Intelligent Systems with Applications 16 (2022) 200152

Contents lists available at ScienceDirect

Intelligent Systems with Applications

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

Multi-tiered Artificial Neural Networks model for intrusion detection in smart homes

Shaleeza Sohail [a,b], Zongwen Fan [c,*], Xin Gu [d], Fariza Sabrina [e]

[a] School of Information Technology, King's Own Institute, Sydney, NSW 2000, Australia
[b] College of Engineering, Science and Environment, The University of Newcastle, Sydney, NSW 2000, Australia
[c] College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China
[d] College of Engineering, Science and Environment, The University of Newcastle, Callaghan, NSW 2308, Australia
[e] School of Engineering and Technology, Central Queensland University, Sydney, NSW 2000, Australia

ARTICLE INFO

Keywords:
Cybersecurity
Artificial Neural Networks
Subcategory attack classification
Hyperparameter selection
Explainability

ABSTRACT

In recent years cybersecurity has become a major concern in the adaptation of smart applications. A secure and trusted mechanism can provide peace of mind for users, especially in smart homes where a large number of IoT devices are used. Artificial Neural Networks (ANN) provide promising results for detecting any security attacks on smart applications. However, due to the complex nature of the model used for this technique, it is not easy for common users to trust ANN-based security solutions. Also, the selection of the right hyperparameters for ANN architecture plays a crucial role in the accurate detection of security attacks. This paper proposes Multi-tiered ANN Model for Intrusion Detection (MAMID) that is a novel and scalable solution for optimal hyperparameter selection to detect security attacks with high accuracy. The explainability analysis of the predictions made by the model is also provided for establishing trust among users. The approach considers a subset of the dataset for quick, scalable and optimal selection of hyperparameters to reduce the overhead of the process of ANN architecture design. Using a very recent IoT dataset the proposed approach showed high performance for intrusion detection with 99.9%, 99.7%, and 97.7% accuracy for binary, category, and subcategory classification of attacks. To the best of the authors' knowledge, no previous research work has been able to achieve attack detection at the subcategory level beyond 90%.

1. Introduction

The Internet of Things (IoT) has been recognised as a disruptive technology with a huge impact on our everyday lives. Some widely used applications are smart home, smart health, smart city, smart grid, smart agriculture, smart cars, and logistic tracking to name a few (Lee, 2020). IoT devices are connected via the Internet to collect and exchange data with each other for a particular purpose (Yang et al., 2017). Due to a huge demand for IoT-based applications, IoT devices are continuously evolving in terms of low cost, easy handling, small form factor and sensor precision. However, security and privacy have not been given priority while designing these devices (Thamilarasu & Chawla, 2019). Therefore, a huge amount of research has focused on securing IoT-based applications; however, a lot more needs to be done to completely protect IoT networks from intruders.

Smart home applications have played an important role in the massive growth of IoT usage. A smart home allows individuals to fully connect their home appliances/devices with each other and enables the owner to control the devices remotely, as shown in Fig. 1, which can provide unprecedented control and comfort. But this benefit comes with a huge security risk if adequate security measures are not considered. In a smart home environment, one of the biggest challenges is the users' inability to understand, appreciate and take security precautions. Most IoT devices in smart homes are designed with an extended set of sensors and actuators. For example, a smart light may come with a microphone that may leave the IoT system vulnerable to security and privacy intrusions. A common household resident may not have the knowledge and skills to interpret and safeguard the IoT system (Dolan et al., 2020). Hence, there is a huge need for an automated mechanism to protect smart homes from security and privacy intrusions.

* Corresponding author. All authors contributed equally to this work.
E-mail addresses: Shaleeza@koi.edu.au (S. Sohail), zongwen.fan@hqu.edu.cn (Z. Fan), xin.gu@newcastle.edu.au (X. Gu), f.sabrina@cqu.edu.au (F. Sabrina).

https://doi.org/10.1016/j.iswa.2022.200152
Received 23 August 2022; Received in revised form 13 October 2022; Accepted 8 November 2022
Available online 11 November 2022

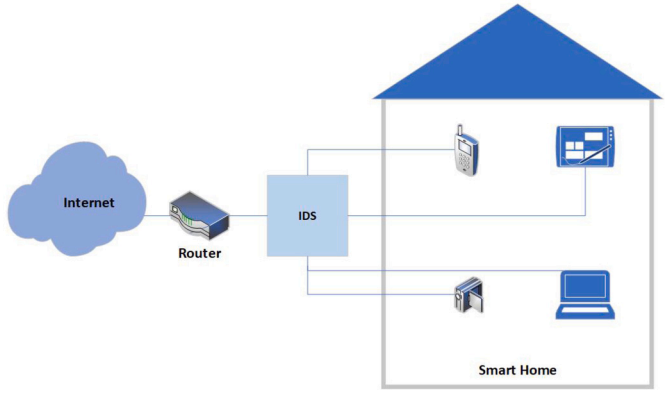


Fig. 1. Smart home scenario.

In recent years, many research efforts have been focused on automated intrusion detection systems using Machine Learning (ML). The essence of a ML system is to train the activity model based on the signature, pattern or characteristics of attacks which will be compared against the new traffic behaviour. Several supervised and unsupervised machine learning algorithms could be used for this purpose, such as random forest, decision tree, support vector machine, long short-term memory (a type of neuron network), and Artificial Neural Networks (ANN). ANN is one of the machine learning algorithms that has been widely used for classification tasks across many domains due to its black box nature of adapting to the underlying system (Zhang, 2000). This property makes ANN highly suitable for network traffic intrusion detection when a high dimensional data set is involved due to its robust ability of model building when the pattern is buried (Dreiseitl & Ohno-Machado, 2002). A very important aspect of ANN's high performance lies in the optimal selection of its hyperparameters for that dataset (Choraś & Pawlicki, 2021). Considerable effort is required for finding the optimal setup, which is a non-trivial task as it not only affects performance but can be a very time-consuming exercise due to the availability of a large number of options for each parameter involved in the setup.

MAMID, a novel scalable mechanism for hyperparameter selection for ANN model is proposed in this work that provides significantly better results when compared to other related work. As already discussed, for an ANN model the selection of hyperparameters plays an important and significant role when it comes to prediction accuracy. A simple and scalable approach to detect security attacks is proposed based on detection accuracy, computational overhead and time constraint. The accuracy of prediction is important as missing any attack in case of the wrong detection can be very damaging. Computation overhead plays an important role as finding the most optimal solution using extensive computational resources may not be feasible and affordable. Time is one of the constraints as such detection has to be real-time and any model that takes a long time to train or detect is not suitable for intrusion detection in this scenario. The selection of the hyperparameters has been evaluated to analyse the efficiency of the proposed model. It is important to note that the proposed approach has been evaluated on a new IoT dataset, IoTID20 (Ullah & Mahmoud, 2020) as the dataset shows recent network traffic characteristics (collected in 2020). The explainability analysis of the model shows feature importance for the predictions which can be useful for making users trust the model output.

The main contributions of this paper are:

- An optimal hyperparameter selection method for designing ANN with low overhead has been provided.
- The proposed approach shows high performance for security attack detection at the subcategory level for a recently captured IoT dataset which shows the applicability of the approach for the detection of new attacks.
- The explainability of the ANN model has been analysed which can be helpful for user understanding and trust building.

