# Machine Learning and Deep Learning-based Intrusion Detection system for IoT sensors and networks

Mahmoud Aloulou

**Abstract:**
With the growing number of IoT-connected devices, the number of IoT attacks exploded and the development of Intrusion Detection Systems became necessary to overcome the security problems in the IoT networks. A lot of research papers/studies have succeeded to develop machine learning and deep learning models that are able to detect attacks and differentiate them from normal traffic. In the following, we will show the results of a machine learning model namely Random Forrest, and a 1D-CNN-LSTM deep learning model in terms of accuracy, recall, precision, and F1 score.
**keywords: IDS, Machine/Deep Learning, security attacks, classification**

## Introduction

Recently, the number of IoT applications has increased extensively, especially applications like smart houses, smart cities, fitness-care programs, and wearables. This is why the number of connected devices is expected to jump from 27 billion in 2017 to 125 billion in 2030.

Those IoT devices exchange a lot of confidential information and data which explains the increasing number of IoT attacks from 10.3 million attacks in 2017 to 32.7 million attacks in 2018 which equals a 217.5% increase. The problem that security analysts are facing is the fact that attack detection in IoT is more difficult than in the past and even advanced techniques like cryptography have a hard time identifying threats. In fact, IoT suffers from weak network protocols which explain the fact that some attacks remain undetected for a long time. Hence the necessity of an Intrusion Detection System (IDS) that is capable of detecting and classifying attacks in the fastest time possible and in an automated manner.
An **Intrusion Detection System** (IDS) is a surveillance system that detects suspicious attack activity and generates alerts when detected. Based on these warnings A Security Operations Center (SOC) analyst or incident responder can investigate the issue and take appropriate measures to eliminate the threat.

## Datasets, Machine Learning, Deep Learning, and Classification metrics

There is a big number of available datasets like CICIDS 2018, NSL-KDD, UNSW-NB, Botnet (BOT-IoT), CSE-CIC-IDS 2018, and also the TON-IoT datasets have been used when we checked the bibliography to classify normal traffic and attacks.
Plenty and different machine learning algorithms and deep learning models were used to build IDS(s): when reading about past research papers CNN, RNN, LSTM, and AE were used concerning the deep learning models. Algorithms like KNN, SVM, decision trees, Random Forrest and Adaboost were used when speaking about machine learning.
The results of the previous studies show that the deep learning model performs much better most of the time.

Metrics used during the evaluation are not only the classic ones like accuracy, recall, precision, and F1 score but also metrics like detection rate(DR= TP/(TP+FN)) and false alarm rate (FAR=FP/(FP+TP)) were used in some studies with the aim being to maximize the detection rate DR and minimize the false alarm rate FAR.

Results-wise, using the UNSW-NB15 dataset researchers were able to classify **9** different attack types with an accuracy of 99.76% and even more for other models. Using the NSL-KDD researchers were able to classify 5 types of traffic: normal and 4 other attacks (DOS, R2L, probe, and U2R) with 99.02 accuracy and high precision, recall, and F1 score.

## Data exploration, preprocessing, and model development

In the sequel, we will use 2 CSV files from the TON-IoT datasets: the GPS tracker one and the Modbus one.

### The GPS Tracker dataset

It consists of a CSV file containing 58960 traffic recordings with 7 columns ( 3 time-related information columns + latitude + longitude and label and type where the label is either 0 if the traffic is normal or 1 if it’s an attack and the type column, is a string which can be one of these strings: 'normal', 'backdoor', 'ddos', 'password', 'injection', 'ransomware', 'xss' , 'scanning'. So using the dataset for classification can help us classify if the traffic is normal or an attack with 7 different attack possibilities.

