A Deep Learning Based Intrusion Detection System for IIoT Networks

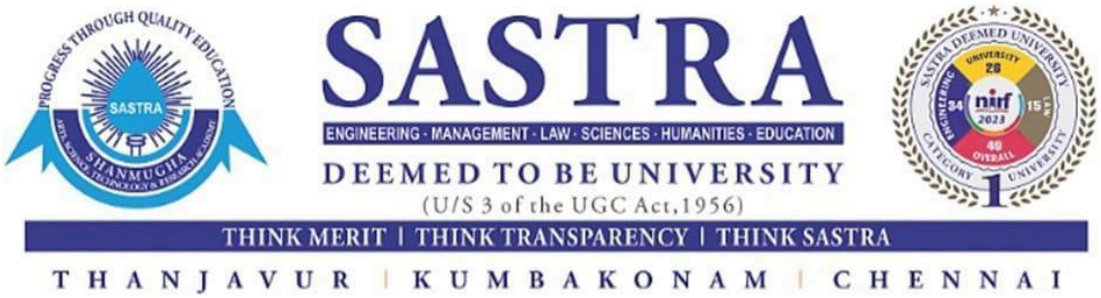
Report submitted to the SASTRA Deemed to be University
as the requirement for the course

CSE300 - MINI PROJECT
B. Tech Computer Science and Engineering

Submitted by

Abinash S (Reg. No.: 225003003)
Srivatsan N (Reg. No.: 225003156)
Hemachandran S K (Reg. No.: 225003050)

May 2024



Department of Computer Science and Engineering
SRINIVASA RAMANUJAN CENTRE
Kumbakonam, Tamil Nadu, INDIA - 612 001

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act, 1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

THANJAVUR | KUMBAKONAM | CHENNAI

Department of Computer Science and Engineering

SRINIVASA RAMANUJAN CENTRE

Kumbakonam, Tamil Nadu, INDIA - 612 001

May 2024

Bonafide Certificate

This is to certify that the report titled **"A Deep Learning Based Intrusion Detection System for IIoT Networks"** submitted as *in partial fulfilment of the requirements for the award of the degree of B.Tech.,* Computer Science and Engineering to the SASTRA Deemed to be University, is a bonafide record of the work done by **Mr. Abinash S (225003003), Mr. Srivatsan N (225003156), Mr. Hemachandran S K (225003050)** during the sixth semester of the academic year 2023-2024, in the Srinivasa Ramanujan Centre, under my supervision. This project report has not formed the basis for the award of any degree, diploma, associateship, fellowship or other similar title to any candidate of any University.

Signature of Project Supervisor : 3/5/24.

Name with Affiliation : Dr. Priyanga S
AP-II / CSE / SRC / SASTRA

Date : 03/05/2024

Mini Project *Viva voce* held on 06/05/2024

6/5

Examiner 1

6/5/24

Examiner 2

Acknowledgement

I would like to thank our Honorable Chancellor **Prof. R. Sethuraman** for providing us with an opportunity and the necessary infrastructure for carrying out this project as a part of our curriculum.

I would like to thank our Honorable Vice-Chancellor **Dr. S. Vaidhyasubramaniam and Dr. S. Swaminathan**, Dean, Planning & Development, for the encouragement and strategic support at every step of our college life.

I extend our sincere thanks to **Dr. R. Chandramouli,** Registrar, SASTRA Deemed to be University for providing the opportunity to pursue this project.

I extend our heartfelt thanks to **Dr. V. Ramaswamy,** The Dean and **Dr. A. Alli Rani,** Associate Dean, Srinivasa Ramanujan Centre for their constant support and suggestions when required without any reservations.

I express my sincere gratitude to **Dr. V. Kalaichelvi,** Associate Professor, Department of Computer Science and Engineering for her unconditional support and encouragement for completing the project.

My guide **Dr. S. Priyanga,** Assistant Professor, Department of Computer Science & Engineering, Srinivasa Ramanujan Centre was the driving force behind this whole idea from the start. Her deep insight in the field and invaluable suggestions helped me in making progress throughout our project work.

I also thank the Project Coordinator **Dr. S. Priyanga,** Assistant Professor, Department of Computer Science & Engineering, Srinivasa Ramanujan Centre for her ever-encouraging spirit and meticulous guidance for the completion of the project. We also thank the panel members for their valuable comments and insights which made this project better.

I would like to extend our gratitude to all the teaching and non-teaching faculties of the Srinivasa Ramanujan Centre who have either directly or indirectly helped us in the completion of the project. I gratefully acknowledge all the contributions and encouragement from my family and friends resulting in the successful completion of this project. I thank you all for providing me with an opportunity to showcase my skills through the project.

LIST OF FIGURES

ABBREVIATIONS

CNN	Convolution Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
ReLU	Rectified Linear Activation Function
IIoT	Industrial Internet of Things
ICS	Industrial Control System
IDS	Intrusion Detection System
IoT	Internet of Things
CPS	Cyber-Physical Systems
CI	Critical Infrastructure
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
DoS	Denial of Service
API	Application Programming Interface
UNSW	University of New South Wales

ABSTRACT

In recent years, the advancements in the network and cloud technologies have led to the growth of the Internet of Things (IoT) in industrial sectors. The Industrial Internet of Things (IIoT) comprises many connected devices and service data. IIoT networks are inherently vulnerable to cyberattacks as they depend on the associated networks for data exchange and communication. Therefore, it accentuates the necessity for ensuring the security of the IIoT systems for uninterrupted services. Intrusion Detection Systems (IDS) are widely adopted as a promising solution to detect cyber threats and attacks on the Industrial Control Systems (ICS) and IIoT networks. The existing IDS approaches lack generalisation, and involve misclassification errors and high false alarm rates. Considering these issues, a "Hybrid CNN + LSTM", deep learning-based IDS is designed to identify malicious activities from IIoT networks. The proposed "CNN + LSTM" employs a Convolution Neural Network (CNN) to identify the attack events and regular operations. The experiments are carried out using a benchmark datasets, UNSW-NB15 and the outcomes demonstrate the dominance of the proposed approach by achieving higher accuracy for the datasets respectively.

KEYWORDS: Intrusion Detection System, Convolutional Neural Network, Internet of Things, Long Short-Term Memory

TABLE OF CONTENTS

CHAPTER 1

SUMMARY OF THE BASE PAPER

Title: A Hybrid CNN + LSTM based Intrusion Detection System for Industrial IoT Networks

Journal Name: Engineering Science and Technology, an International Journal (JESTECH)

Author: Hakan Can Altunay, Zafer Albayrak

Published Date: 2022

1.1 SUMMARY

- The main objective of the paper is to observe the number of cyber-attacks on the IIoT networks increases day by day, an Intrusion Detection System for IIoT networks is proposed using Deep Learning model
- The authors proposed Convolutional Neural Network, Long Short-Term Memory and CNN-LSTM hybrid model that will detect and classify intrusions in IIoT networks by analyzing patterns and extract the features from the data
- The model is trained to identify different types of attacks and anomalies in the network traffic data, improves detection rate
- The proposed models for IDS were tested on complex datasets where there are large amounts of data and high numbers of attributes
- UNSW-NB15 dataset is used to determined the data, namely, normal and abnormal and compared with existing deep learning models
- The performance of the various proposed models were evaluated using UNSW-NB15 binary class and multi-class classification procedure and results are observed
- A comparison was made between CNN, LSTM, and CNN+LSTM methods. It is observed that the proposed approach has outperformed than the existing approaches in terms of accuracy and detection rate

1.2 BACKGROUND

IoT incorporates the digital and physical attributes to optimize the regular tasks involved in Cyber-Physical Systems (CPS), smart grids, smart cities and IIoT [1]. The IIoT was introduced to improve the communication between industrial and consumer-based applications. ICS is among the Critical Infrastructures (CIs) that highly rely on IoT networks and aim to increase the profitability and production of Industry 4.0. The complexity of ICS is evident in the distribution of industrial services and the adoption of industrialized techniques, including in manufacturing industries, water treatment plants, gas pipelines, and power plants [2].

The role of cybersecurity is more significant as the contribution of IIoT makes a greater impact on the growth of business operations. However, the escalating number of devices involved in IIoT networks raises concerns about safeguarding ICS, becoming a

serious threat to business operations [3] and creating complex attack surfaces for control and network elements in CPS networks. The existing security mechanisms like authentication systems, firewalls, and IDS are inefficient against advanced cyber attacks like Flame [4], Duqu [5], and seismic attacks [6] that exploit ICS networks due to the varying protocols [7].

In the emerging digital era, the researchers focused on developing the IDS that is capable of protecting the intricate attack scenarios [8]. Most of the existing approaches rely on static rules and predefined behaviours which are limited to recognising only the known attack scenario. In addition, most of the IDSs are tailored for generic communication protocols like TCP and UDP, neglecting control protocols like Modbus and DNP3 [8]. These limitations make the IDS weaker and exhibit a low detection rate due to extensive security features, overfitting, and imbalanced data. The above-mentioned scenarios insist the research medium provide a robust attack detection approach to protect IIoT-based networks against cyber-attacks, ensuring reliability. Extracting the relevant features from the CPS network and control data remains a significant challenge to enhance detection accuracy and reduce false alarms [9].

1.2.1 CONVOLUTIONAL NEURAL NETWORK

The CNN is a type of feed-forward neural network designed to extract hierarchical and spatial information from data. In the context of the UNSW-NB15 dataset, CNN is employed to analyze and classify the different attacks. During training, the model learns from input data by defining a set of filter values that enable it to recognize patterns and features within the input data. These filters systematically scan the input data, extracting relevant information at different layers of the network. As the input data passes through the layers, the network adjusts its weights through backpropagation, minimizing errors and refining its ability to classify network traffic accurately. By using convolutional filters to map parts of the input data with varying window sizes and strides, CNN efficiently captures features within the input data. This hierarchical feature extraction process makes CNN well-suited for tasks such as classifying network traffic into different categories based on its characteristics.

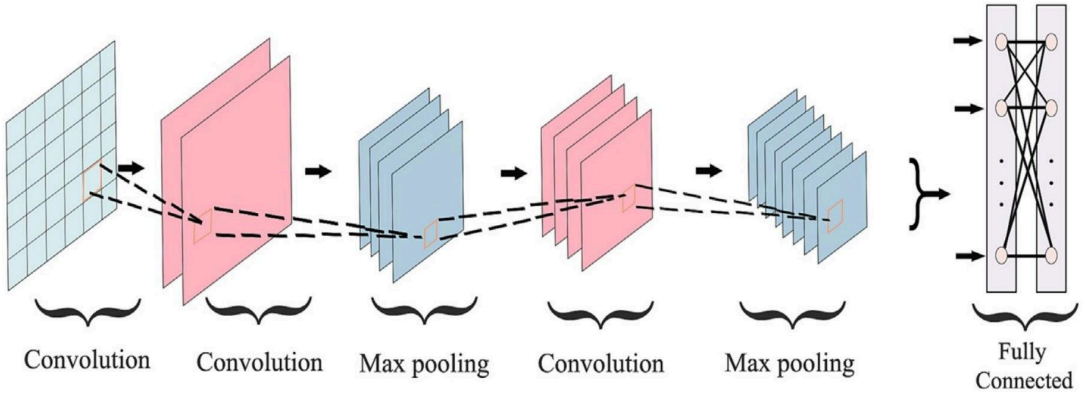


Fig. 1.1 Architecture of CNN

1.2.2 LONG SHORT-TERM MEMORY

The LSTM network is a type of Recurrent Neural network (RNN) designed to process and classify sequential data, such as time series or natural language text. In the context of the UNSW-NB15 dataset, LSTM networks are utilized to analyze and classify various attacks. During training, the LSTM model learns from input sequences by dynamically updating its memory cells, which enables it to capture long-term dependencies and temporal patterns within the data. Unlike traditional RNNs, LSTM networks incorporate gated mechanisms that regulate the flow of information through the network, allowing them to retain information over extended time periods and mitigate the vanishing gradient problem. The LSTM network operates by processing input sequences step by step, with each step updating its internal state based on the current input and the previous state. This process enables the network to learn intricate patterns and relationships within the sequential data. Through backpropagation, the network adjusts its parameters, including the weights and biases of the gates, to minimize errors and optimize its ability to classify network traffic accurately.

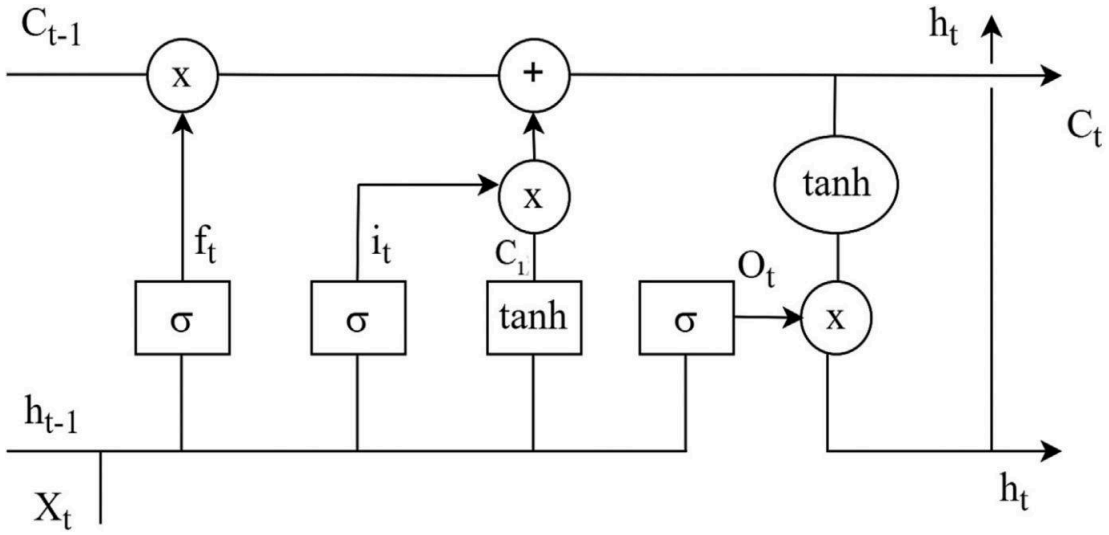


Fig 1.2 Architecture of Long Short-Term Memory

1.2.3 HYBRID CNN+LSTM MODEL

A hybrid CNN-LSTM model combines the strengths of Convolutional Neural Networks and Long Short-Term Memory networks to analyze and classify sequential data such as network traffic, as seen in the context of the UNSW-NB15 dataset. This model architecture integrates the hierarchical feature extraction capabilities of CNNs with the ability of LSTMs to capture long-term dependencies and temporal patterns within sequential data. During training, the hybrid model leverages CNN layers to extract spatial features from the input data, such as patterns and structures within the network traffic. These CNN layers consist of convolutional filters that systematically scan the input data, capturing relevant information at different levels of abstraction. The output of the CNN layers is then passed to the LSTM layers, which process the sequential data over time steps. Within the LSTM layers, the model dynamically updates its memory cells based on the sequential input, allowing it to

capture both short-term and long-term dependencies within the data. The gated mechanisms within the LSTM architecture regulate the flow of information, enabling the model to retain important contextual information over extended periods. Through the joint training of both CNN-based and LSTM-based components, the hybrid model learns to effectively classify network traffic into different categories based on its characteristics. As the input data passes through the layers, the model adjusts its parameters through backpropagation, optimizing its ability to accurately classify various types of network attacks.

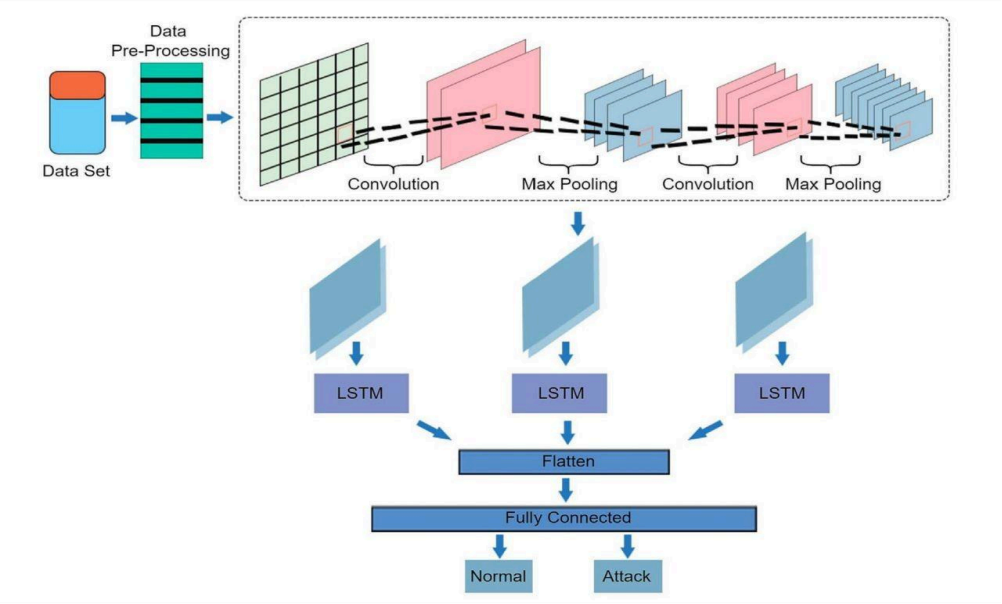


Fig 1.3 Architecture of Hybrid CNN+LSTM

1.3 PROBLEM DEFINITION

- To develop a deep learning-based Intrusion Detection System for enhancing the security of IIoT
- To implement an efficient preprocessing technique for identifying the relevant features of IIoT data
- To develop a deep learning based classification approach for achieving higher Detection Rate and improved classification accuracy

1.4 DATASET DESCRIPTION

This dataset was developed by the University of New South Wales (UNSW) Network Security and Cybercrime group in Australia. The dataset consists of network traffic which was collected over nine months. The dataset is also tested under attack scenarios that involve Denial of Service (DoS), probing, and penetration vulnerabilities. This is an obtainable dataset that has 1,75,341 records and 49 attributes with 2 classes normal and attack for binary classification and 9 classes namely, Normal, Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, Worms for multi class classification.