The paper is organised as follows: section 2 discusses the related literature in ANN-based intrusion detection systems and also the research carried out using IOTID20 dataset. Section 3 provides complete details of the proposed multi-tiered ANN model for intrusion detection. The hyperparameter selection is given in section 4. Section 5 provides the categorisation results and analysis for intrusion detection purposes. The conclusion has been made, and future work options have been identified in section 6.

2. Related work

There has been a huge amount of research actively going on in the area of cybersecurity where machine learning algorithms are used for anomaly and attack detection. A brief review of some of the literature that is relevant to this work has been provided in the following subsections.

2.1. Machine learning based intrusion detection

As mentioned above, machine learning based anomaly and intrusion detection is an active research area; some of the commonly used algorithms are Random Forest (RF), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Multi-Layered Perceptions (MLP), Artificial neural network (ANN) etc. (Islam et al., 2020).

The use of an ensemble classifier can improve attack detection accuracy as two different algorithms can resolve the issues in two different manners. Random Forest and Average One-Dependence Estimator are used for resolving attribute dependency and for improving the classification accuracy of network traffic into anomaly or malicious classes (Jabbar et al., 2017). A cluster-based ensemble classifier consisting of ADTree and KNN shows the high accuracy of attack detection for intrusion detection systems (Jabbar, Aluvalu, Reddy, 2017). An ensemble approach consisting of the Bagging method combined with REPTree results in high classification accuracy while the Bagging method takes much less time to build the model (Gaikwad & Thool, 2015). A bio-inspired genetic security approach shows quick detection of misbehaving devices in smart cities during mobile cloud sourcing operations when IoT devices are employed (Kantipudi et al., 2022). A complete smart home monitoring system that detects and prevents physical water system failures, effects of extreme weather conditions, climate condition issues and suspicious security events uses sensors and fuzzy logic for next-generation IoT active monitoring (Woźniak et al., 2020). The system uses physical sensor data and user information to control the home environment for high comfort and security.

Among different machine learning algorithms, ANN has been considered one of the most efficient algorithms for anomaly or intrusion detection (Tsaih et al., 2018) (Choraś & Pawlicki, 2021). Taher et al. (2019) compared the performance of SVM and ANN for detecting network intrusion and found that ANN performs better than SVM. The authors used NSL-KDD dataset in this work.

Choraś and Pawlicki (2021) proposed an ANN based system for intrusion detection for multi-class classification. In this work, the authors tried various possible options of different hyperparameters to come up with the most efficient system. The authors argue that the selection of hyperparameters plays a very important role in classification results and a little change in hyperparameter setup has a major impact on accuracy. The hyperparameters that were fine-tuned in this work were the activation function, the optimiser, the number of epochs, the number of neurons and batch size. The performance of the proposed model was evaluated using two popular datasets (NSL-KDD and CICIDS2017).

Experimental results show that the most effective model achieves an accuracy of 99.909% for multi-class attack detection.

Hodo et al. (2016) presented an ANN-based network intrusion detection system for IoT networks. This work focused on binary classification - detecting normal and attack traffic (DDoS/DoS). The performance of the proposed system was tested in a simulated IoT network and the result shows 99.4% accuracy in detecting Distributed Denial of Service (DDoS) and Denial of Service (DoS) attacks.

Subba et al. (2016) proposed an ANN-based intrusion detection system that could be easily deployed for real-time intrusion detection. The authors argued that some of the existing machine learning based systems might achieve high performance in intrusion detection but suffer from high complexity and computational overhead. So, in this work, the authors proposed a three-layer ANN model (with one hidden layer) to minimise the computational overhead for training and execution of the system. The performance of the model has been analysed using the NSL-KDD dataset. The results from the performance evaluation show that this proposed system performs better than the Naive Bayes based model and is similar as SVM and C4.5 based IDS models.

From the above discussion, it is apparent that ANN is a good choice for intrusion detection purposes but faces the challenge of being complex for the optimal selection of hyperparameters.

2.2. IoT dataset

When machine learning algorithms are used for attack detection, then datasets become very crucial as training and testing of these algorithms cannot be carried out without a dataset. Recently, several datasets have been used by researchers for this purpose, like UNSW-NB15 (Moustafa & Slay, 2015) and Bot-IoT (Koroniotis et al., 2019). The IoTID20 dataset has been chosen for training and testing the proposed ANN model as this is one of the newest datasets collected in the IoT environment and it may depict more realistic and up-to-date network traffic characteristics. Another important aspect to mention is that as this dataset was collected and made available for use very recently, there are a handful of research utilising this dataset for experimentation. Moreover, none of the existing research (until June 2021) was able to achieve a subcategory classification accuracy of more than 90% for this dataset. This dataset has been considered to analyse the IoT intrusion detection model as there is a scope for improvement.

Ullah and Mahmoud (2020) collected and experimented with this dataset while considering several machine learning approaches. Anomalous activities are detected at binary, category and subcategory levels. Many machine learning algorithms (SVM, Gaussian NB, LDA, Logistic Regression, Decision Tree, and Random Forest) and an ensemble method are used for IoT attack detection with varying results. For binary and category classification Decision Tree, Random Forest and Ensemble methods produced very good results with F-Score of more than 95% in most cases. However, when it comes to subcategory classification, none of the algorithms was able to get F-Score and Accuracy of more than 88%. In security scenarios, getting high accuracy for attack detection is highly recommended and the intrusion detection system with 88% accuracy may not be relied upon completely. Hence, it can be deduced that there is room for improvement to the machine learning model in order to achieve high accuracy and precision for subcategory based attack detection using this dataset.

The importance of feature selection has been analysed for anomaly-based intrusion detection approaches and the IoTID20 dataset is used for testing the hybrid feature selection approach using the Random Forest algorithm (Maniriho et al., 2020). The authors only discussed partial results for three attack categories indicating more than 99.9% accuracy. The accuracy for detection of some attack categories is very high which shows the strength of the hybrid feature selection approach that only uses a subset of features that are common among the features selected using Information Gain and Gain Ration methods. However, the applicability of such a feature selection approach cannot be justified unless complete results are provided and analysed for all category and subcategory-based attacks.

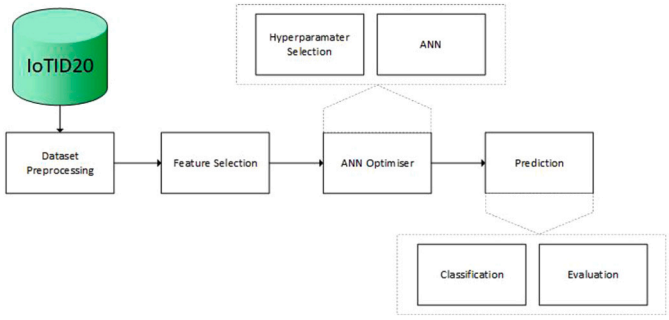


Fig. 2. Proposed multi-tiered ANN model for intrusion detection (MAMID).

