**, such as:
- **Distributed Denial-of-Service (DDoS) and DoS attacks**, which overload network resources and disrupt services.

Step–5: problem statement



Problem Statement

Software-Defined Networking (**SDN**) is a transformative approach to **network management**, offering centralized control, programmability, and improved scalability. However, **SDN's centralized architecture** introduces significant **security risks**, making it an attractive target for cyber attackers.

SDN networks are vulnerable to **sophisticated cyber threats**, such as:
- **Distributed Denial-of-Service (DDoS) and DoS attacks**, which overload network resources and disrupt services.
- **Web-based intrusions & User-to-Root (U2R) attacks**, which exploit vulnerabilities in SDN controllers to gain unauthorized access.
- **Hijacking of SDN controllers**, allowing attackers to manipulate network traffic for malicious purposes.

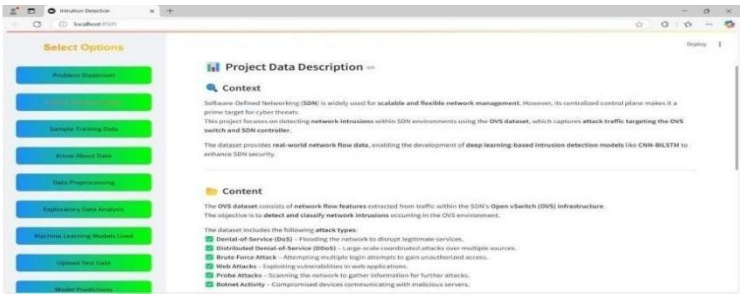
While **Intrusion Detection Systems (IDSs)** are essential for securing SDNs, current IDS approaches face major challenges:
- **High false detection rates** due to the dynamic and reconfigurable nature of SDN.
- **Data redundancy & imbalance**, leading to suboptimal model performance.
- **Slow response time**, which delays attack mitigation in real-time environments.

Project Objective

This study aims to enhance **Network Intrusion Detection Systems (NIDSs) for SDNs** using a **hybrid deep learning model** that combines:
✅ **Convolutional Neural Networks (CNNs)** – Extracts spatial patterns in network traffic.
✅ **Bidirectional Long Short-Term Memory (BiLSTM)** – Captures temporal dependencies in sequential network data.

Our objective is to develop a **highly accurate, efficient, and scalable IDS** that:
✓ Detects **multiclass intrusions** in SDN environments.
✓ Reduces **false positives** and improves detection reliability.
✓ Adapts to **real-time network traffic analysis** for early threat mitigation.

Step–6: project data description



📊 Project Data Description

🔍 Context

Software-Defined Networking (**SDN**) is widely used for **scalable and flexible network management**. However, its centralized control plane makes it a prime target for cyber threats.
This project focuses on detecting **network intrusions** within SDN environments using the **OVS dataset**, which captures **attack traffic targeting the OVS switch and SDN controller**.

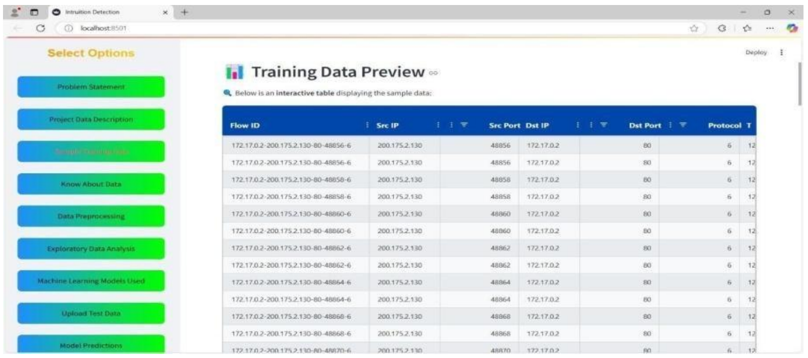
The dataset provides **real-world network flow data**, enabling the development of **deep learning-based intrusion detection models** like **CNN-BILSTM** to enhance SDN security.