1.5 PROPOSED METHODOLOGY

The proposed methodology for Intrusion Detection System using the UNSW-NB15 dataset with a hybrid CNN+LSTM model begins with a comprehensive exploration of the structure and characteristics of the dataset to understand its underlying features and the diverse array of attacks it encompasses. Subsequently, data preprocessing is conducted, entailing the removal of extraneous features, treatment of missing values, standardization of feature scales, and encoding of categorical variables. Following this, the architecture of the hybrid model is designed, incorporating CNN layers for spatial feature extraction and LSTM layers for capturing temporal dependencies within sequences. By integrating CNN and LSTM components, the model effectively learns spatial and temporal patterns concurrently, thereby enhancing its efficiency in detecting intrusions. The dataset is then partitioned into training, validation, and testing subsets, facilitating the training and evaluation phases. During training, the hybrid CNN+LSTM model is trained on the training subset utilizing suitable optimization algorithms and loss functions, with the validation subset utilized for hyperparameter tuning to mitigate overfitting. Subsequently, the model's performance is rigorously assessed on the testing subset using standard evaluation metrics such as accuracy, precision, recall and F1-score. The results obtained are subjected to comprehensive analysis, including comparison with baseline models or existing methodologies, elucidation of strengths and limitations, and identification of potential avenues for further refinement. Finally, the implications of the findings are discussed within the context of Intrusion Detection and cybersecurity, offering insights into future research directions in the field. This methodological framework provides a systematic approach for the development and evaluation of a hybrid CNN+LSTM model for intrusion detection, facilitating robust experimentation and meaningful analysis for scholarly inquiry.

1.6 TECHNOLOGIES USED

To implement this project some of the technologies used are described below

TensorFlow: This open source library is employed for data flow programming. Is instrumental in constructing and training machine learning models. It offers a set of tools, libraries and resources that aid in the creation of machine learning and deep learning models. For this project Keras was utilized – a user high level API that simplifies the process of building deep learning models with minimal coding.

There are different Python libraries those are used in Deep Learning and some of them used are:

1. **NumPy:** A package for computations in Python NumPy supports multi dimensional arrays and includes a range of mathematical functions for array operations.
2. **Pandas:** This library introduces data structures such as DataFrame to manage data effectively. It also provides features for data cleansing, exploration and transformation.
3. **Matplotlib:** Known as a plotting library in Python, Matplotlib enables the creation of static, interactive and animated visualizations. It encompasses a variety of plotting functions designed to visualize both data and model performance metrics

4. **Scikit-learn:** This machine learning library offers effective tools for tasks, like data mining and analysis.

The machine used for this project is Google Collab, Jupyter Notebook and laptop with following specifications:

- Processor:Processor: 11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz, 2995 Mhz, 2 Core(s), 4 Logical Processor(s)
- Random Access Memory(RAM): 4GB
- Operating System: Windows 11 x64-based PC

1.7 FUNCTIONS USED IN SOURCE CODE

1.7.1 Conv1D

A 1D convolutional neural network is a type of neural network architecture commonly used for analyzing sequential data, such as time series or sequences of text. It applies convolutional operations along one dimension of the input data, systematically scanning and extracting local patterns or features. This allows the network to capture hierarchical representations of the input, learning to recognize both low-level and high-level features.

1.7.2 Dense

Dense layer, also known as a fully connected layer, connects every neuron in one layer to every neuron in the next layer. This dense connectivity allows the layer to learn complex relationships between features in the input data by performing a linear transformation followed by a non-linear activation function.

1.7.3 Dropout

Dropout is a regularization technique utilized in neural networks to prevent overfitting. During training, randomly decided neurons are neglected or "dropped out" with a sure chance. This helps to save the co-model of neurons and improves the generalization capability of the model.

1.7.4 LSTM

The LSTM module is combined with a Conv1D layer to implement a Hybrid CNN-LSTM model.

CHAPTER 2

MERITS AND DEMERITS OF BASE PAPER

2.1 RELATED WORKS

[10] has proposed a lightweight power IIoT gateway for intrusion detection that focuses on providing solutions for both signatures and anomalies. The designed framework is not robust against attack variants. To prevent the exploitation of Modbus TCP vulnerabilities in industrial systems, [11] proposed Deep Packet Inspection (DPI) enabled IDS. Though the effective use of DPI helps in enhancing security and data flow monitoring, it has failed to eliminate the vulnerabilities in the entire network. To detect DDoS attacks, a Machine learning-based detection framework is designed [12]. The features that represent the network traffic have been extracted to reduce the computational overhead of the detection model. However, the effectiveness and robustness of the feature selection algorithm is unexplained. [13] is focused on identifying the phishing attacks that target IoT devices. This work employed the benefits of random forest and support vector machines. The data is pre-processed using Principal Component Analysis (PCA) and the non-numerical data is eliminated for further processing which leads to data loss. The intrusions in IIoT traffic are identified by [14]. Deep-IFS, a forensic model that manages heavy IIoT-traffic issues based on local representations of the data. However, the model failed to detect unknown attacks. This approach lacks the use of unsupervised learning algorithms. [15] proposed a hypergraph and Rough Set Theory-based feature selection model to reduce the complexity of the attack detection technique. The proposed technique is limited to identifying the known attacks with reduced features. Moreover, the effectiveness of the algorithm is considered only in the benchmark datasets and it is not used in real-time implementation. The hypergraph-based deep learning approach is developed by [16] to identify both known and unknown attacks. Though the model has achieved the highest detection rate, faulty components are also involved in the attack vectors that compromise the quality of the detection algorithm. [17], to detect cyber-attacks against SCADA networks, an ensemble-based attack detection model is developed that is inefficient against unlabelled data. [18] has developed a countermeasure technique to identify network intrusions and application intrusions. The proposed technique is based on LSTM and autoencoders. From the extensive study, it is clear that the novel threats pose challenges that emphasize the improved security mechanisms to ensure data integrity in IoT sectors and in need for a robust model to defend against cyber-attacks and vulnerabilities.

2.2 MERITS

- No additional machine learning methods are used in the feature selection process; deep learning methods performed this task alone
- The experimental results obtained by the proposed deep learning models are superior to the previous models developed in the same dataset
- The results demonstrate the superiority of the deep learning models in detecting abnormal events within large and complex datasets

- The proposed CNN + LSTM model attained high accuracy in all types of in the UNSW-NB15 dataset generated specifically for IIoT networks
- The Hybrid CNN+LSTM model captures contextual information and learning from historical network traffic patterns which reduces the False Positive Rate.
- The proposed model uses CNN to extract spatial features and LSTM to capture temporal patterns, thereby enhancing the detection rate

2.3 DEMERITS

- The proposed models require more training durations due to the complexity of the dataset
- Dataset imbalance presents a significant challenge, potentially biasing model performance
- Supervised learning increases vulnerability to overfitting when encountering new attack patterns
- Identifying the optimal Hyperparameters for the model requires an extensive trial-and-error process which is time-consuming
- UNSW-NB15 is a sensor-generated dataset which adds complexity and increases the difficulty of training models

CHAPTER 3
SOURCE CODE

3.1 PREPROCESSING

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from sklearn import preprocessing
train = pd.read_csv("UNSW_NB15_training-set.csv")
test = pd.read_csv("UNSW_NB15_testing-set.csv")
data = pd.concat([train, test], axis=0)
data[data['service']=='-']
data['service'].replace('-',np.nan,inplace=True)
data.dropna(inplace=True)
features = pd.read_csv("NUSW-NB15_features.csv",encoding='cp1252')
features['Type '] = features['Type '].str.lower()
nominal_names = features['Name'][features['Type ']=='nominal']
integer_names = features['Name'][features['Type ']=='integer']
binary_names = features['Name'][features['Type ']=='binary']
float_names = features['Name'][features['Type ']=='float']

cols = data.columns
nominal_names = cols.intersection(nominal_names)
integer_names = cols.intersection(integer_names)
binary_names = cols.intersection(binary_names)
float_names = cols.intersection(float_names)

for c in integer_names:
  pd.to_numeric(data[c])
for c in binary_names:
```

```
  pd.to_numeric(data[c])
for c in float_names:
  pd.to_numeric(data[c])

num_col = data.select_dtypes(include='number').columns
cat_col = data.columns.difference(num_col)
cat_col = cat_col[1:]
data_cat = data[cat_col].copy()
data_cat = pd.get_dummies(data_cat,columns=cat_col)
data = pd.concat([data, data_cat],axis=1)
data.drop(columns=cat_col,inplace=True)
num_col = list(data.select_dtypes(include='number').columns)
num_col.remove('id')
num_col.remove('label')
print(num_col)

minmax_scale = MinMaxScaler(feature_range=(0, 1))
def normalization(df,col):
        for i in col:
                arr = df[i]
                arr = np.array(arr)
                df[i] = minmax_scale.fit_transform(arr.reshape(len(arr),1))
        return df
data = normalization(data.copy(),num_col)

bin_label = pd.DataFrame(data.label.map(
        lambda x:'normal' if x==0 else 'abnormal')
)
bin_data = data.copy()
bin_data['label'] = bin_label
```

```
le1 = preprocessing.LabelEncoder()
enc_label = bin_label.apply(le1.fit_transform)
bin_data['label'] = enc_label
multi_data = data.copy()
multi_label = pd.DataFrame(multi_data.attack_cat)
multi_data = pd.get_dummies(multi_data,columns=['attack_cat'])

le2 = preprocessing.LabelEncoder()
enc_label = multi_label.apply(le2.fit_transform)
multi_data['label'] = enc_label
num_col.append('label')

corr_bin = bin_data[num_col].corr()
num_col = list(multi_data.select_dtypes(include='number').columns)
corr_multi = multi_data[num_col].corr()
corr_ybin = abs(corr_bin['label'])
highest_corr_bin = corr_ybin[corr_ybin >0.3]
highest_corr_bin.sort_values(ascending=True)
bin_cols = highest_corr_bin.index
bin_data = bin_data[bin_cols].copy()
bin_data.to_csv('bin_data.csv')

corr_ymulti = abs(corr_multi['label'])
highest_corr_multi = corr_ymulti[corr_ymulti >0.3]
highest_corr_multi.sort_values(ascending=True)
print(corr_ymulti[corr_ymulti >0.0])
multi_cols = highest_corr_multi.index
multi_data = multi_data[multi_cols].copy()
multi_data.to_csv('multi_data.csv')
```

3.2 CONVOLUTIONAL NEURAL NETWORK

3.2.1 IMPORTING LIBRARIES

```
import numpy as np
import pandas as pd
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D, Dropout, Dense, LSTM,  Flatten
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping, ModelCheckpoint
from plot import plot_training_vs_validation
from sklearn.metrics import confusion_matrix, classification_report
```

3.2.2 IMPORTING DATASETS

```
bin_data = pd.read_csv("bin_data.csv")
multi_data = pd.read_csv("multi_data.csv")
```

3.2.3 BINARY CLASS CLASSIFICATION

```
X = bin_data.drop(columns=['label'],axis=1)
Y = bin_data['label']
X_train, X_test, y_train, y_test = train_test_split(
        X, Y, test_size=0.20, random_state=50
)
X_train_reshaped = X_train.values.reshape(
        (X_train.shape[0], X_train.shape[1], 1)
)
X_test_reshaped = X_test.values.reshape(
        (X_test.shape[0], X_test.shape[1], 1)
)
```

```
unique_labels = np.unique(y_train)
num_classes = len(unique_labels)

print("Unique Labels:", unique_labels)
print("Number of Classes:", num_classes)

cnn_model = Sequential()
cnn_model.add(Conv1D(
        filters=64,
        kernel_size=3,
        activation='relu',
        input_shape=(X_train.shape[1], 1)
        )
)
cnn_model.add(MaxPooling1D(pool_size=2))
cnn_model.add(Dropout(0.35))
cnn_model.add(MaxPooling1D(pool_size=2))
cnn_model.add(Dropout(0.15))
cnn_model.add(Flatten())
cnn_model.add(Dense(64, activation='relu'))
cnn_model.add(Dense(num_classes, activation='softmax'))

adam_optimizer = Adam(learning_rate=1e-4)
cnn_model.compile(
        loss='sparse_categorical_crossentropy',
        optimizer=adam_optimizer,
        metrics=['accuracy']
)
callbacks = [
        EarlyStopping(monitor='val_loss', patience=10),
```

```
        ModelCheckpoint(
        filepath='best_model_cnn.h5',
        monitor='val_loss',
        save_best_only=True)
]
cnn_history = cnn_model.fit(
   X_train_reshaped, y_train,
   epochs=50,
   batch_size=256,
   validation_data=(X_test_reshaped, y_test),
   callbacks=callbacks
)
plot_training_vs_validation(cnn_history, "CNN")
y_test_probabilities = cnn_model.predict(X_test_reshaped)
y_test_pred = np.argmax(y_test_probabilities, axis=1)
conf_matrix = confusion_matrix(y_test, y_test_pred)
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_test_pred))
```

3.2.4 MULTI CLASS CLASSIFICATION

```
X = multi_data.drop(columns=['label'], axis=1)
Y = multi_data['label']
X_train, X_test, y_train, y_test = train_test_split(
        X, Y, test_size=0.20, random_state=50
)
X_train_reshaped = np.expand_dims(X_train.values, axis=2)
X_test_reshaped = np.expand_dims(X_test.values, axis=2)
```

```
cnn_model = Sequential()
cnn_model.add(Conv1D(
        filter=64,
        kernel_size=3,
        activation='relu',
        input_shape=(X_train_reshaped.shape[1], 1)
        )
)
cnn_model.add(MaxPooling1D(pool_size=2))
cnn_model.add(Conv1D(32, kernel_size=3, activation='relu'))
cnn_model.add(MaxPooling1D(pool_size=2))
cnn_model.add(Flatten())
cnn_model.add(Dense(64, activation='relu'))
cnn_model.add(Dropout(0.2))
cnn_model.add(Dense(10, activation='softmax'))
plot_training_vs_validation(cnn_history)
test_loss, test_accuracy = cnn_model.evaluate(X_test_reshaped, y_test)
print(f"Test Loss: {test_loss}")
print(f"Test Accuracy: {test_accuracy}")
```

3.3 LONG SHORT-TERM MEMORY

3.3.1 IMPORTING LIBRARIES

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense, LSTM
from plot import plot_training_vs_validation
from sklearn.metrics import confusion_matrix, classification_report
```

```
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping, ModelCheckpoint
```

3.3.2 IMPORTING DATASETS

```
bin_data = pd.read_csv("bin_data.csv")
multi_data = pd.read_csv("multi_data.csv")
```

3.3.3 BINARY CLASS CLASSIFICATION

```
X = bin_data.drop(columns=['label'],axis=1)
Y = bin_data['label']
X_train, X_test, y_train, y_test = train_test_split(
        X, Y, test_size=0.20, random_state=50
)
X_train_reshaped = X_train.values.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test_reshaped = X_test.values.reshape((X_test.shape[0], X_test.shape[1], 1))

unique_labels = np.unique(y_train)
num_classes = len(unique_labels)
print("Unique Labels:", unique_labels)
print("Number of Classes:", num_classes)

lstm_model = Sequential()
lstm_model.add(LSTM(64, activation='relu', input_shape=(14, 1)))
lstm_model.add(Dense(64, activation='relu'))
lstm_model.add(Dense(num_classes, activation='softmax'))

adam_optimizer = Adam(learning_rate=1e-4)
lstm_model.compile(
        loss='sparse_categorical_crossentropy',
        optimizer=adam_optimizer,
```

```
        metrics=['accuracy']
)

lstm_history = lstm_model.fit(
    X_train_reshaped, y_train,
   epochs=50,
   batch_size=256,
   validation_data=(X_test_reshaped, y_test),
   callbacks=callbacks
)

plot_training_vs_validation(lstm_history, "LSTM")
y_test_probabilities = lstm_model.predict(X_test_reshaped)
y_test_pred = np.argmax(y_test_probabilities, axis=1)
conf_matrix = confusion_matrix(y_test, y_test_pred)
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_test_pred))
```