Qaddoura et al. (2021) have proposed a multistage approach for anomaly detection for imbalanced datasets like IoTID20. The authors used approaches like oversampling and K-means++ clustering in order to mitigate the impact of imbalance among the examples for different categories. The results showed good accuracy for binary classification only. The classification of categories and subcategories was not undertaken by this research work and hence, the effectiveness of this approach for multi-class classification in the IoTID20 dataset cannot be assessed. Similarly, Anjum (Farah, 2020) has discussed cross dataset evaluation considering IoTID20 and Bot-IoT datasets. However, all the attack detection was only for binary or category level and subcategory level classification was not considered by the author.

3. Multi-tiered ANN model for intrusion detection (MAMID)

In this section, the proposed model is discussed that uses ANN for intrusion detection by selecting the most suitable hyperparameters for high prediction accuracy, low computational overhead and real-time detection.

3.1. Methodology

The proposed model is shown in Fig. 2. The proposed system consists of four parts: Dataset Preprocessing, Feature Selection, ANN Optimiser, and Prediction. At the start, the dataset is preprocessed to eliminate any redundant data and missing values. After that, feature selection is performed in order to find the minimum set of features that can be effectively used for classification (discussed later in detail). Then the data is passed to the ANN optimiser that finds the optimal neural network topology and hyperparameters for configuring the neural network for training and testing the model. At the final stage, the predictions are made for real-time detection of malicious traffic. Importantly, not only the prediction accuracy but also the computational overhead and time complexity is considered to apply this approach to the smart home scenario that requires real-time detection and may not have access to high-end servers. The following subsections discuss the main components of the proposed system in detail.

The purpose of the three types of classification is that the binary classification detects attacks and normal traffic. Category-based classification detects the category of the attack as well. The third and most challenging classification is of subcategories of the attacks. The category and subcategory of attacks need to be detected as the response for different attacks needs to be different based on their type. Such information can provide a very useful starting point for Cybersecurity professionals when quick recovery of the system needs to be performed after the attack.

For the binary classification of intrusion detection, the binary cross-entropy is used as the loss function and the number of neurons in the last layer of ANN is set to one with the Sigmoid function used. In this case, if the value of the function output is larger than 0.5, its label is

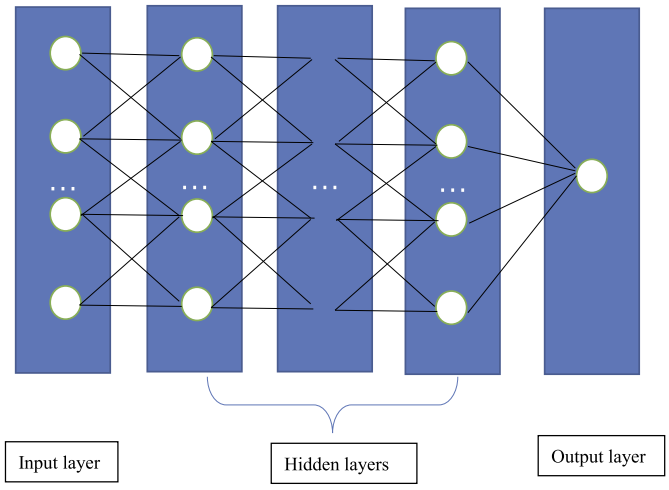


Fig. 3. A structure of a classic artificial neural network, including one input layer, multiple hidden layers, and one output layer.

set to 1, otherwise set to 0. While for multiclassification, the categorical cross-entropy is used as the loss function and the number of neurons in the last layer is the same as the number of classes where one-hot category encoding is utilised for the labels. By doing so, even if a sample is similar between classes, the optimal label will be selected.

3.2. The ANN model

The ANN model consists of three types of layers (Mehlig, 2019), input layer, hidden layer and output layer, as shown in Fig. 3. In each layer, there are many nodes called neurons. The input layer contains the input data feeding into the model. The number of neurons is the same as the number of input features. Each neuron from the input layer interconnects with one or more neurons in the second layer with a weight. The output of the second layer will go to the next layer until the output layer.

Given a three-layer ANN, the weights and a bias term between the input layer and the hidden layer are $\boldsymbol{W}^1(x_1, x_2, x_3)$ and b^1, respectively. The inner product between the weight matrix and the input data is transformed by an activation function $g(x)$ (e.g. the Sigmoid activation function in Eq. (2)). The mapped values are fed into the next layer until the output layer. For binary classification, there is only one neuron in the output layer. The feed-forwarded process can be formulated as given in Eq. (1).

$$f(X) = \boldsymbol{W}^2 g(\boldsymbol{W}^1 \boldsymbol{X} + b^1) + b^2, \tag{1}$$

$$g(x) = 1/(1 + exp(-x)), \tag{2}$$

where $g(\cdot)$ is a Sigmoid activation function, it can also be Softmax, Tanh, ReLU activation functions etc, $\boldsymbol{W}^1$ and $\boldsymbol{W}^2$ are weight matrices for the input and hidden layers, respectively, and b^1 and b^2 are bias terms added to the hidden and output layers, respectively.

The ANN model consists of one hidden layer for the experiments. The hyperparameters that are tuned to get the best results are: the number of neurons, the number of epochs, batch size, the optimiser function, the activation function for input and hidden layers, and the activation function for the output layer.

3.3. ANN optimiser

The optimisation of ANN setup is the most important aspect when it comes to the accuracy of the prediction and the computational overhead involved in the process (Choraś & Pawlicki, 2021). With the correct selection of topology and hyperparameters, the model can achieve the highest accuracy of the ANN classifiers. Fig. 4 shows hyperparameter

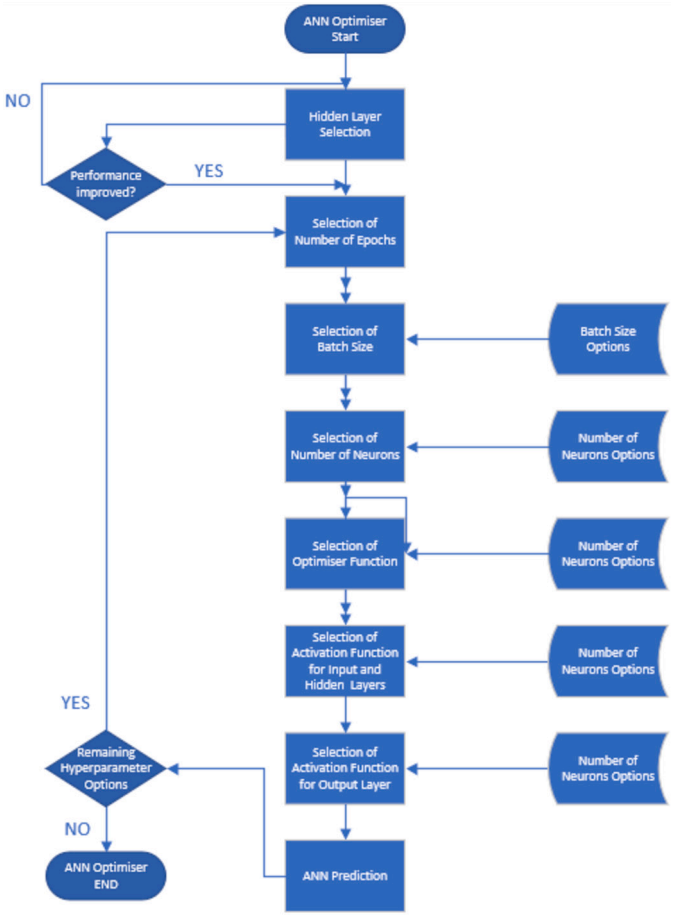


Fig. 4. ANN Optimiser.

selection process by ANN optimiser. A general description of the hyperparameters is given below. The actual options for each hyperparameter that are experimented with are given in Section 4.