📁 Content

The **OVS dataset** consists of **network flow features** extracted from traffic within the SDN's **Open vSwitch (OVS) infrastructure**.
The objective is to **detect and classify network intrusions** occurring in the OVS environment.

The dataset includes the following **attack types**:
✅ **Denial-of-Service (DoS)** – Flooding the network to disrupt legitimate services.
✅ **Distributed Denial-of-Service (DDoS)** – Large-scale coordinated attacks over multiple sources.
✅ **Brute Force Attack** – Attempting multiple login attempts to gain unauthorized access.
✅ **Web Attacks** – Exploiting vulnerabilities in web applications.
✅ **Probe Attacks** – Scanning the network to gather information for further attacks.
✅ **Botnet Activity** – Compromised devices communicating with malicious servers.

Step–7: Sample training data



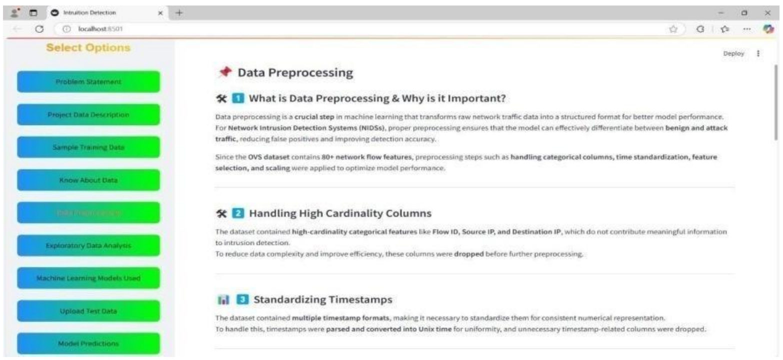
Training Data Preview

Below is an **interactive table** displaying the sample data:

Flow ID	Src IP	Src Port	Dst IP	Dst Port	Protocol	T
172.17.0.2-200.175.2.130-80-48856-6	200.175.2.130	48856	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48856-6	200.175.2.130	48856	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48858-6	200.175.2.130	48858	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48858-6	200.175.2.130	48858	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48860-6	200.175.2.130	48860	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48860-6	200.175.2.130	48860	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48862-6	200.175.2.130	48862	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48862-6	200.175.2.130	48862	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48864-6	200.175.2.130	48864	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48864-6	200.175.2.130	48864	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48868-6	200.175.2.130	48868	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48868-6	200.175.2.130	48868	172.17.0.2	80	6	12
172.17.0.2-200.175.2.130-80-48870-6	200.175.2.130	48870	172.17.0.2	80	6	12

Step–8: Know About Data

Step–9: Data Preprocessing Steps



Data Preprocessing

1 What is Data Preprocessing & Why is it Important?

Data preprocessing is a **crucial step** in machine learning that transforms raw network traffic data into a structured format for better model performance. For **Network Intrusion Detection Systems (NIDSs)**, proper preprocessing ensures that the model can effectively differentiate between **benign and attack traffic**, reducing false positives and improving detection accuracy.

Since the **OVS dataset** contains **80+ network flow features**, preprocessing steps such as **handling categorical columns, time standardization, feature selection, and scaling** were applied to optimize model performance.