3.3.4 MULTI CLASS CLASSIFICATION

```
X = multi_data.drop(columns=['label'],axis=1)
Y = multi_data['label']
X_train, X_test, y_train, y_test = train_test_split(
        X, Y, test_size=0.20, random_state=50
)
X_train_reshaped = X_train.values.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test_reshaped = X_test.values.reshape((X_test.shape[0], X_test.shape[1], 1))
lstm_model_multi = Sequential()
lstm_model_multi.add(LSTM(
```

```
        64, activation='relu', input_shape=(69, 1))
)
lstm_model_multi.add(Dense(64, activation='relu'))
lstm_model_multi.add(Dense(num_classes, activation='softmax'))
adam_optimizer = Adam(learning_rate=1e-4)
lstm_model_multi.compile(
        loss='sparse_categorical_crossentropy',
         optimizer=adam_optimizer,
        metrics=['accuracy']
)

callbacks = [
        EarlyStopping(monitor='val_loss', patience=10),
        ModelCheckpoint(
                filepath='best_model_cnn.h5',
                monitor='val_loss', save_best_only=True
        )
]

lstm_history_multi = lstm_model_multi.fit(
    X_train_reshaped, y_train,
   epochs=50,
   batch_size=256,
   validation_data=(X_test_reshaped, y_test),
   callbacks=callbacks
)
plot_training_vs_validation(lstm_history, "LSTM")
y_test_probabilities = lstm_model.predict(X_test_reshaped)
y_test_pred = np.argmax(y_test_probabilities, axis=1)
conf_matrix = confusion_matrix(y_test, y_test_pred)
```

```
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_test_pred))
```

3.4 HYBRID CNN + LSTM

3.4.1 IMPORTING LIBRARIES

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D, Dropout, Dense, LSTM
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping, ModelCheckpoint
from plot import plot_training_vs_validation
from sklearn.metrics import confusion_matrix, classification_report
```

3.4.2 IMPORTING DATASETS

```
bin_data = pd.read_csv("bin_data.csv")
multi_data = pd.read_csv("multi_data.csv")
```

3.4.3 BINARY CLASS CLASSIFICATION

```
X = bin_data.drop(columns=['label'],axis=1)
Y = bin_data['label']
X_train, X_test, y_train, y_test = train_test_split(
        X, Y, test_size=0.20, random_state=50
)
X_train_reshaped = X_train.values.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test_reshaped = X_test.values.reshape((X_test.shape[0], X_test.shape[1], 1))
hybrid_model_bin = Sequential()
```

```
hybrid_model_bin.add(Conv1D(
        filters=64,
        kernel_size=3,
        activation='relu',
        input_shape=(X_train_reshaped.shape[1], 1)
        )
)
hybrid_model_bin.add(MaxPooling1D(pool_size=2))
hybrid_model_bin.add(LSTM(units=100, activation='relu', return_sequences=True))
hybrid_model_bin.add(Dropout(0.35))
hybrid_model_bin.add(LSTM(units=60, activation='relu'))
hybrid_model_bin.add(Dropout(0.15))
hybrid_model_bin.add(Dense(units=10, activation='softmax'))
adam_optimizer = Adam(learning_rate=1e-4)
hybrid_model_bin.compile(
        loss='sparse_categorical_crossentropy',
        optimizer=adam_optimizer,
        metrics=['accuracy']
)
callbacks = [
        EarlyStopping(monitor='val_loss', patience=10),
        ModelCheckpoint(
                filepath='best_model_cnn.h5',
                monitor='val_loss',
                save_best_only=True
        )
]
hybrid_model_bin_history = hybrid_model_bin.fit(
        X_train_reshaped, y_train,
        epochs=50,
```

```
        batch_size=256,
        validation_data=(X_test_reshaped, y_test),
        callbacks=callbacks
)
plot_training_vs_validation(hybrid_model_bin_history, "CNN-LSTM | Binary")
y_test_probabilities = hybrid_model_bin_history.predict(X_test_reshaped)
y_test_pred = np.argmax(y_test_probabilities, axis=1)
conf_matrix = confusion_matrix(y_test, y_test_pred)
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_test_pred))
```

3.4.4 MULTI CLASS CLASSIFICATION

```
X = multi_data.drop(columns=['label'],axis=1)
Y = multi_data['label']
X_train, X_test, y_train, y_test = train_test_split(
        X, Y, test_size=0.20, random_state=50
)
X_train_reshaped = X_train.values.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test_reshaped = X_test.values.reshape((X_test.shape[0], X_test.shape[1], 1))
print(X_train_reshaped.shape)
print(X_test_reshaped.shape)
X_train_padded = np.pad(X_train_reshaped, ((0, 0), (0, 1), (0, 0)), mode='constant')
X_test_padded = np.pad(X_test_reshaped, ((0, 0), (0, 1), (0, 0)), mode='constant')
print(X_train_padded.shape)
print(X_test_padded.shape)
adam_optimizer = Adam(learning_rate=1e-4)
cnn_model.compile(
                loss='sparse_categorical_crossentropy',
```

```
                optimizer=adam_optimizer,
                metrics=['accuracy']
        )
callbacks = [
        EarlyStopping(monitor='val_loss', patience=10),
        ModelCheckpoint(
                filepath='best_model_cnn.h5',
                monitor='val_loss',
                save_best_only=True
        )
]
hybrid_multi_history = cnn_model.fit(
        X_train_padded, y_train,
        epochs=50,
        batch_size=256,
        validation_data=(X_test_padded, y_test),
        callbacks=callbacks
)
plot_training_vs_validation(cnn_history, "CNN-LSTM")
y_test_probabilities = cnn_model.predict(X_test_padded)
y_test_pred = np.argmax(y_test_probabilities, axis=1)