The dataset distribution is 35000 normal traffic and 23960 attack recordings. The dataset distribution considering the different attack types can be seen in **Table 1** and graph below:

| Traffic type | count |
|---|---|
| normal | 35000 |
| ddos | 5000 |
| injection | 5000 |
| backdoor | 5000 |
| password | 5000 |
| ransomware | 2833 |
| xss | 577 |
| scanning | 550 |

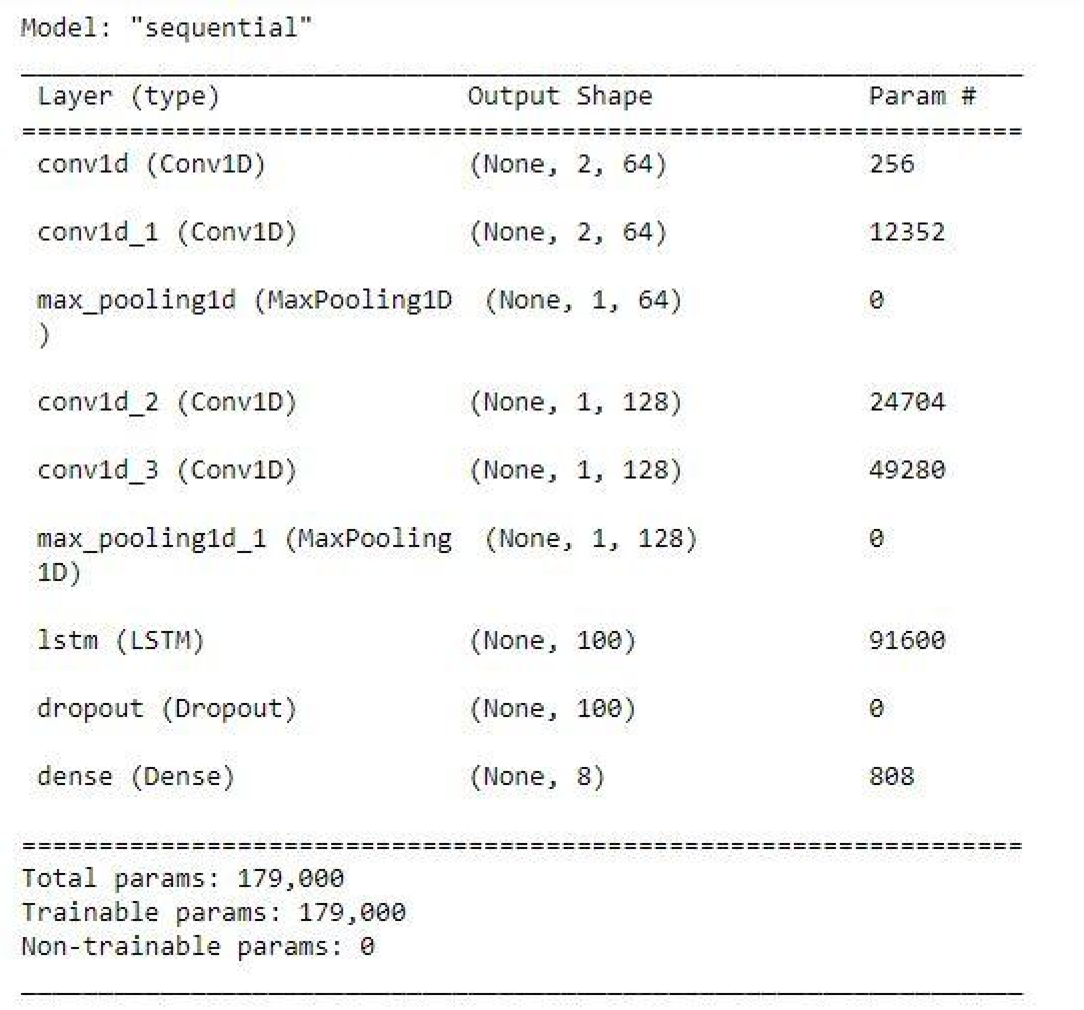
**Table 1: the GPS Tracker dataset traffic type distribution**

While the modeling phase, we choose the latitude and longitude as the predictors and the traffic type as the target. After the data preprocessing which consisted of standard scaling of the predictors, we used the Random Forrest, Decision Tree, and Naive Bayes algorithms and a 1D-CNN-LSTM model for the classification while trying Hyperparameter tuning.
The results presented in the following can possibly be improved with even more hyperparameter tuning. After data splitting, model training, and testing, the models had the following accuracy results in **Table 2**.