Optimisation methods are crucial when it comes to the effective training of neural networks. Optimisation methods are used to reduce losses and hence, improve performance by changing weights and the learning rate of the neural network. Finding the optimisation method suitable for the dataset can play a vital role in prediction accuracy. Some commonly used optimisers are gradient descent, stochastic gradient descent, momentum, nesterov accelerated gradient and adaptive moment estimation (ADAM). Every optimiser has its strengths and weaknesses; for example, gradient descent is computationally inexpensive but requires large memory. On the other hand, ADAM converges rapidly but is computationally expensive (Doshi, 2019).

Epoch is when complete training data has passed through the neural network and updated the internal model parameters. Most of the time all samples of the training data cannot pass in one go and hence, each epoch is divided into several batches. A single epoch may result in underfitting and multiple epochs are required as the internal parameters need to be changed multiple times for better generalisation. Once the number of epochs starts increasing the model starts fitting better until it reaches optimal fitting and if the number of epochs are still increased then the overfitting can happen (Afaq & Rao, 2020).

Batch size defines the number of samples that the ANN algorithm must walk through before updating internal parameters. The batch size can affect the performance of the ANN algorithm in terms of prediction accuracy and training time. In general, a smaller batch size is better in generalising to the test set than large batch training (Smith et al., 2017).

The activation function lets the neural network learn complex patterns in the data by adding non-linearity into the model. Various

non-linear activation functions are available such as Sigmoid, Softmax, Tanh, ReLU etc. Choice of activation function can impact the performance of the training process (Hayou et al., 2019). The activation function for the input layer, hidden layer and output layer may be different based on the dataset.

Based on the domain knowledge, resource restrictions and dataset characteristics, a number of hyperparameter options are chosen. ANN optimiser tests the prediction accuracy of the neural network using these optimisers with a small subset of data to find the most suitable set of options.

3.4. Explainability of the proposed model

Advanced predictive models like ANN achieve good performance by using complex models which are difficult to explain. However, assessing how a particular prediction is made is vital for understanding and trusting the model by the end user. The explainability of the proposed model has been assessed using SHapley Additive exPlanations (SHAP) by identifying feature relevance on the basis of each sample (Lundberg & Lee, 2017). SHAP explains what the model is doing by finding the important relationships between the input features and the predicted outcome, which allows the users to interpret the predictions of the ANN model. SHAP uses an explainer method to combine the inputs to evaluate the effects on the predictive model and produces a locally interpretable model for every prediction (Lundberg & Lee, 2017). The explainer illustrates the SHAP value for each feature that illustrates the importance of each feature for a particular prediction.

After training the ANN model with a training subset of the dataset, the SHAP explainer analyses the testing subset. The explainer generates the output of the ANN model and assigns a SHAP value to each feature for prediction. These SHAP values indicate which feature increases or decreases the influence on the prediction to belong to a particular class (Huang et al., 2020). The summary plot is used to visualise the effects of the indicated features on the prediction of the attack subcategory. The force plot has been used to show how features affect the prediction for a single value.

4. Hyperparameter selection by ANN optimiser

In this section the hyperparameter selection by ANN optimiser for IoTID20 dataset for attack detection at binary, category and subcategory level has been discussed.

4.1. ANN setup

The optimal ANN setup is found by using a grid search, which completes an all-encompassing search over the hyperparameter's space. The grid search parameters included:

- Number of epochs
- Batch size
- Number of neurons
- optimiser function
- Activation function for input and hidden layers
- Activation function for output layer

It is worth mentioning that the experiments were conducted with one, two, or three hidden layers. The results were not significantly different. Therefore, in this study, only one hidden layer was chosen for hyperparameter selections.

The full cycle of learning and adapting the weights of the network is called epoch. In this model, 100 epochs and 200 epochs were used in the grid search. The particular count of samples utilised in one iteration is called batch size. 10 and 100 batches were used in this grid search. The units or nodes in the input and hidden layers are called neurons. In this model, 100 neurons and 200 neurons were used in the grid search.

The optimiser functions evaluated in this model are:

- Adaptive Moment Estimation (Adam) - a stochastic optimisation method that uses a randomly selected data subset to create a stochastic approximation, instead of using the entire data set to calculate the actual gradient.
- Root Mean Square Propagation (rmsprop) - an extension of gradient descent and the AdaGrad version of gradient descent that uses a decaying average of partial gradients in the adaptation of the step size for each parameter.
- Stochastic gradient descent (SGD) - an iterative method for optimising an objective function with suitable smoothness properties. It can be regarded as a stochastic approximation of gradient descent optimisation since it replaces the actual gradient with an estimate thereof.
- AdaMax (AdaMax) - an optimisation method that implements the AdaMax algorithm. It is a variant of Adam based on the infinity norm.
- AdaGrad (Adagrad) - a stochastic optimisation method that adapts the learning rate to the parameters. It performs smaller updates for parameters associated with frequently occurring features, and larger updates for parameters associated with infrequently occurring features.

In the input and hidden layer, and in the output layer, the activation functions were evaluated independently. The Activation functions evaluated in this model are:

- Rectified Linear Unit (ReLU) - the range of the activation value is from 0 to infinity.
- Hyperbolic Tangent (tanh) - the range of the activation value is from -1 to 1.
- Sigmoid (sigmoid) - the range of the activation value is from 0 to 1.
- Softplus (softplus) - it is the softer (smoother) version of ReLU and softplus. The range of the activation value is from 0 to infinity.
- Softmax (softmax) - it is used for multiclass classification by converting a vector of values to a probability distribution. The elements of the output vector are in range (0, 1) and sum to 1.

The evaluations were conducted for three types of classifications:

- Binary classification
- Category classification
- Subcategory classification

For binary classification, the dataset has two classes: Anomaly or Normal as indicated in the data field called Label. For category classification the dataset has six classes: Miari, DoS, Scan, MITM, ARP Spoofing, or Normal indicated in the data field called Category. For subcategory classification the dataset has ten classes: DoS-Synflooding, Mirai-Ackflooding, Mirai-Hostbruteforceg, Mirai-HTTP Flooding, Mirai-UDP Flooding, MITM ARP Spoofing, Scan Hostport, Scan Port OS, or Normal indicated in the data field called Subcategory.

For each classification model, it required 1,000 experiments to evaluate all the pre-defined hyperparameter options. That is Epochs (2) × Batch (2) × Neurons (2) × optimiser (5) × Activation I (5) × Activation II (5) = $2 \times 2 \times 2 \times 5 \times 5 \times 5 = 1000$ experiments. The aim of the study is to show that the results of the subset dataset can be effectively used to classify the full dataset. Therefore, this study required 6,000 experiments including subset and full dataset experiments. The first 3,000 experiments were conducted over a subset of the full dataset to find the optimal hyperparameter selection, and the second 3,000 experiments were conducted to validate the optimal selection with the full dataset of 625,783 records.