conf_matrix = confusion_matrix(y_test, y_test_pred)
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_test_pred))
```

CHAPTER 4

SNAPSHOTS

4.1 BINARY CLASS

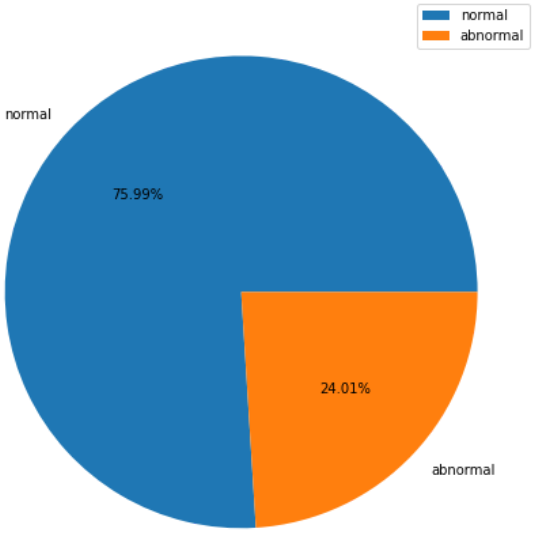


Fig 4.1 Pie Chart Distribution of Normal and Abnormal Labels for Binary class Classification

4.2 MULTI CLASS

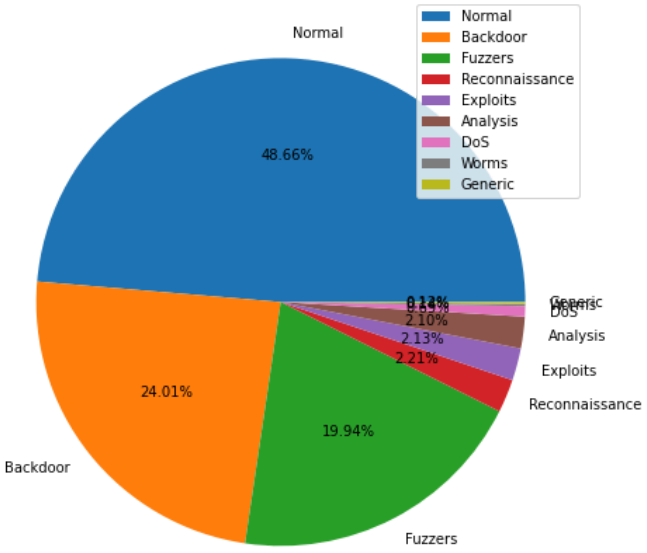


Fig 4.2 Pie Chart Distribution of Normal and Abnormal Labels for Multi class Classification

CHAPTER 5

RESULT AND DISCUSSIONS

5.1 RESULT ANALYSIS

5.1.1 CONVOLUTIONAL NEURAL NETWORK

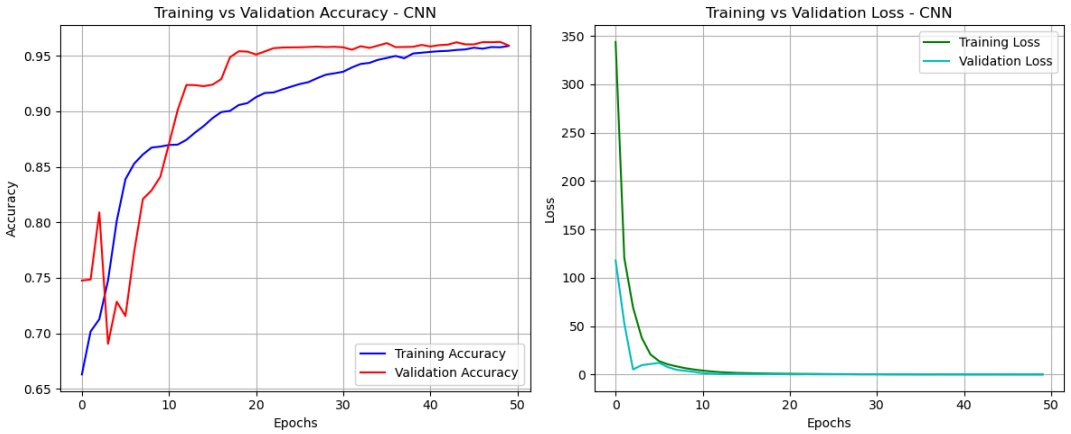


Fig 5.1 Training vs Validation Accuracy and Loss for Binary Class CNN

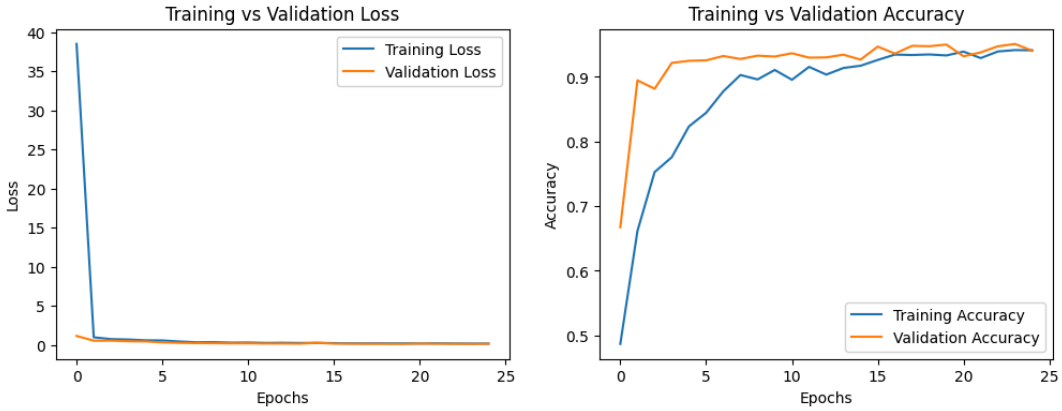


Fig 5.2 Training vs Validation Accuracy and Loss for Multi Class CNN

5.1.2 LONG SHORT-TERM MEMORY

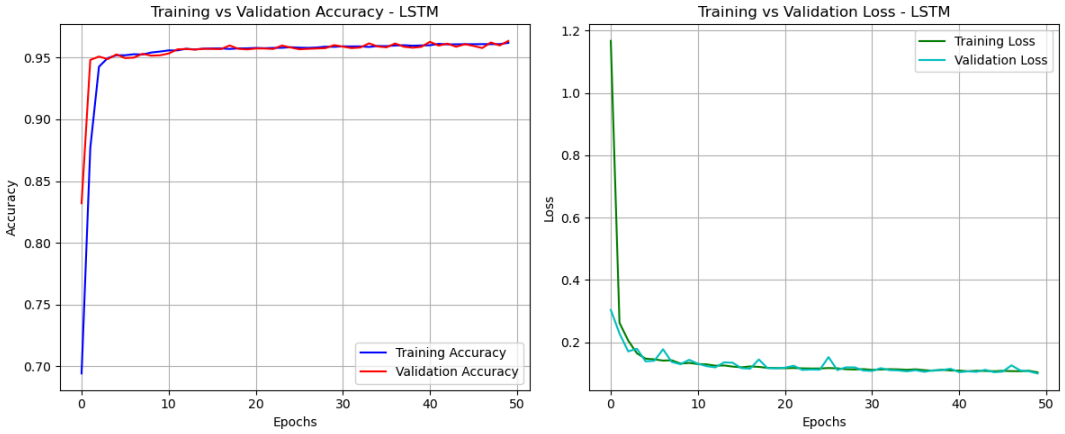


Fig 5.3 Training vs Validation Accuracy and Loss for Binary Class LSTM

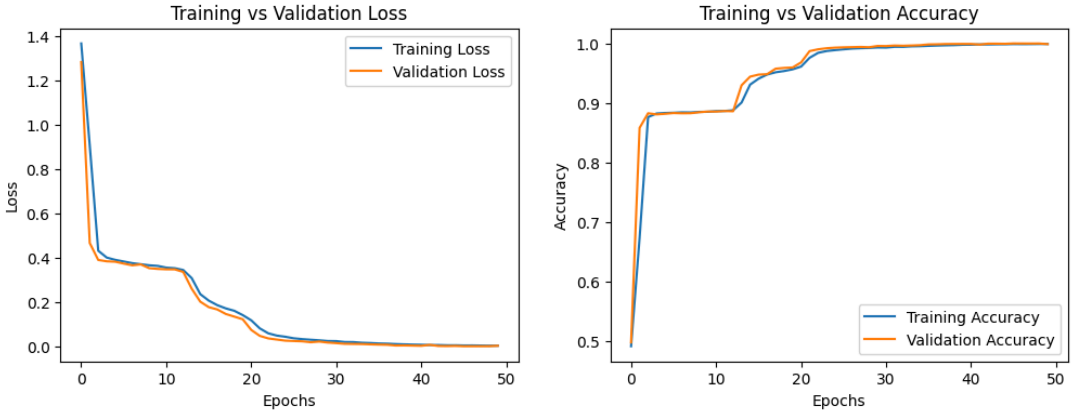


Fig 5.4 Training vs Validation Accuracy and Loss for Multi Class LSTM

5.1.3 HYBRID CNN + LSTM

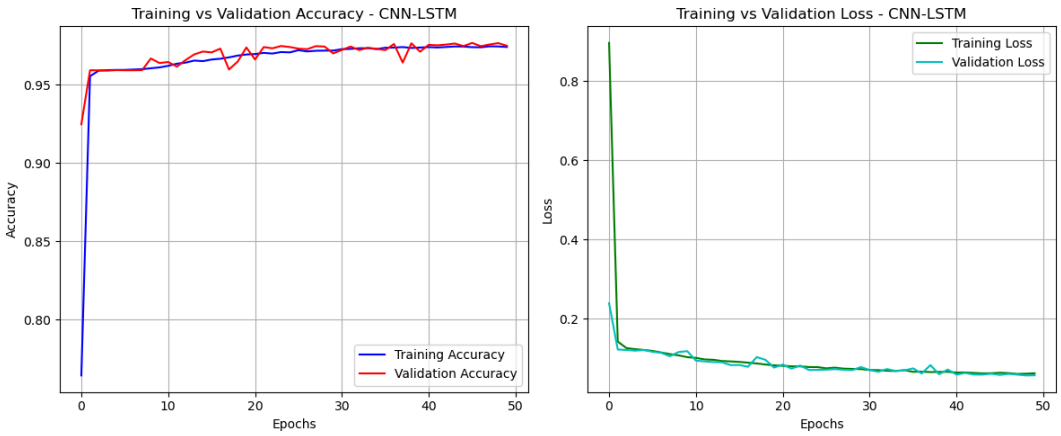


Fig 5.5 Training vs Validation Accuracy and Loss for Binary Class Hybrid CNN+LSTM

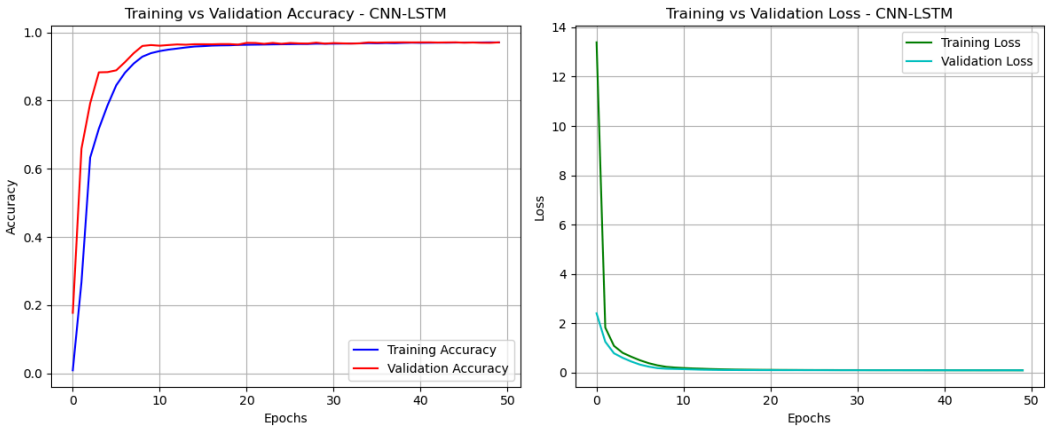


Fig 5.6 Training vs Validation Accuracy and Loss for Multi Class Hybrid CNN+LSTM

5.2 RESULT COMPARISON

Table 5.7 Performance Comparison

MODEL	BASE PAPER		PROPOSED METHODOLOGY	
	Binary Classification	Multi-Class Classification	Binary Classification	Multi-Class Classification
CNN	99.15%	99.26%	96%	95%
LSTM	99.05%	98.91%	96%	99%
CNN-LSTM	99.84%	99.80%	97%	97%

CHAPTER 6

CONCLUSIONS AND FUTURE PLANS

6.1 CONCLUSION

The emerging growth of IoT technologies has led to an increased usage of smart devices in the industrial sector. The increased utilisation of technology has made the networks more sophisticated and are highly susceptible to cyberattacks and intrusions. Therefore, it is essential to develop an effective attack detection approach to ensure the security of IIoT networks. Hence, this work has presented the “Hybrid CNN+LSTM”, a deep learning-based attack detection model which integrates the benefits of feature selection and deep learning architecture that are capable of detecting attacks originating from both IICS and external networks. The model is trained on labelled attacks to recognize the accurate identification of malicious and legitimate events. One of the models utilized in the study, “Hybrid CNN+LSTM” have high accuracy and less time complexity over conventional CNN-based and LSTM-based approaches.

As a future work, the proposed algorithm can be deployed in various Cyber Physical System testbeds to analyse the adaptiveness of the model in learning different CPS traffic and improving the detection accuracy.

CHAPTER 7

REFERENCES

[1] Mekala, Sri Harsha, Zubair Baig, Adnan Anwar, and Sherali Zeadally. "Cybersecurity for industrial IoT (IIoT): Threats, countermeasures, challenges and future directions." Computer Communications, 2023.

[2] Adepu, Sridhar, and Aditya Mathur. "An investigation into the response of a water treatment system to cyber attacks." In 2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE), pp. 141-148. IEEE, 2016.

[3] B. Bencsáth, G. Pék, L. Buttyán, M. Félegyházi, Duqu: A Stuxnet-like malware found in the wild, CrySyS Lab Tech. Rep. 14 (2011) 1–60.

[4] D. Lee, Flame: Massive cyber-attack discovered, researchers say, BBC News 5 (28), 2012

[5] Wu, Di, Zhongkai Jiang, Xiaofeng Xie, Xuetao Wei, Weiren Yu, and Renfa Li. "LSTM learning with Bayesian and Gaussian processing for anomaly detection in industrial IoT." IEEE Transactions on Industrial Informatics 16, no. 8 pp. 5244-5253, 2019

[6] Bencsáth, Boldizsár, Gábor Pék, Levente Buttyán, and Márk Félegyházi. "Duqu: A Stuxnetlike malware found in the wild." CrySyS Lab Technical Report 14 pp. 1-60,2011

[7] Ahmad, U., A node pairing approach to secure the Internet of Things using machine learning. Journal of Computational Science, 62, p.101718, 2022

[8] Lin, Chih-Yuan, and Simin Nadjm-Tehrani. "Protocol study and anomaly detection for serverdriven traffic in SCADA networks." International Journal of Critical Infrastructure Protection, 100612, 2023

[9] Thakkar, Ankit, and Ritika Lohiya. "Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System." Information Fusion Vol. 90, pp. 353-363, 2023

[10] Kirupakar, J., and S. Mercy Shalinie. "Situation aware intrusion detection system design for industrial IoT gateways." In 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), pp. 1-6. IEEE, 2019.

[11] Nyasore, Osborn N., Pavol Zavarsky, Bobby Swar, Raphael Naiyeju, and Shubham Dabra. "Deep packet inspection in industrial automation control systems to mitigate attacks exploiting modbus/TCP vulnerabilities." In 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), pp. 241- 245. IEEE, 2020.

[12] Kachavimath, Amit V., Shubhangeni Vijay Nazare, and Sheetal S. Akki. "Distributed denial of service attack detection using naïve bayes and k-nearest neighbor for network forensics." In 2020 2nd International conference on innovative mechanisms for industry applications (ICIMIA), pp. 711-717. IEEE, 2020.

[13] Naaz, Sameena. "Detection of phishing in the internet of things using a machine learning approach." International Journal of Digital Crime and Forensics (IJDCF) 13, no. 2 pp. 1-15, 2021

[14] Wiyono, Rizky Tri, and Niken Dwi Wahyu Cahyani. "Performance analysis of decision tree c4. 5 as a classification technique to conduct network forensics for botnet activities in the internet of things." In 2020 International Conference on Data Science and Its Applications (ICoDSA), pp. 1-5. IEEE, 2020

[15] Priyanga, S., M. R. Gauthama Raman, Sujeet S. Jagtap, N. Aswin, Kannan Kirthivasan, and V. S. Shankar Sriram. "An improved rough set theory based feature selection approach for intrusion detection in SCADA systems." Journal of Intelligent & Fuzzy Systems 36, no. 5 pp. 3993-4003, 2019

[16] Krithivasan, Kannan, S. Pravinraj, and Shankar Sriram VS. "Detection of cyberattacks in industrial control systems using enhanced principal component analysis and hypergraph-based convolution neural networks (EPCA-HG-CNN)." IEEE Transactions on Industry Applications 56, no. 4 pp.4394-4404, 2020

[17] Abdel-Basset, Mohamed, Victor Chang, Hossam Hawash, Ripon K. Chakrabortty, and Michael Ryan. "Deep-IFS: Intrusion detection approach for industrial internet of things traffic in fog environments." IEEE Transactions on Industrial Informatics 17, no. 11 pp. 7704-7715, 2020

[18] Koroniotis, Nickolaos, Nour Moustafa, and Elena Sitnikova. "A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework." Future Generation Computer Systems 110 pp. 91-106, 2020

[19] Zolanvari, Maede, Marcio A. Teixeira, Lav Gupta, Khaled M. Khan, and Raj Jain. "Machine learning-based network vulnerability analysis of industrial Internet of Things." IEEE Internet of Things Journal 6, no. 4 pp. 6822-6834, 2019

[20] S. Tamy, H. Belhadaoui, M.A. Rabbah, N. Rabbah, M. Rifi, An evaluation of machine learning algorithms to detect attacks in SCADA network, in: 2019 7th Mediterranean Congress of Telecommunications, CMT, IEEE, pp. 1–5, 2019

CHAPTER 8

APPENDIX - BASE PAPER

Engineering Science and Technology, an International Journal 38 (2023) 101322

HOSTED BY

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

A hybrid CNN + LSTMbased intrusion detection system for industrial IoT networks

Hakan Can Altunay [a], Zafer Albayrak [b,*]

[a] Department of Computer Technologies, Carsamba Chamber of Commerce Vocational School, Ondokuz Mayis University, Samsun 55200, Turkey
[b] Department of Computer Engineering, Faculty of Technology, Sakarya University of Applied Sciences, Sakarya 54100, Turkey

ARTICLE INFO

Article history:
Received 30 June 2022
Revised 29 October 2022
Accepted 25 December 2022

Keywords:
Intrusion detection system
Convolutional neural network
Internet of Things
IIoT
Long short term memory

ABSTRACT

The Internet of Things (IoT) ecosystem has proliferated based on the use of the internet and cloud-based technologies in the industrial area. IoT technology used in the industry has become a large-scale network based on the increasing amount of data and number of devices. Industrial IoT (IIoT) networks are intrinsically unprotected against cyber threats and intrusions. It is, therefore, significant to develop Intrusion Detection Systems (IDS) in order to ensure the security of the IIoT networks. Three different models were proposed to detect intrusions in the IIoT network by using deep learning architectures of Convolutional Neural Network (CNN), Long Short Term Memory (LSTM), and CNN + LSTM generated from a hybrid combination of these. In the study conducted by using the UNSW-NB15 and X-IIoTID datasets, normal and abnormal data were determined and compared with other studies in the literature following a binary and multi-class classification. The hybrid CNN + LSTM model attained the highest accuracy value for intrusion detection in both datasets among the proposed models. The proposed CNN + LSTM architecture attained an accuracy of 93.21% for binary classification and 92.9% for multi-class classification in the UNSW-NB15 dataset while the same model attained a detection accuracy of 99.84% for binary classification and 99.80% for multi-class classification in the X-IIoTID dataset. In addition, the accurate detection success of the implemented models regarding the types of attacks within the datasets was evaluated.

1. Introduction

The IoT has been widely adopted by several industries such as health care, energy, and household appliance manufacturers lately. The increasing use of IoT technology in the industry has significantly changed our lifestyles [1]. The concept of IIoT has emerged with the use of IoT in the industry for different and special purposes based on the advancement of technology. IIoT contains several things including an actuator, sensor, control system, communication channels, integration interface, advanced security systems, vehicle network systems, and smart home products. All things within the IIoT can be controlled through the internet. The use of the industrial internet of things in various industries has increased the capacities of various sectors such as device quality systems, product safety and management systems, and logistics management systems as well as substantially enhancing labor productivity. Moreover, IIoT technology connects a physical area to a virtual one by authorizing various applications, interconnect components, and product support [2].

The Internet Protocol using the MQTT as the messaging protocol to send messages to the target device, Modbus TCP protocol enabling the communication between PLCs, and communication protocols like LoRaWAN enabling the transmission of data over long distances with low power consumption are frequently used in the IIoT networks [3,4]. Moreover, the majority of IIoT devices can receive the incoming data and transmit these to other devices after processing them. These features sensitize the devices against some privacy and security threats bearing the potential of risking their industrial internet of things systems and the applications they belong to [5]. The most essential feature of the IIoT nodes is being active the whole time during the data collection, processing, and transmission processes. All layers in the IIoT are listed as the application, data processing, network, and sensing layers. Control and optimization and data flow are conducted between these layers [6]. In addition, each layer may put the systems within the IIoT at risk since they have a tendency for various attacks and intrusions. The most common cyber threats and unauthorized access violations in the IIoT ecosystem are access control attacks, partial or full contamination of data, jamming attacks, DoS attacks, DDoS

* Corresponding author.
E-mail addresses: hakancan.altunay@omu.edu.tr (H.C. Altunay), zaferalbayrak@subu.edu.tr (Z. Albayrak).

https://doi.org/10.1016/j.jestch.2022.101322
2215-0986/

attacks, exploit attacks, and backdoor attacks [7]. Several organizations use Intrusion Detection Systems (IDS) in order to resist malicious attacks and protect the security of IIoT nodes and networks. These IDSs can be configurated in either of the layers presented in 1 [8].

IDS enables the protection of the integrity, privacy, and security of data transmitted over the network and, therefore, it plays a crucial role in the IIoT. Tasks of preventing an IIoT network from attacks partially or wholly or preventing, detecting, reacting, and reporting malicious activity are performed by the IDS [9]. It is challenging to run intrusion detection systems that do not use machine learning algorithms and are based on statistical methods on big data. IDSs generated through statistical methods are behavior-based. Summaries based on previous rules are used with the aim of modeling the behaviors on the network. However, this method is not preferred today due to the fact that it is challenging to measure these summaries [10]. The classification of the traditional IDSs is usually presented as signature-based, anomaly-based, and hybrid IDSs. Signature-based IDSs basically extract the behavior models of intruders. The characteristic features of the intrusions that have been encountered before are analyzed in these models named intrusion signatures. As the intrusion signatures in the generated database and malware behavior match, these behaviors are detected as intrusions. The signature-based IDSs do not generate false positives; they also detect each intrusion whose signature is within the database. The problem with signature-based intrusion detection systems is the inability to detect intrusions whose signatures are unknown and, accordingly, the generation of high rates of false negatives. In order to lower the rate of false negatives, the database that is generated with intrusion signatures must be updatable. Anomaly-based IDSs, on the other hand, are based on the principle of distinguishing abnormal events and normal events on the network. In anomaly-based IDSs, the behavior profiles of the users in the system are identified first. The behaviors differing from the normal behaviors are identified as abnormal behaviors. The higher the rate of accurate identification of the normal behavior profiles, the higher the rate of accurate identification of the abnormal behaviors. Normal behaviors are constantly updated in anomaly-based IDSs. Therefore, intrusions that have not been encountered before can be detected through this method. The disadvantage of anomaly-based IDSs is the possibility of generating high rates of false alarms in various intrusion types [11]. Hybrid IDSs were designed to reduce the negative aspects of the signature- and anomaly-based IDSs. High false-positive rates and low detection accuracy reduce the accuracy of the results of intrusion detection in traditional IDSs. They also cannot prevent events

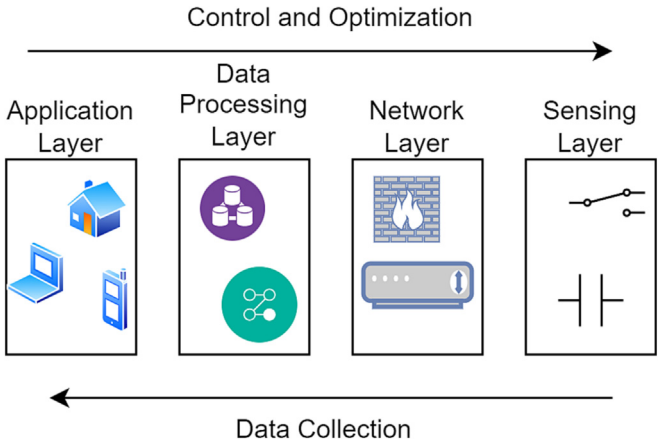


Fig. 1. General IIoT architecture.

such as Slowloris DoS attacks [12]. Researchers investigated the implementation of Deep Learning-based techniques, which is a specialized subset of Artificial Intelligence, in order to increase the performance of IDSs [13].

Machine Learning (ML) methods are the most frequently preferred methods to perform intrusion detection. ML can empower the experiential learning and decision-making processes of the various systems by improving skills and capacity without any open programming [14]. The five types of ML algorithms are listed as supervised, unsupervised, semi-supervised, reinforcement, and federated. Supervised ML techniques perform the learning operation from a predefined dataset in order to determine future predictions as well as improving the decision-making processes. Moreover, unsupervised machine learning approaches are used for undefined data. All data are fit to be used in supervised learning machines. In addition, the conditions may not have been fulfilled at the time when the algorithm starts. The reinforcement machine learning approach is the method calculating the rewards or errors based on the interactions in a specific medium [15]. Federated learning is a technique preserving the privacy area where other learning types fall short. Data transmitted from multiple independent devices are kept upon being anonymized. In this method, the algorithms are trained over the servers [16].