| **Model** | **train accuracy** | **test accuracy** |
|---|---|---|
| **Random Forrest (RF)** | 99.97 % | **89.65** % |
| **Decision Tree (DT)** | 99.97 % | 86.86 % |
| **Naive Bayes** | 60 % | 60.35 % |
| **1D-CNN-LSTM** | 90.45 % | 89.10 % |

**Table 2: Models training and testing accuracy (GPS DATASET)**

The 1D-CNN-LSTM (1) model summary is given in **Figure 1**.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)            (None, 2, 64)             256

 conv1d_1 (Conv1D)          (None, 2, 64)             12352

 max_pooling1d (MaxPooling1D  (None, 1, 64)           0
 )

 conv1d_2 (Conv1D)          (None, 1, 128)            24704

 conv1d_3 (Conv1D)          (None, 1, 128)            49280

 max_pooling1d_1 (MaxPooling  (None, 1, 128)          0
 1D)

 lstm (LSTM)                (None, 100)               91600

 dropout (Dropout)          (None, 100)               0

 dense (Dense)              (None, 8)                 808

=================================================================
Total params: 179,000
Trainable params: 179,000
Non-trainable params: 0
_________________________________________________________________
```

**Figure 1: the 1D-CNN-LSTM(1) proposed model summary**

After plotting the confusion matrixes, we got the results presented in **Table 3.**

| **RF / 1D-CNN LSTM (1)/DT** | **Recall** | **Precision** | **F1 score** |
|---|---|---|---|
| normal | **0.98** / 0.58 / 0.95 | **0.96** / 0.79 / 0.96 | **0.97** / 0.67 / 0.95 |
| ddos | 0.8 / **0.82** / 0.8 | **0.81** / 0.8 / 0.77 | 0.81 / **0.81** / 0.78 |
| injection | **0.84** / 0.76 / 0.8 | **0.87** / 0.85 / 0.83 | **0.85** / 0.8 / 0.81 |
| backdoor | 0.85 / **0.98** / 0.8 | 0.83 / **0.96** / 0.79 | 0.84 / **0.97** / 0.79 |
| password | 0.67 / **0.8** /0.66 | 0.71 / **0.85** /0.63 | 0.69 / **0.83** / 0.64 |
| ransomware | **0.65** / 0.61 /0.66 | 0.76 / **0.84** / 0.67 | 0.7 / **0.7** / 0.67 |
| xss | 0.33 / **0.91** / 0.35 | 0.52 / **0.98** /0.38 | 0.41 / **0.95** /0.36 |
| scanning | 0.89 / 0.26 / **0.92** | **0.99** / 0.55 / 0.98 | **0.94** / 0.35 / **0.95** |

**Table 3: Recall, Precision and F1 score of each class for different models**

The 2 models had very close similar test accuracies, the Random Forrest model had a bigger training accuracy compared to his testing accuracy with a 10.32 rate difference compared to only 1.35 rate difference for the 1D-CNN-LSTM model which can be interpreted as overfitting.
In opposition, the 2 models had very different recall, precision, and F1 scores for most of the classes and we can clearly observe that the Random Forrest model does much better predicting the normal traffic with an F1 score of 0.97 compared to just 0.67 for the 1D-CNN-LSTM model which is very low (In fact, this model has a recall of 0.58 for the normal class which means he classifies only around 58% of the normal traffic as normal and then wrongly classifies 42 % of the normal traffic as attacks which is a huge rate). Thus, the Random Forrest may be preferred.

### The Modbus dataset

It consists of a CSV file containing 51106 traffic recordings with 9 columns ( 3 time-related information columns) + "**FC1_Read_Input_Register" + "FC2_Read_Discrete_Value" + "FC3_Read_Holding_Register"+"FC4_Read_Coil"** and label and type where the label is either 0 if the traffic is normal or 1 if it's an attack and the type column, is a string which can be one of these strings: 'normal', 'backdoor', 'password', 'injection', 'xss' , 'scanning'. So using the dataset for classification can help us classify if the traffic is normal or an attack with 5 different attack possibilities.