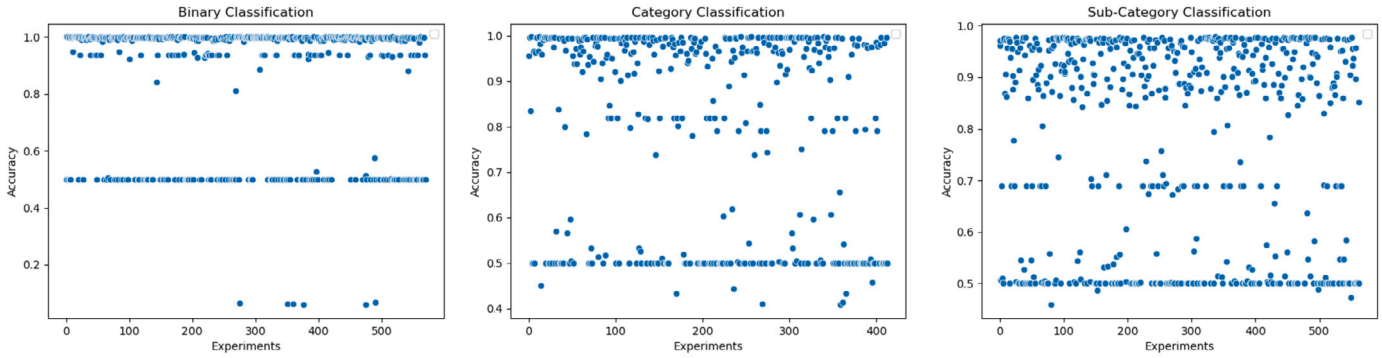


Fig. 5. Accuracy Results from (a) binary (b) category and (c) subcategory classification from Subset dataset experiments.

To better illustrate the procedure of hyperparameter selection, its pseudo-code is given in **Algorithm 1**. Based on the grid search, the optimal hyperparameters are chosen after iterating all the values at hand for model training and evaluated on the validation set.

Algorithm 1 The pseudo-code for selecting the optimal hyperparameter of ANN.

```
Given a training set.
Set the random state as 0.
Set the best accuracy as 0.
Randomly set the optimal hyperparameters.
Split the training set into a training set and a validation set.
for The number of neurons ∈ [100, 200] do
    for The number of epochs ∈ [100, 200] do
        for Batch size ∈ [10, 100] do
            for Optimizer ∈ ['adam', 'rmsprop', 'SGD', 'adamax', 'adagrad'] do
                for Activation ∈ ['relu', 'tanh', 'sigmoid', 'softplus', 'softmax'] do
                    for Activation output ∈ ['relu', 'tanh', 'sigmoid', 'softplus', 'softmax'] do
                        Train the ANN with the selected hyperparameters based on the training set.
                        Evaluate the ANN based on the validation set.
                        if Current accuracy is higher than the best accuracy? then
                            Update the best accuracy as the current accuracy.
                            Update the optimal hyperparameters as the current hyperparameters.
                        else
                            Continue;
                        end if
                    end for
                end for
            end for
        end for
    end for
end for
Output the optimal hyperparameters.
Train the ANN with the best hyperparameters based on the training set.
Evaluate the ANN based on the testing set.
```

4.2. Optimal hyperparameter selection using subset

Because of the mismatched data structure between the data and optimiser activation functions, some experiments were not successful. The analysis is based on the successful experiments over the subset with 10,000 records. 464 out of 1000 binary experiments, 474 out of 1000 category experiments, and 437 out of 1000 subcategory experiments returned successful results. The results over three different classifications were distributed on scatter plots in Fig. 5. By observing this figure, it is easy to see that, with the increasing number of target types, the accuracy rates become lower.

To understand the distribution of the high accuracy rates over 90%, the summary of the results using the predefined hyperparameters for binary classification, category classification, and subcategory classification are listed in Table 1. As can be seen from the table, it is much more challenging to reach 99% with the number of prediction targets increases in the experiments. 57% of binary experiments, 0.5% of category experiments, and 0% subcategory experiments reached over 99% accuracy rates.

Table 1
Top 10 Best hyperparameter setups for binary classification using the full dataset.

Classification	Accuracy	Results	Percentage
Binary (successful experiments = 464)	99+%	263	57%
	95+%	372	80%
	90+%	434	93%
Category (successful experiments = 447)	99+%	2	0.50%
	95+%	262	59%
	90+%	335	75%
Subcategory (successful experiments = 437)	99+%	0	0%
	95+%	45	10%
	90+%	227	52%

To select the optimal hyperparameters, two approaches were taken in this study. The first one is to investigate the hyperparameter settings from the experiments with the highest accuracy results (Approach I). The second approach is to investigate each hyperparameter setting together with its accuracy levels from all experiments (Approach II).

Therefore, Approach I is to investigate the top 10 results of three classifications (Table 2).

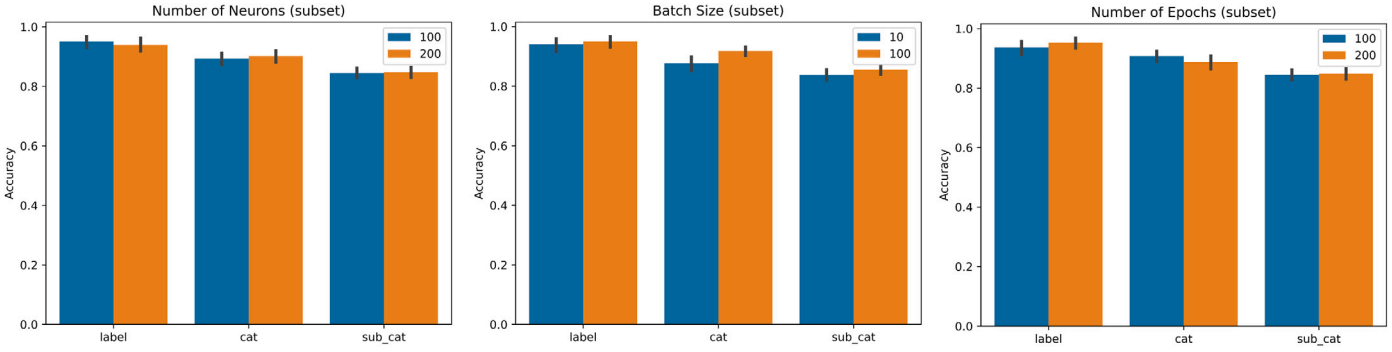
Table 2 shows the range of top 10 accuracy for binary is between 99.82-99.88%, Category is between 98.89-99.04%, and Subcategory is between 95.49-95.65%. It is obvious to see the optimal options for Optimiser, Activation I, and Activation II. The optimal optimiser for top 10 classification results is **adam**, which ran 21 (7 + 6+ 6) out of 30 experiments. The optimal Activation I function for top 10 classification results is **tanh**, which conducted 24 (10+ 7+ 7) out of 30 experiments. The optimal Activation II function for top 10 classification results is **softmax**, which was used in 22 (2+ 10+ 10) out of 30 experiments. However, it is difficult to identify the optimal options for the number of neurons, Batch, and Epochs. For each hyperparameter, a similar number of either option was used for the top 10 classifications.

Approach II is to investigate how well the hyperparameters were used in all experiments in terms of accuracy. Six bar charts were generated to show the accuracy levels of each hyperparameter option for three classification types in all successful experiments.