In the IDSs generated by using deep learning methods, advanced and updated datasets are preferred to perform a higher number of intrusion detection. Deep learning-based intrusion detection systems are usually trained by using the latest datasets generated for intrusion detection. In addition, the types of attacks and attributions are high in number in the up-to-date datasets. The accurate intrusion detection rate of deep learning architectures usually increases as the number of attributes in a dataset increases [17]. Deep learning methods reduce the size of the attribute vector into the optimal number of the required attributes. There is no need for an attribute selection or extraction process to guarantee this in the deep learning methods [18]. In some studies, nevertheless, attributes were determined through any of the machine learning methods prior to the deep learning method in order to accomplish better results [19]. In addition, the results obtained by the model proposed by using the UNSW-NB15 and X-IIoTID datasets were analyzed. UNSW-NB15 is commonly used by researchers [20]. In the UNSW-NB15 dataset, the network traces imitating the network traffic models in the real world were generated in the laboratory environment just like the datasets generated by the IIoT network system [21]. The UNSW-NB15 is a larger and more imbalanced dataset in terms of the number of types of attacks it involves compared to the other datasets used in intrusion detection [22]. It also involves higher variations of network traffic designs compared to the other datasets. Moreover, various datasets used in intrusion detection were generated based on the UNSW-NB15 dataset [23]. X-IIoTID is an up-to-date dataset created with the aim of detecting intrusions in complex IIoT networks. There are 9 types of intrusions in the X-IIoTID dataset namely Reconnaissance, Command and Control, Weaponization, Exploitation, Tampering, Lateral Movement, Exfiltration, Crypto Ransomware, RDoS, and Normal. These 9 types of intrusions also contain various types of intrusions within themselves, thus leading to a total of 18 types of intrusions within the dataset [24].

The main motivation and contribution of this study are summarized below.

The number of cyber-attacks on industrial IoT networks increases day by day, causing pecuniary and non-pecuniary losses. Therefore, it is essential to detect these attacks and take the system under protection. In this study, an intrusion detection system for industrial IoT networks is initially by using deep learning models.

Secondly, the proposed models for intrusion detection systems were tested on complex datasets where there are large amounts of

data and high numbers of attributes considering the gradual increase in the amount of data in industrial IoT networks. The performances of our proposed models were evaluated through binary and multi-class classification procedures conducted on up-to-date datasets, allowing a more reliable and observable process.

Finally, a comparison was performed between the CNN-based, LSTM-based, and hybrid CNN + LSTM methods proposed by ourselves and the studies conducted in the literature. The accuracy rates for the intrusion detection attained by the proposed models were higher compared to the results obtained in the studies conducted in the literature. These results prove that the hybrid CNN + LSTM model leads to an improvement in the performance of IDSs.

In Section II of the article, the related studies were summarized. In Section III, the UNSW-NB15 and X-IIoTID datasets were introduced and their features were explained. In Section IV, the proposed deep learning-based IDS model was disclosed. In Section V, the study was summarized and the obtained results were explained in detail. In the last section, the results were discussed and the article concluded by addressing the future studies planned.

2. Related works

In this section, IDSs that were generated by using machine and deep learning methods and the results obtained from these studies were summarized. In addition, an evaluation was performed on the various models and architectures that were previously implemented for the detection of intrusions on IoT networks.

The authors created a dataset named X-IIoTID for the complex IIoT networks. This dataset was created by collecting data coming from devices with different brands and other devices to which they are connected through various platforms. The detection rate of machine and deep learning algorithms that were frequently used in the study regarding the types of attacks in the X-IIoTID dataset was determined. In addition, the results received by the algorithms used in the study were compared to 18 different datasets. It was stated by the authors that the X-IIoTID dataset attained 20 features required for the IIoT intrusion dataset and is superior to the other 18 datasets [24].

In another study conducted using the X-IIoTID dataset, a reliable model was proposed for data sharing. The proposed model was based on federated learning. Edge devices were included in the consensus computing. The proposed model was tested through deep learning algorithms. According to the results revealed, high efficiency was achieved. It was expressed that the model named SecureIIoT attained an accuracy of 99.79% in binary classification [25].

In another study, a model was proposed for the prevention of ransomware attacks on the devices within the IIoT systems. In the model composed of two sections, the data were first purified by using an auto-encoder, thus a better representation is ensured. Then, intrusion detection and identification were enabled by using a deep neural network. The proposed model was tested through NSL-KDD, ISOT, and X-IIoTID datasets separately. It was stated in the results of the study that the recommended model attained a high rate of detecting the targeted ransomware towards the devices in the IIoT systems [26].

In a study conducted by using UNSW-NB15 and KDD99 datasets, a comparison process was conducted through various techniques such as expectation–maximization, clustering algorithm, and artificial neural network methods. In this study, metrics of false alarm rate and accuracy value were used in the evaluation of the models. It was disclosed that the expectation maximization attained an accuracy value of 78.06% and a false alarm rate of 23.79% in the KDD99 dataset. In the UNSW-NB15 dataset, the expectation–maximization achieved an accuracy value of 78.47% and a false alarm rate of 23.79%. In addition, the artificial neural networks technique obtained a false alarm rate of 21.13% and an accuracy value of 81.34% when tested on the UNSW-NB15 dataset [27].

In another study, an IDS was proposed for the IIoT. In this model, the Genetic Algorithm was used for attribute selection. The Random Forest model was used for the Genetic Algorithm fitness function. Classification methods of Extra Trees, Naive Bayes (NB), Random Forest, Linear Regression (LR), Extreme Gradient Boosting, and Decision Tree were used in the processes where intrusion detection was performed. The genetic algorithm-random forest model generated ten feature vectors in the binary classification and seven in the multi-class classification. In the implementation performed by using the UNSW-NB15 dataset, an area under the curve of 0.98 and a test accuracy of 87.61% were obtained for the binary classification. There were sixteen features in the genetic algorithm-random forest model producing these results [28].

Liu et al. [29] presented an IoT-based intrusion detection system. In this model, a heuristic method was used for the feature selection. In the classification of the intrusions, the Support Vector Machine was preferred. The Light Gradient Boosting Machine (LightGBM) algorithm formed a basis for the particle swarm optimization method used in this research. The proposed models were tested on the UNSW-NB15 dataset. False Alarm Rate and Accuracy Value were determined as the performance metrics. Considering the results, the accuracy rate of the proposed model was found as 86.68%, while the false alarm rate was found as 10.62%. These results indicate that the model proposed within the binary classification had a higher false alarm rate compared to the other studies in the literature. Researchers did not implement the model within the multi-class classification process.

A variational long short-term memory (VLSTM)-based intrusion detection system was proposed by Zhou et al. for the big data systems used in the industrial area. In this model, the features were reconstructed. Then, a selection process was performed among these new features generated. An autoencoder was used in the extraction of the features. In this way, features were extracted from the complex and large dataset. Researchers used the UNSW-NB15 dataset in this implementation. In the evaluation of the performance of the proposed model, metrics of precision, recall, false alarm rate, F1-Score, and the area under the curve were used. The variational long short-term memory method yielded an area under the curve value of 0.895, a recall value of 97.8%, a precision value of 86.0%, and an F1-Score value of 90.7%. These values are higher than some of the studies in the literature. Nevertheless, the researchers expressed that further implementations would enable obtaining better results due to the uneven distributions of the types of attacks within the dataset used in this study [30].

In another study, an extreme learning machine-based IDS was proposed. In this model, an adaptive principal component was used for the feature selection. Features selected by the adaptive principal component were presented to the extreme learning machine to perform the classification process. The proposed model was tested in the KDD and UNSW-NB15 datasets. In addition, the multi-class classification process was performed for both datasets. The accuracy value obtained over the test data was accepted as the performance metric. The proposed model achieved an accuracy rate of 81.22% in the NSL-KDD dataset, while this rate was 70.51% in the UNSW-NB15 dataset. It was expressed that better results were obtained compared to the other studies. Moreover, it was addressed that further studies are required for increasing the accuracy rate in real-life industrial control systems [31].

In another study, researchers used deep neural networks in the intrusion detection system they proposed. The aim of this study

was to design an IDS that can rapidly detect new types of attacks and be used on various platforms. Researchers used 6 different datasets in order to evaluate the performance of the methods. These were Kyoto, CICIDS 2017, KDD-Cup99, NSL-KDD, UNSW-NB15, and WSN-DS datasets. In the study taking the UNSW-NB15, which is an updated dataset, as a reference, it was determined that the deep neural network obtained an F1-Score of 79.7%, a recall of 96.3% a precision of 95.1%, and an accuracy of 76.1% in the binary classification. On the other hand, the deep neural network yielded lower results in the multi-class classification with a precision of 59.7%, an accuracy of 65.1%, a recall of 65.1%, and an F1-Score of 75.6%, compared to the binary classification [32].

In another study, researchers presented an artificial neural network based IDS for IoT networks. This model was implemented to overcome the security issue, which is a major source of concern in IoT networks. The authors considered that IoT devices are usually lacking in the capacity to perform complex calculations in ensuring security, so a machine learning-based IDS was proposed by the authors. The performance of the proposed approach was tested on the UNSW-NB15 dataset. The artificial neural network-based IDS achieved an accuracy value of 84.0%. It was not clearly expressed by the authors how the hyperparameters of the artificial neural network were adjusted to arrive at the conclusion. In addition, no feature selection method was used in the study [33].

Ketzaki proposed an IDS using the artificial neural network to ensure the security of modern communication systems. The model proposed in this research consisted of feature extraction and classification phases. Statistical analysis methods were used for the feature extraction, while the classification process was conducted through artificial neural networks. The accuracy rate obtained by the test data was used to evaluate the performance of the artificial neural network model generated. According to the obtained results, the highest rate of performance was the accuracy rate of 83.9%. The author aims to conduct studies on different models to increase the effectiveness of the implementation in the future [34].

In another study, an intrusion detection model using the J48 classifier algorithm and the support vector machine was proposed. In the study, feature selection was conducted through various methods such as the grey wolf optimizer algorithm, the firefly optimization, and the genetic algorithm. The researchers measured the effectiveness of the models implemented in the experiments by using the UNSW-NB15 dataset. The accuracy values obtained by the firefly optimization-J48, grey wolf optimizer-J48, and the genetic algorithm-J48 models were 86.037%, 85.676%, and 86.874%, respectively. In addition, the accuracy values achieved by the genetic algorithm-SVM, grey wolf optimizer-SVM, and firefly optimization-SVM were determined as 86.387%, 84.485%, and 85.429%, respectively. Although the J48 and the support vector machine methods achieved impressive results in feature selection, future studies to be conducted on large and complex datasets were recommended to be performed by using other methods such as deep learning techniques [35].

In a different study, the researchers developed a new feature selection method by using the Tabu Search algorithm and random forest algorithm together. In this method, the features were searched through the tabu search algorithm. Then, the learned features were extracted through the random forest method. In this research, the UNSW-NB15 dataset was used to evaluate the proposed method. The accuracy value and the False Positive Rate were used in evaluating the performance of the models. The tabu search-random forest method obtained a false positive rate of 3.7% and an accuracy of 83.12% when the random forest method was used in the classification process [36].

In [37], a two-stage tabu search algorithm-based IDS was proposed. In this method, minority intrusion classes were detected in the first stage and the majority intrusion classes were detected in the second stage. In this research, the Random Forest was used as the classification method. The UNSW-NB15 dataset was used for the evaluation of the proposed method. In this research, accuracy and false alarm rate metrics were used in the performance evaluation. The authors only conducted the binary classification process in this research. The results demonstrated that the proposed model obtained a false alarm rate of 15.64% and an accuracy of 85.78%. The authors stated that they aim to develop the model they proposed in future studies.

In [38], the authors proposed a model using Logistic Regression and the genetic algorithm method in the feature selection process. In the model, the classification process was conducted by using the C4.5 method. C4.5 is a tree-based algorithm. In this study, the obtained accuracy value was the main performance metric in the test data. The proposed model attained an accuracy of 81.42%. The accuracy value obtained in this research yielded lower results compared to the other studies in the literature.

In another study, an IDS using various machine learning methods was proposed. In this model, the XGBoost (extreme gradient boosting) was used as the feature extraction method. In this research, the XGBoost algorithm was used for feature extraction in the UNSW-NB15 dataset. This is an ensemble-based algorithm. In addition, the classification process was conducted through the linear regression method. The XGBoost- Linear Regression model obtained an accuracy of 72.53% and 75.51% in the multi-class and binary classification processes, respectively. The authors recommended using oversampling techniques in order to overcome the imbalance problem in the types of attacks within the UNSW-NB15 dataset [39].

In [40], the authors implemented a support vector machine-based network IDS. The system was designed in a way that represents the unique structure of IoT networks. The evaluation of this model was performed on the UNSW-NB15 dataset. The authors considered the main performance metrics of detection rate, accuracy, and false-positive rate. The result demonstrated that the SVM-based network IDS attained an accuracy rate of 85.99% for the binary classification, while this rate was 75.77% in the multi-class classification process.

In another research, a decision tree-based intrusion detection system was proposed to perform online intrusion detection by using the UNSW-NB15 dataset. The decision tree method was used for feature extraction. The information gain method determined the best thirteen attributes. In the classification process, on the other hand, the researchers used an integrated form of tree-based classifiers of CART, QUEST, C5, and CHAID. The proposed model concluded an accuracy of 84.83% for the binary classification process. The intrusion detection model proposed in this study could not detect the intrusions it did not encounter before. Authors expressed that an improvement would be conducted on this model in future studies to solve this issue [41].

The researchers presented an IDS generated by LTSM and RNN as the deep learning methods. The approach was evaluated on the UNSW-NB15 dataset. Moreover, the accuracy value was used as the performance metric. The Long Short-Term Memory method attained an accuracy of 85.42%. The researchers stated that they achieved better results in this study compared to the other studies in the literature [42].