The dataset distribution is 35000 normal traffic and 16106 attack recordings. The dataset distribution considering the different attack types can be seen in **table 4**.

| Traffic type | normal | password | backdoor | injection | xss | scanning |
|---|---|---|---|---|---|---|
| count | 35000 | 5000 | 5000 | 5000 | 577 | 529 |

**Table 4: the Modbus dataset traffic type distribution**

While the modeling phase we choose the“**FC1_Read_Input_Register”,“FC2_Read_Discrete_Value”, “FC3_Read_Holding_Register” and ”FC4_Read_Coil”** as the predictors and the traffic type as the target. After the data preprocessing which consisted of standard scaling of the predictors, we used the Random Forrest algorithm and a 1D-CNN-LSTM model for the classification while trying Hyperparameter tuning.
The results presented in the following can possibly be improved with even more hyperparameter tuning. After data splitting, model training, and testing, the Random Forrest and the 1D-CNN-LSTM had the following accuracy results in **Table 5**.

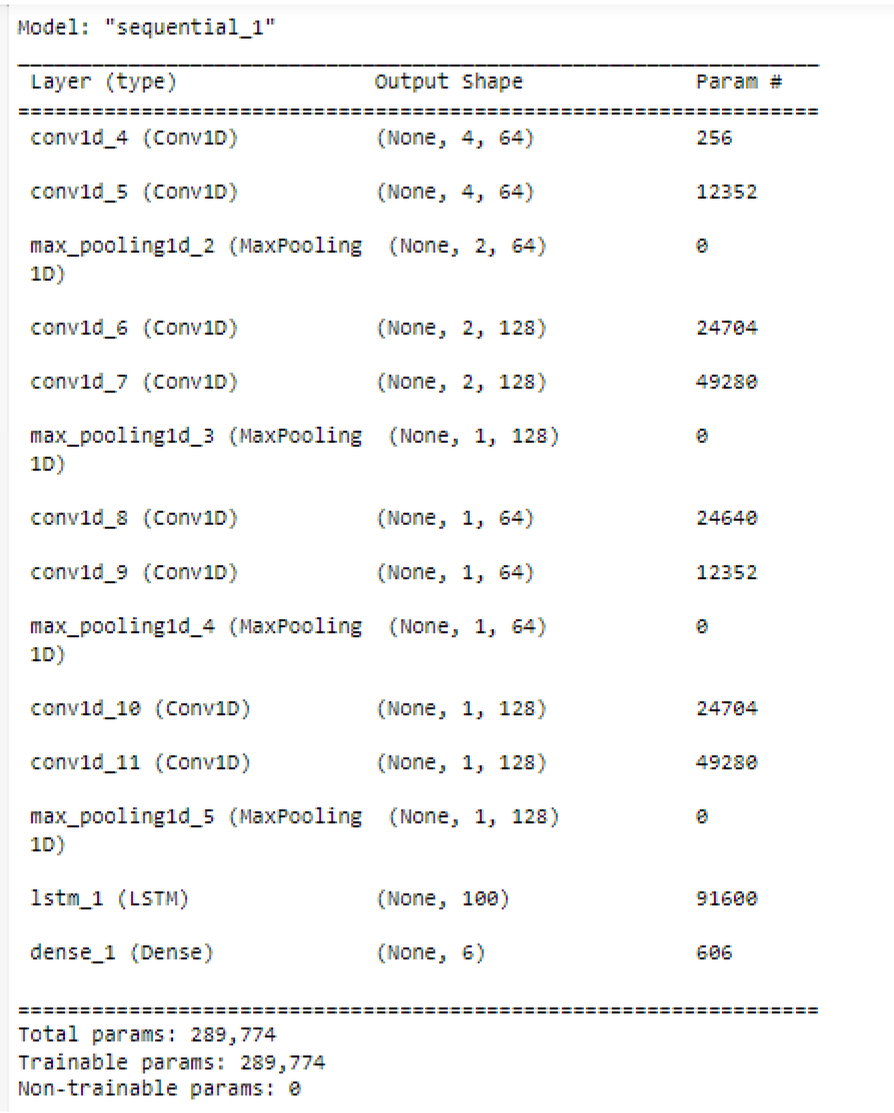
| Model | train accuracy | test accuracy |
|---|---|---|
| **Random Forrest (RF)**<br>**Decision Tree (DT)**<br>**Naive Bayes** | 99.97 %<br>99.97 %<br>68.4 % | 98 %<br>97.99 %<br>68.84 % |
| **1D-CNN-LSTM (2)** | 90.99 % | 88.84 % |

**Table 5 : Models training and testing accuracy (Modbus dataset)**

The 1D-CNN-LSTM (2) model summary is given in **Figure 2**.
After plotting the confusion matrixes we got the results presented in **Table 6.**

| **RF / 1D-CNN LSTM (2) / DT** | **Recall** | **Precision** | **F1 score** |
|---|---|---|---|
| backdoor | 0.95 / 0.66 / **0.96** | **0.9979** / 0.9 / 0.98 | **0.98** / 0.76 / 0.97 |
| injection | 0.98 / 0.74 / **0.99** | **0.99** / 0.94 / 0.97 | **0.99** / 0.83 / 0.98 |
| normal | **0.999** /0.96/ 0.996 | 0.97 / 0.91 / **0.98** | **0.99** / 0.93 / **0.99** |
| password | 0.91 / 0.65 / **0.93** | **1.0** / 0.94 / 0.97 | **0.96** / 0.76 / 0.95 |
| scanning | 0.58 / 0.29 / **0.61** | **1.0** / 0.81 / 0.92 | **0.73** / 0.42 / **0.73** |
| xss | **0.9** / 0.36 / **0.9** | **1.0** / 0.86 / 0.96 | **0.95** / 0.51 / 0.93 |

**Table 6 : Recall, Precision and F1 score of each class for both models (Modbus dataset)**



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_4 (Conv1D)           (None, 4, 64)             256

 conv1d_5 (Conv1D)           (None, 4, 64)             12352

 max_pooling1d_2 (MaxPooling  (None, 2, 64)            0
 1D)

 conv1d_6 (Conv1D)           (None, 2, 128)            24704

 conv1d_7 (Conv1D)           (None, 2, 128)            49280

 max_pooling1d_3 (MaxPooling  (None, 1, 128)           0
 1D)

 conv1d_8 (Conv1D)           (None, 1, 64)             24640

 conv1d_9 (Conv1D)           (None, 1, 64)             12352

 max_pooling1d_4 (MaxPooling  (None, 1, 64)            0
 1D)

 conv1d_10 (Conv1D)          (None, 1, 128)            24704

 conv1d_11 (Conv1D)          (None, 1, 128)            49280

 max_pooling1d_5 (MaxPooling  (None, 1, 128)           0
 1D)

 lstm_1 (LSTM)               (None, 100)               91600

 dense_1 (Dense)             (None, 6)                 606

=================================================================
Total params: 289,774
Trainable params: 289,774
Non-trainable params: 0
```