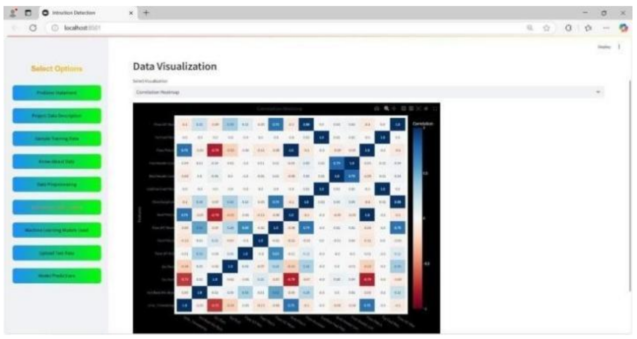
2 Handling High Cardinality Columns

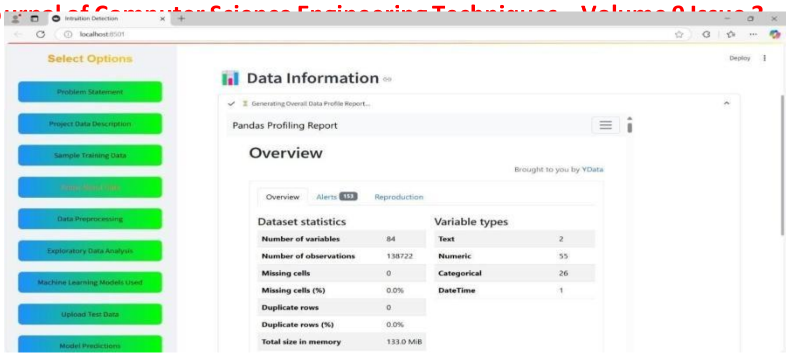
The dataset contained **high-cardinality categorical features** like **Flow ID, Source IP, and Destination IP**, which do not contribute meaningful information to intrusion detection.
To reduce data complexity and improve efficiency, these columns were **dropped** before further preprocessing.

3 Standardizing Timestamps

The dataset contained **multiple timestamp formats**, making it necessary to standardize them for consistent numerical representation.
To handle this, timestamps were **parsed and converted into Unix time** for uniformity, and unnecessary timestamp-related columns were dropped.

Step–10: Exploratory Data Analysis





Data Information

Generating Overall Data Profile Report...

Pandas Profiling Report

Overview

Brought to you by YData

Overview | Alerts 153 | Reproduction

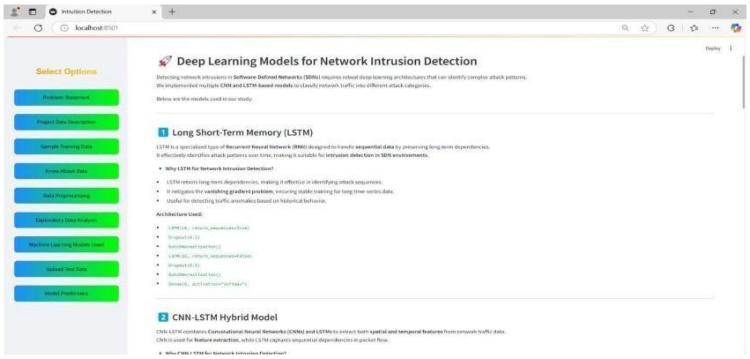
Dataset statistics

Number of variables	84
Number of observations	138722
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	133.0 MiB

Variable types

Text	2
Numeric	55
Categorical	26
DateTime	1

Step–11: Machine learning Models Used



Deep Learning Models for Network Intrusion Detection

Detecting network intrusions in **Software-Defined Networks (SDNs)** requires robust deep learning architectures that can identify complex attack patterns.
We implemented multiple **CNN and LSTM-based models** to classify network traffic into different attack categories.

Below are the models used in our study.

1 Long Short-Term Memory (LSTM)

LSTM is a specialized type of **Recurrent Neural Network (RNN)** designed to handle **sequential data** by preserving long-term dependencies.
It effectively identifies attack patterns over time, making it suitable for **intrusion detection in SDN environments**.

- **Why LSTM for Network Intrusion Detection?**

- LSTM retains long-term dependencies, making it effective in identifying attack sequences.
- It mitigates the **vanishing gradient problem**, ensuring stable training for long time-series data.
- Useful for detecting traffic anomalies based on historical behavior.