An ensemble-based IDS was proposed by Elijah et al. [43]. The LSTM method was used as the deep learning model. The Stochastic Gradient Descent was the optimization algorithm implemented on the long short-term memory algorithm. The Rectified Linear Unit function was used in the binary classification process implemented on the LSTM layers. The researchers used the Softmax function for the multi-class classification process. The proposed model was evaluated through the UNSW-NB15 dataset. The LSTM-based

intrusion detection system obtained an accuracy value of 80.72% in the binary classification, while this value was lower with 72.26% in the multi-class classification.

In another study, an intrusion detection system using deep neural networks was proposed. In this model, residual blocks (ResBlk) were used in a way that feeds the next layer. The residual blocks involve RNNs and CNNs. Two different datasets, NSL-KDD and UNSW-NB15, were used in the study. The accuracy value was determined as the performance metric in evaluating the approach. According to the results of the study performed by using residual blocks, the intrusions were detected with an accuracy of 99.21% in the NSL-KDD dataset and 86.64% in the UNSW-NB15 dataset. The researchers expressed that more applications must be performed to increase and maintain the existing performance numbers [44].

Assiri [45] proposed a random forest- and genetic algorithm-based approach for anomaly classification. In this study, feature and parameter selection was performed through the genetic algorithm. The classification process, on the other hand, was conducted through the random forest method. In addition, the researchers only conducted the binary classification process. Precision, accuracy, and recall were the performance metrics used to evaluate the proposed model. The model attained a recall of 87.00%, a precision of 87%, and a classification accuracy of 86.70%.

In [46], an advanced genetic algorithm- and logistic regression-based intrusion detection model was proposed by the authors. This model was designed by using a multi-objective attribute selection technique. The random forest was one of the machine learning techniques used to evaluate the performance of the proposed method. The accuracy value was determined as the performance metric to measure the effectiveness of the model. The proposed model obtained an accuracy of 64.23% for the multi-class classification task.

In [47], an intrusion detection model was proposed with the aim of speedy attack discovery on industrial IoT networks. Deep auto-encoder and LSTM were used together in the proposed model for intrusion detection. The gas pipeline dataset and UNSW-NB15 dataset were used in the study. It was reported at the end of the study that the gas pipeline dataset attained a detection accuracy of 97.95% while this rate was 97.62% for the UNSW-NB15 dataset.

In [48], an intrusion detection system for IIoT networks was developed by Awotunde et al. A rule-based approach was used for the feature selection in the study, which was performed through the deep learning method. The proposed model was tested by using NSL-KDD and UNSW-NB15 datasets. It was revealed at the end of the study that accuracy rates of 99.0% and 98.9% were attained by the NSL-KDD and UNSW-NB15 datasets, respectively. The authors suggested that the method they proposed is appropriate for the detection and classification of intrusions on IIoT networks.

In [49], a machine learning-based intrusion detection system was proposed. The proposed intrusion detection system was run on X-IIoTID and ToN_IoT, which are both up-to-date datasets. F1-Score was preferred as the evaluation metric in the intrusion detection system proposed by using the XGBoosT model. It was reported at the end of the study that F1-Score rates of 99.9% and 99.87% were obtained for X-IIoTID and ToN_IoT datasets, respectively.

3. Datasets

The performances of the models proposed in this study were determined by using X-IIoTID, which is an up-to-date dataset, and UNSW-NB15, which is a dataset frequently preferred in the literature. The UNSW- NB15 [50] is an advanced dataset widely used by IDS studies. There are forty-two features listed in Table 1 in the UNSW-NB15 dataset. As presented in the list of features in Table 1, thirty-nine features are numerical and three features are categorical.

Table 1
Attributes of the UNSW-NB15 dataset.

Feature	Value	Section Feature
service	nominal	primary
state	nominal	primary
sinpkt	float	time
dpkts	int	primary
dbytes	int	primary
synack	float	time
dttl	int	primary
dload	float	primary
ct_dst_src_ltm	int	connection
sload	float	primary
ct_src_dport_ltm	int	connection
dloss	int	primary
sloss	int	primary
dur	float	primary
proto	nominal	flow
tcprtt	float	time
synacks	float	time
dinpkt	float	time
sjit	float	time
sbytes	int	primary
swin	int	content
response_body_len	int	content
stcpb	int	content
dwin	int	content
djit	float	time
dmean	int	content
trans_depth	int	content
ct_state_ttl	int	general
is_ftp_login	binary	general
sttl	int	primary
ct_ftp_cmd	int	general
ct_flw_http_mthd	int	general
dtcpb	int	content
is_sm_ips_ports	binary	general
ct_srv_src	int	connection
smean	int	content
ct_dst_ltm	int	connection
ct_dst_sport_ltm	int	connection
spkts	int	primary
ct_drc_ltm	int	connection
ackdat	float	time
ct_srv_dst	int	connection

In this study, 70% of the dataset was used in the training phase of the models, 15% in the validation phase, and 15% in the test phase. A verification process was conducted to control whether the best results were obtained in the training process. The UNSW-NB15 intrusion detection dataset contains the attacks of Reconnaissance, DoS, Analysis, Fuzzers, Exploits, Shellcode, Generic, Worms, Backdoor, and Benign [51]. The value distribution of the types of attacks in the UNSW-NB15 dataset is presented in Table 2. As is seen in Table 2, the number of types of attacks in the dataset did not demonstrate a balanced distribution. It is

Table 2
Distribution of attack values in the UNSW - NB15 dataset.

Attacks	Number
Fuzzers	18184
Backdoor	1746
Analysis	2000
Generic	40000
Shellcode	1133
Reconnaissance	10491
DoS	12264
Worms	130
Exploits	33393
Benign	56000

known that the network faced a memorization issue in types of attacks that are few in number within the datasets generated in this way in the previous studies [52]. The synthetic data generation method is used in the literature in order to prevent this issue. Through this method, the types of attacks that are few in number within the dataset balance with each other and, in this way, both a balanced dataset is obtained and the total amount of data is increased [53]. Another dataset used in the study is X-IIoTID. As an up-to-date dataset, X-IIoTID was created for the IIoT devices operating in harmony together and having a complex structure. Created with a holistic approach, this dataset is composed of traffic information and changes obtained from the devices and protocols within the IIoT network. The dataset also contains logs, resources, and alert features from various devices and links. There are 820.834 types of attacks and 68 features within the X-IIoTID dataset. The numbers of the types of attacks within the X-IIoTID dataset are presented in Table 3 [24].

4. Proposed model

Deep learning algorithms have been frequently used in industrial systems in recent times based on the increase in the amount of data. Many abstraction layers within the deep neural network transmit the input data to the output layer. It has no effect on the skill-learning process of feature selection engineering. Several statistical methods can be used to determine whether a classification is accurate or not based on the error probability according to various criteria. The focal point of deep learning is deep networks performing the classification process in the hierarchical networks of multi-layer unsupervised learning. Deep learning networks may vary according to the technology used, deep learning method, and intrusion detection method. CNNs and LSTMs were the deep learning models used in this study. Accuracy and loss metrics were used in each deep learning method to calculate the outputs of these models. In addition, accuracy, recall, F1- score, and precision were used as the performance evaluation metrics.

The temporal component feature within the RNNs and LSTM networks distinguishes these methods from other neural networks. These two models involve time consumption feature and sequence consideration by means of the temporal component. The main difference separating the RNN and LSTM from other neural networks is the fact that they consume time and consider the sequence based on their temporal components. Studies indicate that the neural network is the most productive method in designing intrusion detection systems. In recent times, nevertheless, memory networks, converters, and language tasks have taken precedence over neural networks in terms of performance rates. RNNs can be disintegrated into patches and, at the same time, extended to images that can be approached in parts. Recurrent networks enabling a certain memory form and the memory becoming a part of the human experience are the influencers in the learning process [54].

Table 3
Distribution of attack values in the X-IIoTID dataset.

Attack	X-IIoTID
Generic Scanning	50277
Scanning Vulnerability	52852
Discovering Resources	23148
Fuzzing	1313
Brute Force	47241
Dictionary	2572
Insider Malicious	17447
Reverse Shell	1016
Mitm	117
Modbus Register	5953
MQTT	23524
TCP Relay	2119
Command Control	2863
Exfiltration	22134
Ransomware	458
RDoS	141261
Fake Notification	28
False Data Injection	5094
Normal	4211417

A flow chart of deep learning models proposed for the binary and multi-class classification processes is presented in 2 In the first step, data cleaning was applied to the UNSW-NB15 and X-IIoTID datasets in the data pre-processing layer. After, datasets were normalized by using the min–max method. Then, the datasets were divided into training, validation, and testing sets. In the deep learning classifier step, CNN and LSTM classifiers were generated through the normalized dataset. Deep learning models were trained as binary and multi-class classifiers. In the evaluation phase, as the last step, the proposed IDS based on the deep learning methods was evaluated by using accuracy, recall, precision, loss, and F1-Score for both binary and multi-class classifications belonging to each model generated.

A convolutional neural network is a method of multi-layer artificial neural network that is commonly used in intrusion detection. Convolutional neural networks run a mathematical operation called convolution. Convolution is a specialized kind of linear operation. Convolutional networks are neural networks using convolution in at least one of the layers instead of general matrix multiplication [55]. The CNNs are generally composed of a convolution layer, a pooling layer, and a fully connected layer. Convolutional neural networks can learn to extract complex attributes automatically. An advanced attribute representation is obtained

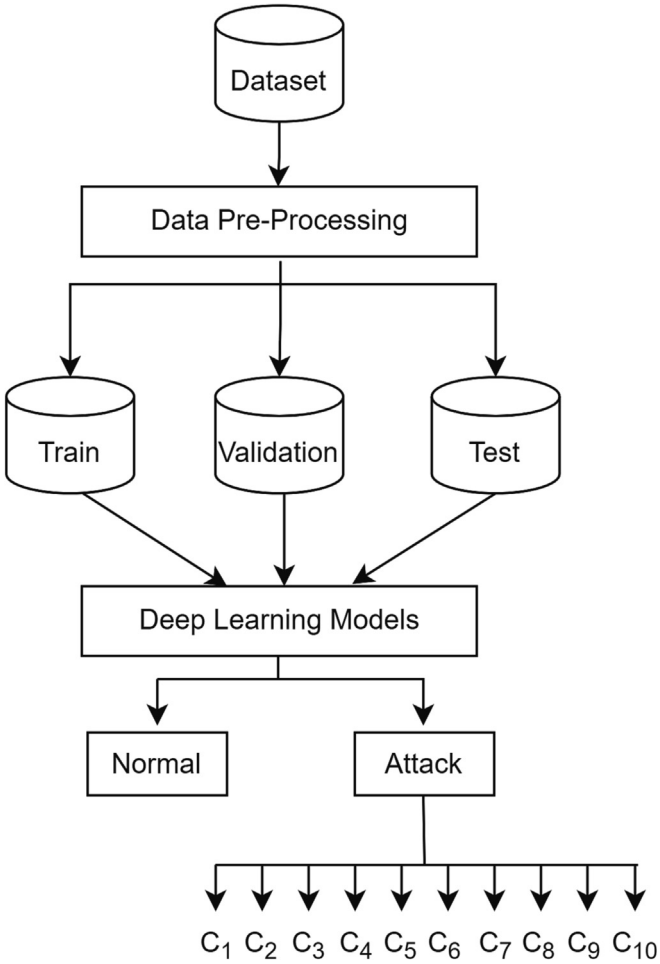


Fig. 2. Flowchart for classification.

in the convolution layer [56]. The convolution operation is presented as demonstrated in Eq. 1. The x_i^a demonstrated here is the attribute i map of the convolution layer a. The ϕ represents the activation function. k_i is the input attribute set of the layer (a-1). w_{ji}^a is the connection weight between attribute i of the convolution layer a and attribute j of the layer (a-1). b_j^a is the deviation in the related layer.

$$x_i^a = \phi\left[\sum_{i \epsilon k_i} x_j^{a-1} * w_{ji}^a + b_j^a\right] \tag{1}$$

The pooling layer follows the convolution layer. The objective of the pooling layer is to reduce the size of the attribute map. This operation ensures the identification of significant attributes, reduces data complexity, and increases the tolerance of the network against environmental changes. The pooling layer can be presented as demonstrated in Eq. 1.

$$x_i^a = \phi\left[\beta_i^a c\left(x_i^{a-1} + b_i^a\right)\right] \tag{2}$$

The c demonstrated here indicates the sub-sampling function while β indicates the weighting matrix. The classification operation is performed through the fully connected layer following the convolution layer and pooling layer. Eq. 1 demonstrates the output function of the fully connected layer.

$$y^m = \phi\left[w^m x^{m-1} + b^m\right] \tag{3}$$

The m demonstrated here indicates the layer index while y^m demonstrates the output of the fully connected layer, x^{m-1} the input of the fully connected layer, w^m the weighting coefficient, and b^m deviation [56].

LSTM cells replace the hidden units of the recurrent networks and have recurrent connections [55]. In the LSTM block, x^t indicates the input vector in time step t while h_{t-1} indicates the hidden state in time step (t-1) and c_{t-1} indicates the memory cell state in time step (t-1). These constitute the inputs of the block. The LSTM includes input, forget, and output gates.

The calculations regarding the cell state, forget, input, and output gates of the LSTM are stated in the equations below. Accordingly, the forget gate $f_i^{(t)}$ for cell i in time step t determines which information can pass through or not through a sigmoid activation function (σ) in Eq. 1. b^f, Z^f and D^f indicate deviation, input weight, and recurrent weights for the forget gates, respectively. In Eq. 1, the update on the state of cell $n_i^{(t)}$ is demonstrated while b, Z, and D indicate the deviation, input weight, and recurrent weights entering the LSTM cell, respectively. In Eq. 1, a calculation of the input gate for the cell $p_i^{(t)}$ is demonstrated and this calculation is performed in a similar way to the forget gate. In Eq. 1, $h_i^{(t)}$ indicates the hidden state while $s_i^{(t)}$ indicates the output gate. The output gate equation is demonstrated in Eq. 1 while b^0, Z^0, and D^0 indicate the deviation, input weight, and recurrent weights, respectively.

$$f_i^{(t)} = \sigma\left(b_i^f + \sum jZ_{i,j}^f x_j^{(t)} + \sum jD_{i,j}^f h_j^{(t-1)}\right) \tag{4}$$

$$n_i^{(t)} = f_i^{(t)} n_i^{(t-1)} + p_i^{(t)} \sigma\left(b_i + \sum jZ_{i,j} x_j^{(t)} + \sum jD_{i,j} h_j^{(t-1)}\right) \tag{5}$$

$$p_i^{(t)} = \sigma\left(b_i^p + \sum jZ_{i,j}^p x_j^{(t)} + \sum jD_{i,j}^p h_j^{(t-1)}\right) \tag{6}$$

$$h_i^t = \tanh\left(n_i^{(t)}\right) s_i^{(t)} \tag{7}$$

$$s_i^{(t)} = \sigma\left(b_i^0 + \sum jZ_{i,j}^0 x_j^{(t)} + \sum jD_{i,j}^0 h_j^{t-1}\right) \tag{8}$$

Processes performed in each phase of the proposed model are explained in detail as follows:

Phase 1: The first phase, pre-processing, was divided into two in itself. This resulted from the many numbers of missing values in some feature columns of the UNSW- NB15 dataset. These missing values cannot be introduced to the learning models. In this stage, all empty cells in the feature column were filled with the "0" value representing the missing value. Then, the cells stored in the memory were converted as texts; each categorical value was represented as a certain numerical value and the conversion process was performed. The normalization process of data was particularly helpful for the systems represented at various levels. Min–max normalization process was performed to assist in the generation of neural networks in a more consistent way. This method has the advantage of conducting all data connections accurately. The increasing function falls below the true value range as the min–max was added in the classification process. Nevertheless, feature values can stay within the existing range [57].

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{9}$$

Phase 2: The datasets were divided into three groups as train, validation, and test in this phase. We separated 70% of the datasets as training, 15% as validation, and 15% as the test set.

Phase 3: We proposed an intrusion detection system through three types of deep learning models based on the CNN, LSTM, and CNN + LSTM deep learning techniques. Hyperparameters implemented for the CNN method are presented in Table 4 and those implemented for the LSTM method are presented in Table 5.

The proposed CNN and LSTM and CNN + LSTM models are displayed in 3, 4, and 5. In addition, the pseudocode of the CNN + LSTM models is stated in Algorithm 1.

Phase 4: In this phase, proposed deep learning approaches were evaluated as intrusion detection methods and their performances were measured based on the determined metrics. The metrics determined to evaluate the performances of each model were accuracy, precision, recall, F1-Score, and loss. In addition, the accuracy rate was considered in detecting the types of attacks within the datasets. There was a recall task for tracing the validation loss per epoch in the training process. In case the validation loss cannot be improved for thirty epochs, the training process is stopped. Accordingly, the evaluation process will also be stopped when the training process ends.

Algorithm 1: CNN_LSTM