**Figure 2: the 1D-CNN-LSTM(2) proposed model summary**

The Random Forrest model has done very well classifying different traffic types since it had a very high testing accuracy of 98 %. Moreover, the model had a very good recall and precision scores ( >= 0.9 except for the “scanning recall”). The Random Forrest model did so much better than the 1D-CNN-LSTM model that had lower recall scores ranging from 0.29 to 0.74 except for the normal class i.e the model is not able to predict almost all the attacks effectively but its attack detection was precise since the precision scores range from 0.81 to 0.94 which is quite acceptable.

**Conclusion**

With the growing number of IoT devices and the IoT attacks because of the weak IoT network protocols and also with the failure of the old security techniques like cryptography to identify threats, the need to implement intelligent Intrusion Detection Systems is more urgent than ever.

Different Machine Learning and Deep Learning models with the help of the available datasets assisted the development of different IDSs that are highly accurate as plenty of models had a very high accuracy exceeding 95 % either in the previous studies or the models proposed above.

**References**

- A survey on Deep Learning based Intrusion Detection Systems on Internet of Things | IEEE Conference Publication | IEEE Xplore
- Intrusion Detection System on IoT with 5G Network Using Deep Learning (hindawi.com)
- A machine learning-based intrusion detection for detecting internet of things network attacks - ScienceDirect
- (PDF) DL-IDS: a deep learning–based intrusion detection framework for securing IoT (researchgate.net)
- The TON_IoT Datasets | UNSW Research