Architecture Used:

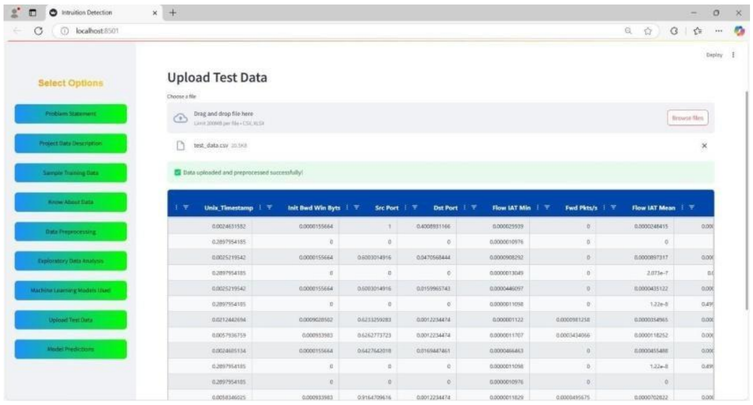
- `LSTM(16, return_sequences=True)`
- `Dropout(0.5)`
- `BatchNormalization()`
- `LSTM(32, return_sequences=False)`
- `Dropout(0.5)`
- `BatchNormalization()`
- `Dense(6, activation='softmax')`

2 CNN-LSTM Hybrid Model

CNN-LSTM combines **Convolutional Neural Networks (CNNs) and LSTMs** to extract both **spatial and temporal features** from network traffic data.
CNN is used for **feature extraction**, while LSTM captures sequential dependencies in packet flow.

- **Why CNN-LSTM for Network Intrusion Detection?**

Step–12:Upload Test Data



Upload Test Data

Choose a file

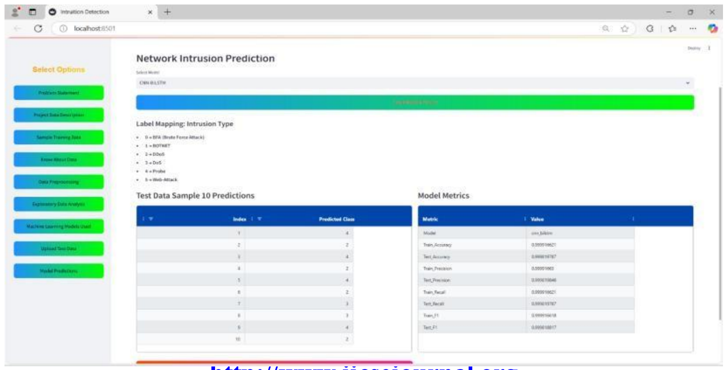
Drag and drop file here
Limit 200MB per file • CSV, XLSX

test_data.csv 20.5KB

Data uploaded and preprocessed successfully!

Unix_Timestamp	Init Bwd Win Byts	Src Port	Dst Port	Flow IAT Min	Fwd Pkts/s	Flow IAT Mean
0.0024631582	0.0000155664	1	0.4008931166	0.000025939	0	0.0000248415
0.2897954185	0	0	0	0.0000010976	0	0
0.0025219542	0.0000155664	0.6003014916	0.0470568444	0.0000908292	0	0.0000897317
0.2897954185	0	0	0	0.0000013049	0	2.073e-7
0.0025219542	0.0000155664	0.6003014916	0.0159965743	0.0000446097	0	0.0000435122
0.2897954185	0	0	0	0.0000011098	0	1.22e-8
0.0212442694	0.0009028502	0.6233259283	0.0012234474	0.000001122	0.0000981258	0.0000354965
0.0057936759	0.000933983	0.6262773723	0.0012234474	0.0000011707	0.0003434066	0.0000118252
0.0024605134	0.0000155664	0.6427642018	0.0169447461	0.0000466463	0	0.0000455488
0.2897954185	0	0	0	0.0000011098	0	1.22e-8
0.2897954185	0	0	0	0.0000010976	0	0
0.0058346025	0.000933983	0.9164709616	0.0012234474	0.0000011829	0.0000495675	0.0000702822

Step–13: Model Predictions



Network Intrusion Prediction

Select Model: CNN-BiLSTM

Label Mapping: Intrusion Type

- 0 → BFA (Brute Force Attack)
- 1 → BOTNET
- 2 → DDoS
- 3 → DoS
- 4 → Probe
- 5 → Web-Attack