```
1:Input: Train_X, Train_Y
2:Hyper-Parameters: optimizer, rate,feature_layers, poolsize,
  batchsize
3:Initialize()
4:Normalization(Train_X, Train_Y)
5:Convolution_1 = Sequential ((Convolution2D(optimizer,dro
  pout,name="Conv2D_1"), MaxPooling2D(poolsize), dropout
  (rate))
6.Convolution_1.compile(Train_X, Train_Y, epochs, batchsize)
7.Convolution_1.fit(Train_X, Train_Y, epochs, batchsize)
8.Convolution_1_feature = Model(inputs,convolution_1("Con
  volution2D:).output) 9. Convolution_1_feature.predict(Trai
  n_X)
10.Lstmmodel = Sequential(Lstm(units, activation,
  recurrent_activation), flatten (units,activation)
11.Lstmmodel.compile(lossfunction, optimizer))
12.Lstmmodel.fit(Convolution_1_feature, Train_Y, batchsize,
  epochs))
```

4.1. Evaluation measures

The accuracy, F1-Score, recall, and precision metrics were used to evaluate the performance of the proposed approach. TP indicates the number of attacks and TN indicates the number of normal traffic classified accurately. FP indicates the number of traffics, which are actually the normal data but misclassified and regarded as attacks. FN indicates the number of attacks misclassified as normal traffic [58].

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{10}$$

$$Recall = \frac{TP}{FN + TP} \tag{11}$$

$$Precision = \frac{TP}{FP + TP} \tag{12}$$

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{13}$$

5. Experimental analysis and results

In this section, the results of the binary and multi-class classifications obtained by the IDS generated by using CNN, LSTM, and CNN + LSTM methods were explained. In addition, the accuracy rates obtained in the UNSW-NB15 and X-IIoTID datasets in this study were compared to those obtained in previous studies. Finally, the accurate detection rate of the types of attacks in the datasets used in this study was presented.

Results obtained for both the binary and multi-class classification processes performed by using LSTM in the UNSW-NB15 dataset are presented in Table 6. The recurrent neural networks must be accurately positioned in order to receive optimal results from the performance metrics in the dataset generated for binary and multi-class labeling. This process shortens the training period of the network as well as decreasing the loss value. In this regard, the performance of the long short-term memory method in the binary classification was determined as superior to the performance in the multi-class classification. The accuracy rate for the test set was found 91.14% in the binary classification, while it was found 91.10% in the multi-class classification. The test loss value was 6.41% in both binary and multi-class classifications. In the LSTM model, the validation accuracy value was determined as 91.05% in the binary classification and 91.08% in the multi-class classification, while this rate was 91.12% in the binary classification and 91.10% in the multi-class classification for the training. The training loss value was 11.83% for both classification processes, while this rate was 11.72% in the binary classification and 11.70% in the multi-class classification for the validation. The result of 100 epochs was received in the training process.

Results obtained for both the binary and multi-class classification processes performed by using the convolutional neural network (CNN) in the UNSW-NB15 dataset are presented in Table 7. The parameters of the convolutional neural network must be accurately adjusted in order to receive optimal results from the performance metrics in the dataset generated for binary and multi-class labeling. In this regard, the performance of the convolutional neural network in the binary classification was determined as equal to the performance in the multi-class classification. The accuracy rate for the test set was determined as 90.09% for both binary and multi-class classifications. The test loss value was 8.30% in the binary classification, while this rate was 8.34% for the multi-class classification. The validation loss value was determined as 12.70% and the training loss value was determined as 12.75% for both binary and multi-class classifications. The validation accuracy rate was 90.05% in the binary classification, while this rate was 90.08% for the multi-class classification. The training accuracy rate, on the other hand, was determined as 90.43% for both binary and multi-class classifications.

Results of the binary and multi-class classification processes conducted on the UNSW-NB15 dataset by using the CNN + LSTM model are presented in Table 8. The CNN + LSTM model attained an accuracy of 93.21% for binary classification and 92.9% for multi-class classification in the test set. The missing value in the test set was determined as 6.21% and 6.28% for binary and multi-class classifications, respectively. The accuracy value was deter-

Table 4
CNN parameters.

Hyperparameter	Value
Optimizer	Gradient Descent
Dropout	0.4
Layers	2
Feature Layer	Fully Connected
Classify Layer	Fully Connected
Learning Rate	0.01, 0.001
Epoch	100

Table 5
LSTM parameters.

Hyperparameter	Value
Activation Function in Hidden Layer	ReLu
Epoch	100
Batch Size	120
Optimizer	Gradient Descent
Layers	3
Number of Neurons	256, 128, 64
Activation Function in Output Layer	Softmax

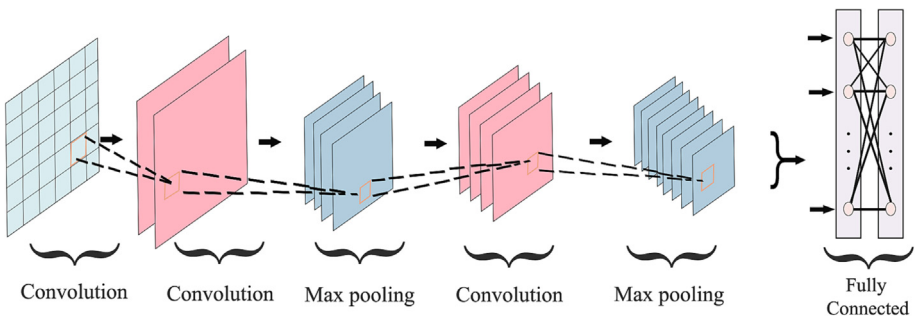


Fig. 3. Proposed CNN architecture.

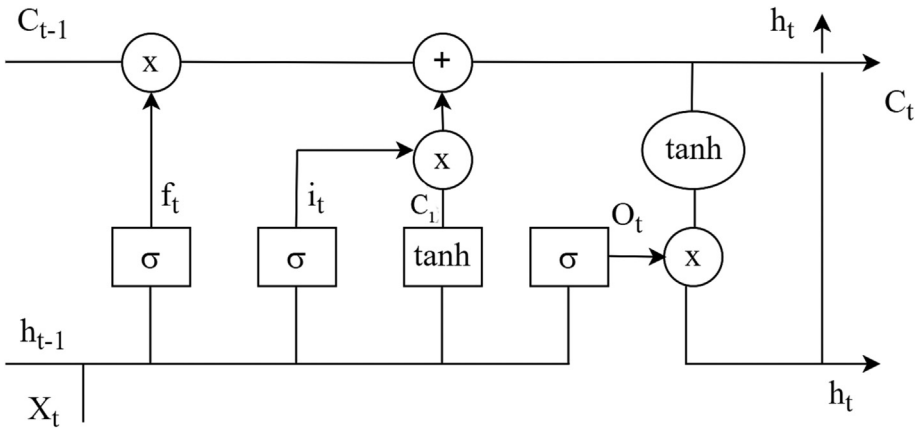


Fig. 4. General LSTM architecture.

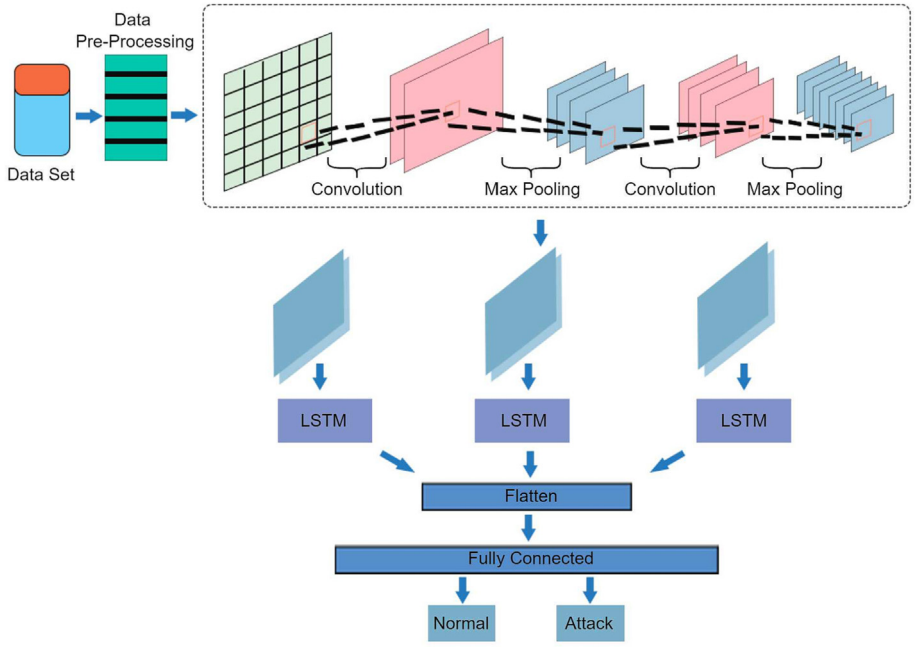


Fig. 5. Proposed CNN + LSTM architecture.

Table 6
Results obtained with the LSTM model.

Metrics	Binary Classification	Multi-Class Classification
Accuracy in Training	91.12%	91.10%
Accuracy in Validation	91.05%	91.08%
Accuracy in Test	91.14%	91.10%
Loss in Training	11.83%	11.83%
Loss in Validation	11.72%	11.70%
Loss in Test	6.41%	6.41%

mined as 93.11% for both binary and multi-class classification in the validation set. The missing value in the validation set was determined as 5.89% and 5.98% for binary and multi-class classifications, respectively. The accuracy value in the training set was determined as 93.84% and 93.26% for binary and multi-class classifications, respectively. The missing value in the training set was determined as 5.19% and 6.07% for binary and multi-class classifications, respectively.

Table 7
Results obtained with the CNN model.

Metrics	Binary Classification	Multi-Class Classification
Accuracy in Training	90.43%	90.43%
Accuracy in Validation	90.05%	90.08%
Accuracy in Test	90.09%	90.09%
Loss in Training	12.75%	12.75%
Loss in Validation	12.70%	12.70%
Loss in Test	8.30%	8.34%

Table 8
Results obtained with the CNN + LSTM model.

Metrics	Binary Classification	Multi-Class Classification
Accuracy in Training	93.84%	93.26%
Accuracy in Validation	93.11%	93.11%
Accuracy in Test	93.21%	92.90%
Loss in Training	5.19%	6.07%
Loss in Validation	5.89%	5.98%
Loss in Test	6.21%	6.28%

The accuracy rates obtained in the classification of the types of attacks within the used datasets are presented in 6 and 7. As is seen in 6, the top three types of attacks detected with the optimal accuracy rate in the UNSW-NB15 dataset are Benign, Reconnaissance, and Generic. Analyzing 6 and Table 2 carefully together, it is observed that the accurate detection rates of the types of attacks that are few in numbers are lower compared to the types of attacks that are high in numbers. It is important that the amount of data used is large and the types of attacks within the dataset are distributed in a balanced way in order to receive better results in intrusion detection systems developed by using deep learning methods.

In the X-IIoTID dataset, the CNN model attained an accuracy of 99.26% for multi-class classification and 99.15% for binary classification in the test set. The missing value was determined as 0.41% for both binary and multi-class classification in the test set. The CNN model attained an accuracy of 99.18% for binary classification and 99.09% for multi-class classification in the validation set. The missing value was determined as 0.27% for binary classification and 0.32% for multi-class classification in the validation set. Moreover, the CNN approach attained an accuracy of 99.11% for both binary and multi-class classification in the training set. The missing value was determined as 0.56% for binary classification and 0.59% for multi-class classification in the training set.

In the X-IIoTID dataset, the other proposed model, LSTM, attained an accuracy of 99.05% for binary classification and 98.91% for multi-class classification in the test set. The accuracy value in the validation set attained by the LSTM model was determined as 98.97% and 99.02% for binary and multi-class classifications, respectively. The LSTM model attained an accuracy of 98.99% for both binary and multi-class classification in the training set. The missing value in the test set was determined as 0.52% and 0.92% for binary and multi-class classifications, respectively. The LSTM model yielded a missing value of 0.78% for binary classification and 0.72% for multi-class classification in the validation set. In the training set, on the other hand, the missing value was determined as 0.75% for both binary and multi-class classification.

Finally, the CNN + LSTM model proposed by ourselves was run on the X-IIoTID dataset. It attained an accuracy of 99.84% for binary classification and 99.80% for multi-class classification in the test set. The missing value of the CNN + LSTM model was determined as 0.12% for both binary and multi-class classification in the test set. An accuracy of 99.78% was attained for both binary and multi-class classification in the validation set while the missing value was determined as 0.14% for binary classification and 0.15% for multi-class classification in the same set. In the training set, on the other hand, an accuracy of 99.81% was attained for binary classification while this rate was 99.79% for multi-class classification. The missing value was determined as 0.11% for binary classification and 0.19% for multi-class classification in the training set.

The values obtained for the binary and multi-class classification procedures in the X-IIoTID dataset through the CNN model are presented in Table 9; those obtained through the LSTM model are presented in Table 10, and those obtained through the CNN + LSTM model are presented in Table 11.

The detection accuracy values obtained for all types of attacks for the multi-class classification procedure in the X-IIoTID dataset are presented in 7. The proposed CNN + LSTM model attained the highest detection accuracy in all types of attacks in this dataset generated specifically for the IIoT networks.

The results obtained from both the studies conducted by using the UNSW-NB15 and X-IIoTID datasets in the literature and the models we proposed are presented in Table 12 and Table 13.

In the UNSW-NB15 dataset, the LSTM model attained a recall value of 91.08%, a precision value of 90.98%, and an F1-Score value of 91.02% for binary classification while attaining a recall value of 91.04%, a precision value of 91.01%, and an F1-Score value of 91.02% for multi-class classification in the training set. In the X-IIoTID dataset, the LSTM model attained a recall value of 99.01%, a precision value of 99.03%, and an F1-Score value of 99.01% for

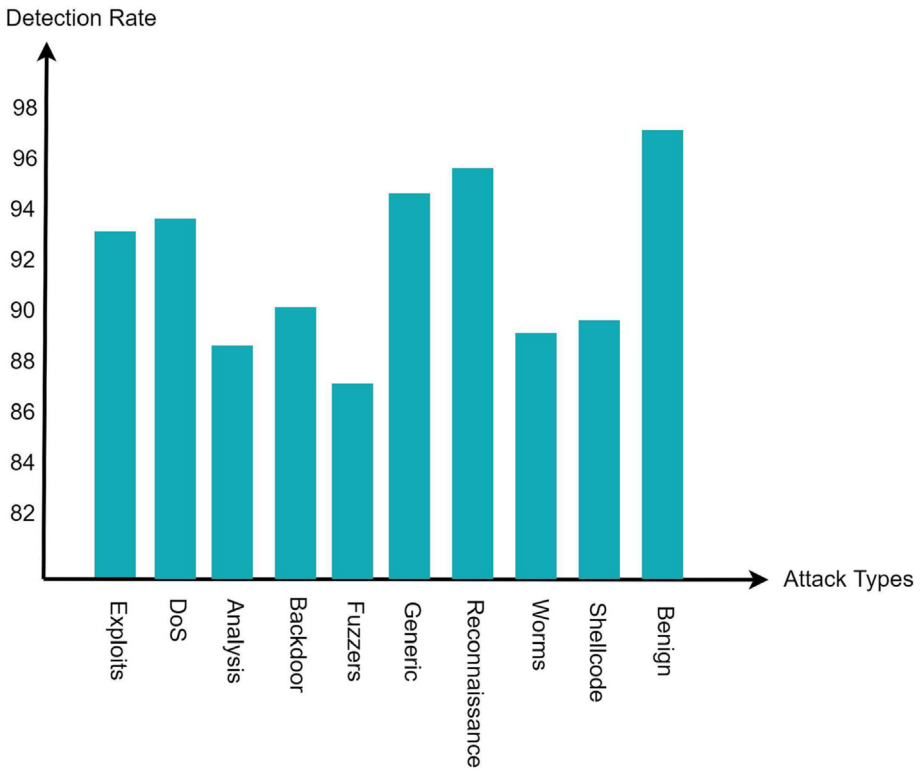


Fig. 6. Detection rate of attack types.

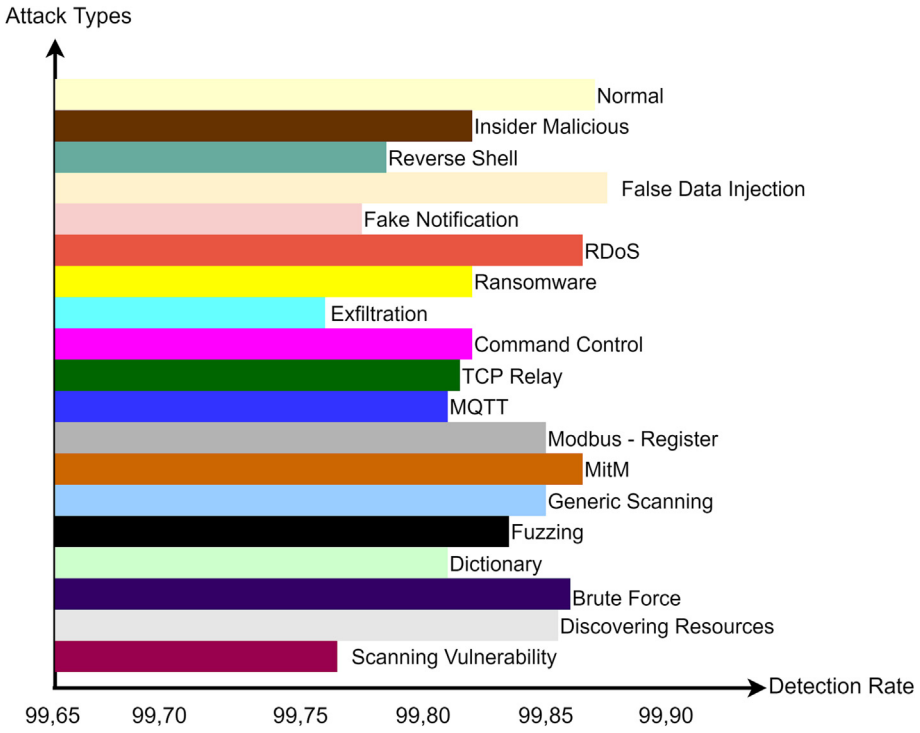


Fig. 7. Detection rate of attack types.

Table 9
Results obtained with the CNN model.

Metrics	Binary Classification	Multi-Class Classification
Accuracy in Training	99.11%	99.11%
Accuracy in Validation	99.18%	99.09%
Accuracy in Test	99.10%	99.26%
Loss in Training	0.56%	0.59%
Loss in Validation	0.27%	0.32%
Loss in Test	0.41%	0.41%

Table 10
Results obtained with the LSTM model.

Metrics	Binary Classification	Multi-Class Classification
Accuracy in Training	98.99%	98.99%
Accuracy in Validation	98.97%	99.02%
Accuracy in Test	99.05%	98.91%
Loss in Training	0.75%	0.75%
Loss in Validation	0.78%	0.72%
Loss in Test	0.52%	0.92%

Table 11
Results obtained with the CNN + LSTM model.

Metrics	Binary Classification	Multi-Class Classification
Accuracy in Training	99.81%	99.79%
Accuracy in Validation	99.78%	99.78%
Accuracy in Test	99.84%	99.80%
Loss in Training	0.11%	0.19%
Loss in Validation	0.14%	0.15%
Loss in Test	0.12%	0.12%

Table 12
Performance comparison of our proposed models with related works in the UNSW-NB15 dataset.

Paper	Model	Binary Classification	Multi-Class Classification
27	ANN	81.34%	–
28	GA-RF	87.61%	–
28	GA-ET	–	77.64%
29	PSO-GBM	86.68%	–
30	VLSTM	–	–
31	ELM	–	70.52%
32	DL-DNN	76.1%	65.10%
33	ANN	84.0%	–
34	ANN	83.9%	–
35	GA-SVM	86.38%	–
35	GWO-SVM	84.48%	–
35	FFA-SVM	85.42%	–
36	TS-RF	83.12%	–
37	IG-TS	85.78%	-
38	GA-DT	81.42%	–
39	XGB-LR	75.51%	72.53%
40	SVM	85.99%	75.77%
41	IG-TREE	84.83%	–
42	DL-LSTM	85.42%	–
43	DL-LSTM	80.72%	72.26%
44	CNN-RNN	86.64%	–
45	GA-RF	86.70%	–
46	GA-RF	–	64.23%
47	AE-LSTM	97.62%	–
48	DL	98.9%	–
Proposed model	CNN	90.09%	90.09%
Proposed Model	LSTM	91.14%	91.10%
Proposed Model	CNN + LSTM	93.21%	92.90%

binary classification while attaining a recall value of 98.82%, a precision value of 98.86%, and an F1-Score value of 98.84% for multi-class classification in the training set.

In the UNSW-NB15 dataset, the CNN model attained a recall value of 90.07%, a precision value of 90.06%, and an F1-Score value of 90.06% for binary classification while attaining a recall value of 90.05%, a precision value of 90.04%, and an F1-Score value of 90.04% for multi-class classification in the training set. In the X-IIoTID dataset, the CNN model attained a recall value of 99.05%, a precision value of 99.02%, and an F1-Score value of 99.03% for binary classification while attaining a recall value of 99.14%, a preci-

Table 13
Performance comparison of our proposed models with related works in the X-IIoTID dataset.

Paper	Model	Binary Classification	Multi-Class Classification
24	DL-ML	99.54%	99.45%
25	DL	99.79%	–
26	CDAE-DNN	98.33%	97.21%
49	XGBoosT	99.9%	–
Proposed model	CNN	99.15%	99.26%
Proposed Model	LSTM	99.05%	98.91%
Proposed Model	CNN + LSTM	99.84%	99.80%

sion value of 99.18%, and an F1-Score value of 99.15% for multi-class classification in the training set.

In the UNSW-NB15 dataset, the hybrid CNN + LSTM model attained a recall value of 93.10%, a precision value of 92.91%, and an F1-Score value of 93.00% for binary classification while attaining a recall value of 92.45%, a precision value of 92.74%, and an F1-Score value of 92.59% for multi-class classification in the training set. In the X-IIoTID dataset, the hybrid CNN + LSTM model attained a recall value of 99.55%, a precision value of 99.67%, and an F1-Score value of 99.60% for binary classification while attaining a recall value of 99.72%, a precision value of 99.38%, and an F1-Score value of 99.54% for multi-class classification in the training set.