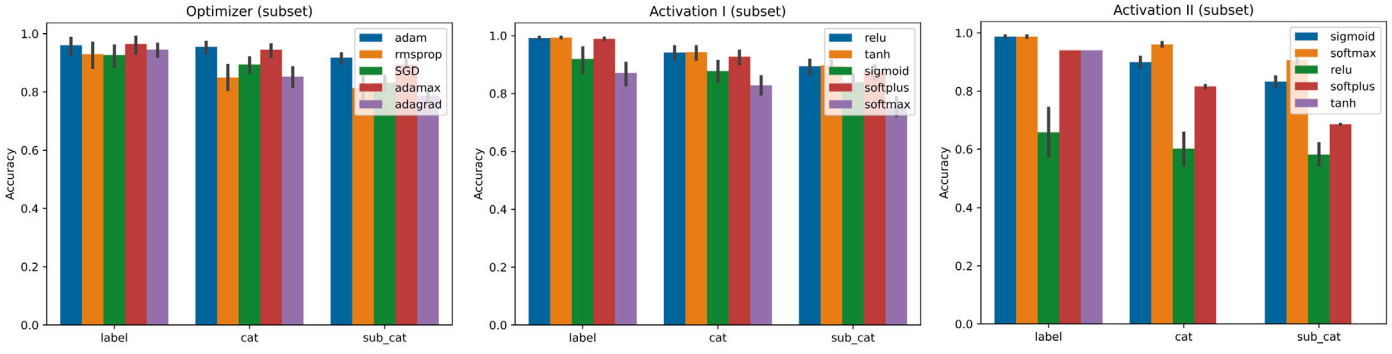
The bar charts of epochs, batch and neurons show not much difference in either hyperparameter selection as shown in Fig. 6. The results from this approach are similar to Approach I. Combining the results from both approaches, the optimal selections were made. 200 was selected as the optimal number of neurons, 100 was selected as the batch size, and 200 was selected as the optimal number of epochs.

Table 2
Top 10 Best hyperparameter setups for binary classification using full dataset.

Output	Accuracy	Neurons	Batch	Epoch	Optimiser	Activation I	Activation II
Binary	**99.88%**	**100**	**10**	**200**	**Adam**	**tanh**	**sigmoid**
	99.86%	100	100	200	rmsprop	tanh	sigmoid
	99.85%	200	100	200	Adam	tanh	sigmoid
	99.84%	200	10	100	Adam	tanh	sigmoid
	99.84%	100	10	100	Adam	tanh	sigmoid
	99.84%	200	100	100	Adam	tanh	sigmoid
	99.83%	100	100	200	Adam	tanh	sigmoid
	99.83%	200	100	200	rmsprop	tanh	softmax
	99.82%	200	100	100	Adam	tanh	softmax
	99.82%	200	100	200	rmsprop	tanh	sigmoid
Category	**99.04%**	**200**	**10**	**200**	**Adamax**	**tanh**	**softmax**
	99.03%	100	100	200	Adam	tanh	softmax
	99.00%	200	10	100	Adam	sigmoid	softmax
	98.98%	100	10	200	Adam	sigmoid	softmax
	98.96%	100	10	200	AdaMax	tanh	softmax
	98.93%	200	100	200	Adam	tanh	softmax
	98.93%	200	100	100	Adam	tanh	softmax
	98.92%	200	10	100	Adam	tanh	softmax
	98.92%	200	10	100	AdaMax	tanh	softmax
	98.89%	200	10	200	AdaMax	ReLU	softmax
Subcategory	**95.65%**	**100**	**10**	**200**	**AdaMax**	**tanh**	**softmax**
	95.63%	200	10	100	Adam	sigmoid	softmax
	95.62%	100	10	200	Adam	sigmoid	softmax
	95.61%	100	100	200	Adam	tanh	softmax
	95.58%	100	10	100	Adam	tanh	softmax
	95.56%	100	100	200	rmsprop	tanh	softmax
	95.55%	200	100	100	Adam	tanh	softmax
	95.53%	100	10	100	Adam	sigmoid	softmax
	95.50%	200	10	200	AdaMax	tanh	softmax
	95.49%	100	10	100	rmsprop	tanh	softmax



(a) Number of neurons. (b) Batch size. (c) Number of epochs.



(d) Optimiser. (e) Activation I. (f) Activation II.

Fig. 6. Different hyperparameters for ANN.

Using Approach I, **adam** was selected for optimiser, **tanh** was selected for Activation I, and **softmax** was selected for Activation II. This selection was supported by the results from Approach II. In Fig. 6, **adam**, **tanh**, and **softmax** generated the highest accuracy as optimiser, Activation I, and Activation II. The final selection of the hyperparameters is shown in (Table 3).

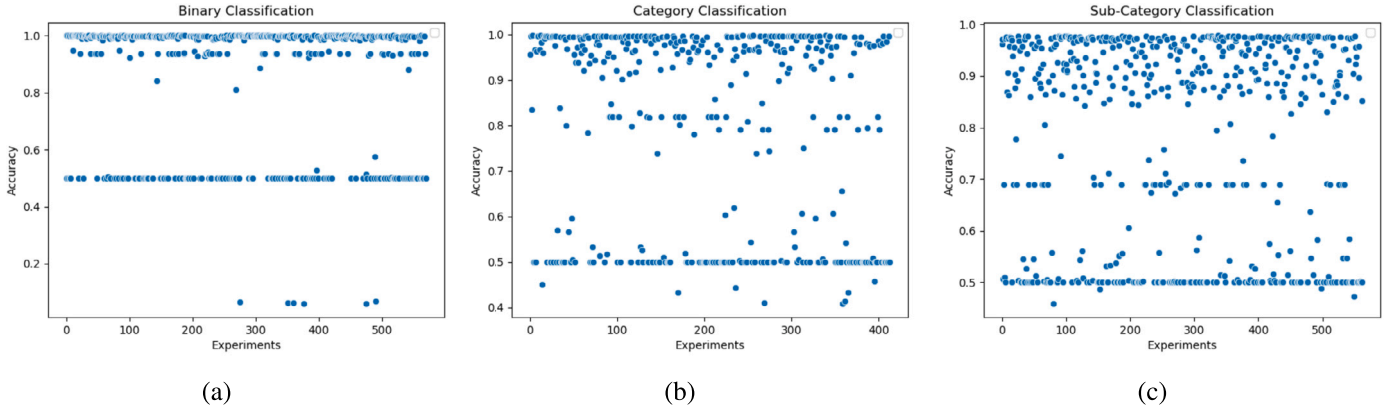


Fig. 7. Accuracy Results from (a) binary (b) category and (c) subcategory classification from full dataset experiments.

Table 3
Selected hyperparameters based on Approaches I and II results.

Hyperparameter	Selection
Epochs	200
Batch	100
Neurons	200
Optimiser	adam
Activation I	tanh
Activation II	softmax

4.3. Validation with full dataset

Although the full dataset was used for validation purposes, full experiments were conducted on this dataset to support the optimal hyperparameter selection from the subset dataset. The procedures are explained in the following steps. Through the grid search, 3,000 experiments were conducted on the full dataset with 625,783 records.