Test Data Sample 10 Predictions

Index	Predicted Class
1	4
2	2
3	4
4	2
5	4
6	2
7	3
8	3
9	4
10	2

Model Metrics

Metric	Value
Model	cnn_bilstm
Train_Accuracy	0.999916621
Test_Accuracy	0.999919787
Train_Precision	0.99991663
Test_Precision	0.999919846
Train_Recall	0.999916621
Test_Recall	0.999919787
Train_F1	0.999916618
Test_F1	0.999918817

Conclusion

The CNN-BiLSTM model improves network intrusion detection in SDN by effectively identifying complex attack patterns. By integrating CNN and LSTM architectures with an attention mechanism, the system improves the detection of both known and unknown cyber threats. Additionally, the hybrid feature selection process (combining filter-based, wrapper-based, and auto-encoder techniques) optimizes feature extraction, reducing computational overhead while maintaining high accuracy. This approach enables real time, scalable, and adaptive intrusion detection within SDN environments by leveraging the centralized control plane to respond dynamically to security threats. Compared to traditional methods, the proposed system demonstrates higher detection accuracy, lower false positive rates, and improved adversarial robustness.

Future Scope

The proposed hybrid deep learning model with feature selection for intrusion detection in Software Defined Networks (SDNs) opens several avenues for future research. One key direction is the real-time implementation and optimization of the model to ensure low latency and high throughput in dynamic SDN environments. Enhancing the model's adaptability to detect emerging and zero-day attacks through online or continual learning is also crucial. Moreover, integrating explainable AI techniques can improve transparency and trust in the system's decisions. Future work can explore cross-domain applicability to test the model's robustness across different SDN architectures and traffic patterns. Additionally, improving feature selection methods using advanced optimization techniques could further enhance detection accuracy while reducing computational overhead, making the model more suitable for deployment in resource-constrained environments like edge or fog computing.

References

1. P. Choobdar, M. Naderan, and M. Naderan, ''Detection and multi-class classification of intrusion in software defined networks using stacked autoencoders and CICIDS2017 dataset,'' Wireless Pers. Commun., vol. 123, no. 1, pp. 437–471, Mar. 2022.
2. M. S. ElSayed, N.-A. Le-Khac, M. A. Albahar, and A. Jurcut, ''A novel hybrid model for intrusion detection systems in SDNs based on CNN and anew regularization technique,'' J. Netw. Comput. Appl., vol. 191, Oct. 2021, Art. no. 103160.
3. T. Bakhshi and B. Ghita, ''Anomaly detection in encrypted internet traffic using hybrid deep learning,'' Secur. Commun. Netw., vol. 2021, pp. 1–16,sep. 2021.
4. M. Latah and L. Toker, ''An efficient flow-based multi-level hybrid intrusion detection system for software-defined networks,'' CCF Trans. Netwvol. 3, nos. 3–4, pp. 261–271, Dec. 2020, doi: 10.1007/s42045020-00040-z.
5. L. Zhang, J. Huang, Y. Zhang, and G. Zhang, ''Intrusion detection model of CNN- BiLSTM algorithm based on mean control,'' in Proc. IEEE 11th Int. Conf. Softw. Eng. Service Sci. (ICSESS), Oct. 2020, pp. 22–27.
6. K. Jiang, W. Wang, A. Wang, and H. Wu, ''Network intrusion detection combined hybrid sampling with deep hierarchical network,'' IEEE Access,vol. 8, pp. 32464– 32476, 2020.
7. M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, ''InSDN: A novel SDN intrusion dataset,'' IEEE Access, vol. 8, pp. 165263–165284, 2020, doi: 10.1109/ACCESS.2020.3022633.
8. N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, ''Survey onSDN based network intrusion detection system using machine learning approaches,'' Peer- Peer Netw. Appl., vol. 12, no. 2, pp. 493–501, Mar. 2019, doi: 10.1007/s12083-017-0630-0.