According to the results obtained, the CNN + LSTM model attained a higher accuracy value for binary and multi-class classification processes in the UNSW-NB15 and X-IIoTID datasets compared to both other studies in the literature and the other two models used in this study, CNN and LSTM. As a result of the tests conducted in the study, it was observed that the accuracy values obtained in the X-IIoTID dataset, which is one of the latest datasets generated to detect intrusions in the IIoT network, were higher than the values in the UNSW-NB15 dataset. Consequently, it was concluded that the X-IIoTID is a well-defined dataset for complex IIoT networks.

6. Conclusion

In this article, an IDS based on deep learning models was proposed to detect intrusion in an IIoT environment. The study was performed by using CNN, LSTM, and CNN + LSTM methods as intrusion detection systems. Deep learning models were implemented by using the X-IIoTID and UNSW-NB15 datasets in order to determine the abnormal patterns in this research. No additional machine learning method was used in the feature selection process; deep learning methods performed this task alone. The experimental results of the proposed deep learning models were superior to the previous methods developed in the same dataset. These results demonstrate the superiority of deep learning methods in detecting abnormal events within large and complex datasets.

In order to avoid possible memorization problems in future studies, synthetic data generation into the used dataset is considered. It is planned to improve the proposed framework by retraining our network with the new dataset generated following the generation of synthetic data. In addition, it is planned to increase the detection accuracy of the model in the up-to-date datasets by using machine learning methods for feature selection.

CRediT authorship contribution statement

Hakan Can Altunay: Conceptualization, Methodology, Software. **Zafer Albayrak:** Data curation, Writing - original draft.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Centenaro, F. Granelli, L. Vangelista, A survey on technologies, standards and open challenges in satellite iot, IEEE Commun. Survey Tutor. 23 (2021) 1693–1720, https://doi.org/10.1109/COMST.2021.3078433.

[2] L. Aversano, M.L. Bernardi, M. Cimitile, R. Pecori, A systematic review on deep learning approaches for iot security, Comput. Sci. Rev. 40 (2021), https://doi.org/10.1016/j.cosrev.2021.100389 100389.

[3] B. Valeske, A. Osman, F. Römer, R.as iiot elements of industry 4.0, Res. Nondestr. Eval. 31 (2020) 340–369, https://doi.org/10.1080/09349847.2020.1841862.

[4] C. Ozarpa, M. Aydin, I. Avci, International security standards for critical oil, gas, and electricity infrastructures in smart cities: A survey study, In the Proceedings of the third International Conference on Smart City Applications (2021) 1167–1179.

[5] Z. Lv, L. Qiao, A.K. Singth, Q. Wang, Ai-empowered iot security for smart cities, ACM Trans. Internet Technol. 21 (2021) 1–21, https://doi.org/10.1145/3406115.

[6] C.C. Andrei, G. Tudor, M.A. Calin, Industrial internet of things (iiot) integration in power grids, 9th International Confere Tschuncky, Next generation nde sensor systems nce on Modern Power Systems (MPS), IEEE Romania.

[7] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, Survey of intrusion detection systems: techniques, datasets, and challenges, Cybersecurity 2 (2019) 1–22, https://doi.org/10.1186/s42400-019-0038-7.

[8] W. Qin, S. Chen, M. Peng, Recent advances in industrial internet: insights and challenges, Digital Commun. Networks 6 (2020) 1–13, https://doi.org/10.1016/j.dcan.2019.07.001Get.

[9] X.D. Zhang, Machine learning, a matrix algebra approach to artificial intelligence, Springer 6 (2020) 223–440.

[10] S. Pengfei, L. Pengju, L. Qi, L. Chenxi, L. Xiangling, H. Ruochen, C. Jinpeng, Dl-ids: Extracting features using cnn-lstm hybrid network for intrusion detection system, Secur. Commun. Networks (2020) 11, https://doi.org/10.1155/2020/8890306.

[11] A.A. Aldweesh, A. Derhab, A.Z. Emam, Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues, Knowl.-Based Syst. 189 (2020), https://doi.org/10.1016/j.knosys.2019.105124 105124.

[12] A.B. Oncul, Y. Celik, A hybrid deep learning model for classification of plant transcription factor proteins, Signal Image and Video Processing Springer (2022) 1–7, https://doi.org/10.1007/s11760-022-02419-5.

[13] M.S. Alabadi, Y. Celik, Anomaly detection for cyber-security based on convolutional neural network: A survey, International Congress on Human-Computer Interaction, Optimization and Robotic Applications, IEEE (2020) 1–14. https://doi.org:10.1109/HORA49412.2020.9152899.

[14] K. Suthar, Q.P. He, Multiclass moisture classification in woodchips using iiot wi-fi and machine learning techniques, Comput. Chem. Eng. 154 (2021), https://doi.org/10.1016/j.compchemeng.2021.107445 107445.

[15] M. Shafiq, Z. Tian, A.K. Bashir, X. Du, M. Guizani, Iot malicious traffic identification using wrapper-based feature selection mechanisms, Comput. Secur. 94 (2020) 161–175, https://doi.org/10.1016/j.cose.2020.101863.

[16] X. Zhiang, G. Yijia, C. Chinmay, H. Qiaozhi, C. Shengbo, Y. Keping, A simple federated learning-based scheme for security enhancement over internet of medical things, IEEE Journal of Biomedical and Health Informatics. DOI 10.1109/JBHI.2022.3187471.

[17] I. Avci, C. Ozarpa, Machine learning applications and security analysis in smart cities, Machine Learning for Smart Environments/Cities (2022) 183–197.

[18] P.P. Zhang, C. Wang, C. Jiang, Z. Han, Deep reinforcement learning assisted federated learning algorithm for data management of iiot, IEEE Trans. Industr. Inf. 17 (2021) 8475–8484. https://1doi.org/10.1109/TII.2021.3064351.

[19] G. Vallathan, A. John, C. Thirumalai, S. Mohan, G. Srivastava, J.C.W. Lin, Suspicious activity detection using deep learning in secure assisted living iot environments, J. Supercomputing 77 (2021) 3242–3260, https://doi.org/10.1007/s11227-020-03387-8.

[20] V. Kumar, A.K. Das, D. Sinha, Statistical analysis of the unswnb15 dataset for intrusion detection, in: Computational intelligence in pattern recognition, Springer, 2020, pp. 279–294, https://doi.org/10.1007/978-981-13-9042-5_24.

[21] S. Bagui, M. Walauskis, R. DeRush, H. Praviset, S. Boucugnani, Spark configurations to optimize decision tree classification on unsw-nb15, Big Data and Cognitive Computing 6. doi: 10.3390/bdcc6020038.

[22] I.I. Dutt, S. Borah, I.K. Maitra, Pre-processing of kdd99 & unsw- nb network intrusion datasets, Turkish J. Comput. Math. Educ. 12 (2021) 1762–1776. https://doi.org/10.17762/turcomat.v12i11.6111.

[23] M.A. Ferrag, L. Maglaras, S. Moschoyiannis, H. Janicke, Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study, J. Inform. Secur. Appl. 50 (2020), https://doi.org/10.1016/j.jisa.2019.102419 102419.

[24] M.A.M.A. Hawawreh, E. Sitnikova, N. Aboutorab, X-iiotid: A connectivity-agnostic and device-agnostic intrusion data set for industrial internet of

things, IEEE Internet Things J. 9 (2022) 3962–3977, https://doi.org/10.1109/JIOT.2021.3102056.
[25] A. Makkar, T.W. Kim, A.K. Singh, J. Kang, J.H. Park, Secureiiot environment: Federated learning empowered approach for securing iiot from data breach, IEEE Transactions on Industrial Informatics (Early Access. doi: 10.1109/TII.2022.3149902.
[26] M.A. Hawawreh, E. Sitnikova, N. Aboutorab, Asynchronous peer-to-peer federated capability-based targeted ransomware detection model for industrial iot, IEEE Access 9 (2021) 148738–148755, https://doi.org/10.1109/ACCESS.2021.3124634.
[27] N. Moustafa, J. Slay, The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set, Inform. Secur. J.: Global Perspective 25 (2016) 18–31, https://doi.org/10.1080/19393555.2015.1125974.
[28] S.M. Kasongo, An advanced intrusion detection system for iiot based on ga and tree based algorithms, IEEE Access 9 (2021) 113199–113212, https://doi.org/10.1109/ACCESS.2021.3104113.
[29] J. Liu, D. Yang, M. Lian, M. Li, Research on intrusion detection based on particle swarm optimization in iot, IEEE Access 9 (2021) 38254–38268, https://doi.org/10.1109/ACCESS.2021.3063671.
[30] X. Zhou, Y. Hu, W. Liang, J. Ma, Q. Jin, Variational lstm enhanced anomaly detection for industrial big data, IEEE Trans. Industr. Inf. 17 (2021) 3469–3477, https://doi.org/10.1109/TII.2020.3022432.
[31] J. Gao, S. Chai, B. Zhang, Y. Xia, Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis, Energies 12 (2019) 1223, https://doi.org/10.3390/en12071223.
[32] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, S. Venkatraman, Deep learning approach for intelligent intrusion detection system, IEEE Access 7 (2019) 41525–41550, https://doi.org/10.1109/ACCESS.2019.2895334.
[33] S. Hanif, T. Ilyas, M. Zeeshan, Intrusion detection in iot using artificial neural networks on unsw-15 dataset, IEEE 16th International Conference Smart Cities, Improving Quality of Life Using ICT & IoT AI (HONET-ICT) (2019) 152–156.
[34] E. Ketzaki, A. Drosou, S. Papadopoulos, D. Tzovaras, A lightweighted ann architecture for the classification of cyber-threats in modern communication networks, 10th International Conference Networks of the Future (NoF (2019) 17–24.
[35] O. Almomani, A feature selection model for network intrusion detection system based on pso, gwo, ffa and ga algorithms, Symmetry 12 (2020) 1046, https://doi.org/10.3390/sym12061046.
[36] A. Nazir, R.A. Khan, A novel combinatorial optimization based feature selection method for network intrusion detection, Comput. Secur. 102 (2021), https://doi.org/10.1016/j.cose.2020.102164 102164.
[37] W. Zong, Y.W. Chow, W. Susilo, A two-stage classifier approach for network intrusion detection, in: International Conference Information Security Practice and Experience Cham, Springer, 2018, pp. 329–340, https://doi.org/10.1109/ACCESS.2021.3063671.
[38] C. Khammassi, S. Krichen, A ga-lr wrapper approach for feature selection in network intrusion detection, Comput. Secur. 70 (2017) 255–270, https://doi.org/10.1016/j.cose.2017.06.005.
[39] S.M. Kasongo, Y. Sun, Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset, Journal of big data 7. doi: 10.1186/s40537-020-00379-6.
[40] D.D. Jing, H.B. Chen, Svm based network intrusion detection for the unsw-nb15 dataset, 2019 IEEE 13th International Conference on ASIC (ASICON) (2019) 38254–38268. doi: 10.1109/ASICON47005.2019.8983598.
[41] V.V. Kumar, D. Sinha, A.K. Das, S.C. Pandey, R.T. Goswami, An integrated rule based intrusion detection system: Analysis on unswnb15 data set and the real time online dataset, Cluster Comput. 23 (2020) 1397–1418, https://doi.org/10.1007/s10586-019-03008-x.
[42] A. Aleesa, M. Younis, A.A. Mohammed, S.N.,Deep intrusion detection system with enhanced unsw-nb15 dataset based on deep learning techniques, J. Eng. Sci. Technol. 16 (2021) 711–727.
[43] A.V. Elijah, A. Abdullah, N. Jhanjhi, M. Supramaniam, B. Abdullateef, Ensemble and deep-learning methods for two-class and multi-attack anomaly intrusion detection: An empirical study, Int. J. Adv. Comput. Sci. Appl. 10 (2019) 520–528.
[44] P.P. Wu, H. Guo, N. Moustafa, Pelican: A deep residual network for network intrusion detection, 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W) (2020) 55–62.
[45] A. Assiri, Anomaly classification using genetic algorithm-based random forest model for network attack detection, Comput. Mater. Continua 66 (2021) 767–778. https://doi:10.32604/cmc.2020.013813.
[46] C. Khammassi, S. Krichen, A nsga2-lr wrapper approach for feature selection in network intrusion detection, Comput. Netw. 172 (2020), https://doi.org/10.1016/j.comnet.2020.107183 107183.
[47] A.K. Izhar, K. Marwa, P. Dechang, K. Nasrullah, H. Yasir, S. Hatem, Enhancing iiot networks protection: A robust security model for attack detection in internet industrial control systems, Ad Hoc Netw. 134 (2022), https://doi.org/10.1016/j.adhoc.2022.102930 102930.
[48] B.A. Joseph, C. Chinmay, E.A. Abidemi, Intrusion detection in industrial internet of things networkbased on deep learning model with rule-based feature selection, Wireless Commun. Mobile Comput. (2021) 17, https://doi.org/10.1155/2021/7154587.
[49] L. Thi-Thu-Huong, E. Yustos, K. Howon, Xgboost for imbalanced multiclass classification-based industrial internet of things intrusion detection systems, Sustainability 14. doi: 10.3390/su14148707.
[50] M.A. Omari, M. Rawashdeh, F. Qutaishat, M. Alshira'H, N. Ababneh, An intelligent tree-based intrusion detection model for cyber security, Journal of Network and Systems Management 29. doi: 10.1007/s10922-021-09591-y.
[51] S. Moualla, K. Khorzom, A. Jafar, Improving the performance of machine learning-based network intrusion detection systems on the unsw- nb15 dataset, Computational Intelligence and Neuroscience. doi: 10.1155/2021/5557577.
[52] H.C. Altunay, Z. Albayrak, Network intrusion detection approach based on convolutional neural network, European, J. Sci. Technol. 26 (2021) 22–29. https://doi.org/10.31590/ejosat.954966.
[53] S.I. Popoola, B. Adebisi, R. Ande, M. Hammoudeh, K. Anoh, A.A. Atayero, An integrated rule based intrusion detection system: Analysis on unswnb15 data set and the real time online dataset, Sensors 21 (2021) 1397–1418, https://doi.org/10.3390/s21092985.
[54] N. Park, H.K. Ahn, Multi-layer rnn based short-term photovoltaic power forecasting using iot dataset, AEIT International Annual Conference (AEIT), IEEE 23. https://doi.org/10.23919/AEIT.2019.8893348.
[55] I. GoodFellow, Y. Bengio, A, Courville, Deep Learning, 2016.
[56] A. Zhang, Z. Lipton, M. Li, A. Smola, Dive into Deep Learning, 2021.
[57] N.M. Nawi, W.H. Atomi, M.Z. Rehman, The effect of data pre-processing on optimized training of artificial neural networks, Procedia Technol. 11 (2013) 32–39, https://doi.org/10.1016/j.protcy.2013.12.159.
[58] X. Deng, Q. Liu, Y. Deng, S. Mahadevan, An improved method to construct basic probability assignment based on the confusion matrix for classification problem, Inf. Sci. 340–341 (2016) 250–261, https://doi.org/10.1016/j.ins.2016.01.033.

Hakan Can ALTUNAY received the B.S. degree in electronic teacher from Sakarya University, Sakarya, Turkey, in 2006. In 2014, he completed his master's degree in Sakarya University, Institute of Science and Technology, Department of Electronics – Computer Education. He is still continuing his doctorate education in the department of computer engineering at Karabuk University. His research interests include industrial IoT security, SCADA security, network security, data mining and artificial intelligence.

Zafer ALBAYRAK received the Ph.D degree from Department of Computer Science, Sakarya University, Sakarya, Turkey in 2014. He is currently Assistant Professor in Department of Computer Engineering, Sakarya University of Applied Sciences. His main research area is wireless network and network security.