4.3.1. Binary classification

Among the 1000 experiments based on the grid search technique, 571 results were successfully returned using the predefined hyperparameters. 74% of experiments (421 out of 571) returned results above 90% accuracy. 63% of experiments (361 out of 571) returned an accuracy above 95%. 58% of experiments 334 out of 571) returned above 99%. The accuracy results from the 459 experiments were presented as a scatter plot shown in Fig. 7a.

The top 10 accuracy results with detailed hyperparameter setups on this binary classification are presented in Table 4. For all top 10 results, the accuracy rates stay between 99.951% and 99.956%. It is no surprise that all Activation II (the output layer) is sigmoid because this is a classification problem. The best accuracy is 99.9559%, with a mix of 200 neurons, AdaMax optimiser, ReLU activation function in hidden layers, batch size of 10, and 200 epochs. The activation function is ReLU in the top 5 accuracy results. The batch size of 10 is suitable for SGD optimiser while 100 for AdaMax.

4.3.2. Category classification

Using the predefined hyperparameters, the grid search returned 414 results out of 1000 experiments. 53% of experiments (221 out of 414) returned results above 90% accuracy. 47% of experiments (193 out of 414) returned an accuracy above 95%. 25% of experiments (103 out of 414) returned an accuracy above 99%. The accuracy results from the 449 experiments were presented as a scatter plot in Fig. 7b.

The top 10 accuracy results with detailed hyperparameter setups on this category classification (5 classes) are presented in Table 5. For all top 10 results, the accuracy rates stay between 99.7% and 99.72%, which are slightly worse than the binary classification (compared to Table 4). This is because, with the increase in classes, it is more difficult to accurately predict the correct class. It is expected that for multi-class classification, Activation II in the output layer is softmax activation function. The best accuracy is 99.7238%, with a mix of 100 epochs, 200 neurons, adam optimiser, tanh activation function in hidden layers and batch size of 100. Among all the optimisers, adam has the best performance in the top 10 accuracy results.

4.3.3. Subcategory classification

Using the predefined hyperparameters, the grid search returned 563 results out of 1000 experiments. 46% of experiments (259 out of 563) returned results above 90% accuracy. 31% of experiments (174 out of 563) returned an accuracy above 95%. No experiments returned above 99%. The accuracy results from the 563 experiments were presented as a scatter plot in Fig. 7c.

The top 10 results with detailed hyperparameter setups on this subcategory classification (9 classes) are presented in Table 6. For all top 10 results, the accuracy rates stay between 97.75% and 97.84%. These top 10 results are worse than the category and binary classification (compared to Tables 4 and 5) because with the increase of classes, it is more difficult to accurately predict the correct class. For multi-class classifi-

Table 4
Top 10 Best hyperparameter setups for binary classification using full dataset.

Accuracy	Neurons	Batch	Epoch	Optimiser	Activation I	Activation II
0.999559	200	100	200	AdaMax	ReLU	sigmoid
0.999553	200	10	200	SGD	ReLU	sigmoid
0.999550	200	100	100	AdaMax	ReLU	sigmoid
0.999540	100	10	200	SGD	ReLU	sigmoid
0.999530	100	100	200	AdaMax	ReLU	sigmoid
0.999521	200	100	100	adam	sigmoid	sigmoid
0.999519	200	10	200	SGD	tanh	sigmoid
0.999515	100	100	100	adam	tanh	sigmoid
0.999515	100	100	200	adam	sigmoid	sigmoid
0.999511	100	10	100	SGD	ReLU	sigmoid

Table 5
Top 10 best hyperparameter setups for category classification full dataset.

Accuracy	Neurons	Batch	Epoch	Optimiser	Activation I	Activation II
0.997238	200	100	100	adam	tanh	softmax
0.997212	100	100	200	adam	ReLU	softmax
0.997194	200	100	200	adam	ReLU	softmax
0.997151	200	10	200	AdaMax	tanh	softmax
0.997080	200	100	100	adam	ReLU	softmax
0.997070	100	100	200	adam	tanh	softmax
0.997041	100	10	200	SGD	ReLU	softmax
0.997036	200	10	100	AdaMax	tanh	softmax
0.997035	200	100	200	adam	sigmoid	softmax
0.997027	100	100	200	adam	sigmoid	softmax

Table 6
Accuracy Results from subcategory classification from full dataset experiments.

Accuracy	Neurons	Batch	Epoch	optimiser	Activation I	Activation II
0.978400	200	100	200	adam	sigmoid	softmax
0.978192	100	100	200	adam	ReLU	softmax
0.978087	200	100	200	adam	ReLU	softmax
0.977948	200	100	200	adam	tanh	softmax
0.977947	200	10	200	SGD	tanh	softmax
0.977938	200	100	200	AdaMax	tanh	softmax
0.977841	200	100	200	AdaMax	tanh	softmax
0.977635	100	10	200	SGD	ReLU	softmax
0.977635	200	10	200	AdaMax	tanh	softmax
0.977576	100	100	200	adam	tanh	softmax

cation, Activation II in the output layer is softmax activation function. The best accuracy is 99.84%, with a mix of 200 epochs, 200 neurons, adam optimiser, sigmoid activation function in hidden layers and batch size of 100. The epoch for all the top 10 accuracy cases is 200. Among all the optimisers, adam has the best performance in the top 4 accuracy results.

4.4. Optimal hyperparameter selection

The accuracy results of different hyperparameter settings are compared and presented in Fig. 8. The accuracy results are grouped by the classification types: Label (binary classification), Category (5-class classification), and Subcategory (9-class classification).

4.4.1. Number of epochs

The accuracy rates of the different predefined numbers of epochs are presented on a bar chart in Fig. 8. Observing the results, there was almost no difference in choosing either 100 or 200 epochs. However, the top 10 accuracy results listed in Tables 4–6 show that 200 epochs provide better results for all three classification types. There were 6 out of 10 top 10 best hyperparameter settings used 200 epochs in label classification (Table 4), 7 out of 10 in category classification (Table 5), and 10 out of 10 in subcategory classification (Table 6).

4.4.2. Batch size

The accuracy rates obtained by different batch sizes are presented on a bar chart in Fig. 8. The results from this figure show that 100 as batch size provided high accuracy in all prediction experiments. The top 10 accuracy results listed in Tables 4–6 also present that 100 epochs provide better results for all three classification types. There were 6 out of 10 top 10 best hyperparameter settings used 200 epochs in label classification (Table 4), 7 out of 10 in category classification (Table 5), and 7 out of 10 in subcategory classification (Table 6).

4.4.3. Number of neurons

The accuracy rates by the number of neurons are presented on a bar chart Fig. 8. From this figure, it can be seen that there was almost no difference in choosing either 100 or 200 neurons. However, the top 10 accuracy results listed in Tables 4–6 indicate that 200 epochs may provide better results for category and subcategory types. 5 out of 10 top 10 best hyperparameter settings used 200 epochs in label classification (Table 4), 6 out of 10 in category classification (Table 5), and 8 out of 10 in subcategory classification (Table 6).

4.4.4. Optimiser

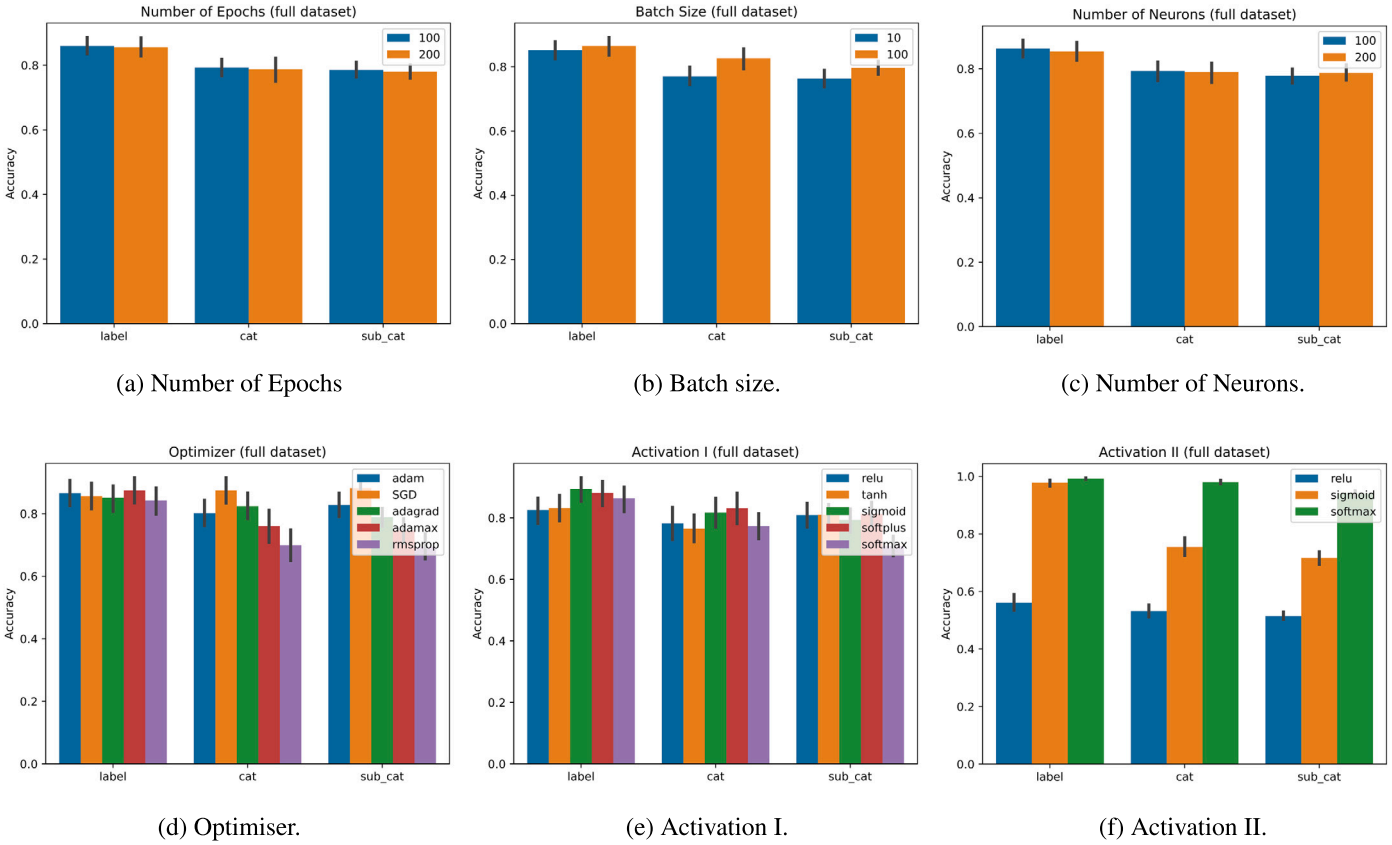
The accuracy rates by five optimiser functions are presented on a bar chart Fig. 8. By observing this figure, SGD could be the best hyperparameter option for all three classification types. In the top 10 accuracy results, AdaMax, adam, and SGD were used in the experiments. In label classification (Table 4), 3 out of top 10 accuracy results used AdaMax, 3 used adam, and 4 used SGD. In category classification (Table 5), 7 used adam, 3 used AdaMax, and 1 used SGD. In subcategory classification (Table 6), 5 used adam, 3 used AdaMax, and 2 used SGD.

4.4.5. Activation for input and hidden layers

The accuracy rates by five activation functions for input and hidden layers are presented on a bar chart Fig. 8. No particular activation function can be observed as the best hyperparameter option for all three classification types. Relu, Tahn, and Sigmoid were selected by the top 10 accuracy results as the activation function for the input and hidden layers. In label classification (Table 4), 6 out of 10 top accuracy results used ReLU, 2 used tanh, and 2 used sigmoid. In category classification (Table 5), 4 used ReLU, 4 used tanh, and 2 used sigmoid. In sub_category classification (Table 6), 3 used ReLU, 6 used tanh, and 1 used sigmoid. tanh may not be the best option for all classifications. It definitely shows the advantages in subcategory classification.

4.4.6. Activation for the output layer

The accuracy rates by five activation functions for output layers are presented on a bar chart Fig. 8. As shown in this figure, Tanh and Softplus are not suitable algorithms for the data structure from this full dataset. Only ReLU, Sigmoid, and Softmax completed the computation and provided results. Based on this figure, Softmax could be the best hyperparameter option for the accuracy function for the output layer. By observing the top 10 accuracy results tables, softmax was used in 10 out of 10 experiments in Table 5 and Table 6. Although Sigmoid was



(a) Number of Epochs (b) Batch size. (c) Number of Neurons.

(d) Optimiser. (e) Activation I. (f) Activation II.

Fig. 8. Accuracy for Three Classification Types with Different Hyperparameters.

Table 7
Binary classification based on the subset dataset.

Source : Subset Dataset
Accuracy : 0.997942

	Precision	Recall	f1-score	Support
Macro	0.948759	0.900779	0.923214	2499
Weighted	0.983063	0.983593	0.98313	2499
Label 1	0.987743	0.994891	0.991304	2349
Label 2	0.909774	0.806667	0.855124	150

Table 8
Binary classification based on full dataset.

Source : Full Dataset
Accuracy : 0.999326

	Precision	Recall	f1-score	Support
Macro	0.984392	0.934559	0.957921	156354
Weighted	0.990477	0.990573	0.990317	156354
Label 1	0.991354	0.998648	0.994988	146503
Label 2	0.977431	0.87047	0.920855	9851

used as the activation function for the output layer in the top 10 accuracy results for Label classification in Table 4, softmax is the optimal activation function for category and subcategory classification.

5. Results

This section provides complete experimental results for the subset dataset and full dataset for all three types of classification.

5.1. Binary classification

Tables 7 and 8 show the binary classification based on the subset dataset and full dataset, respectively. In Table 9, the f1-scores for macro f1-score, weighted f1-score, label 1 f1-score, and label 2 f1-score are 92.3214%, 98.313%, 99.1304%, and 85.5124%, respectively. In Table 10, the f1-scores for macro f1-score, weighted f1-score, label 1 f1-score, and label 2 f1-score are 95.7921%, 99.0317%, 99.4988%, and 92.0855%, respectively. In these two tables, the results for label 1 are higher than label 2 because there are more data for label 1 than label2 and the ANN could fit better to the majority class. In addition, the accuracy is 99.7942% for the subset and 99.9326% for the full dataset. This is expected because, with more data used for training, ANN has better performance in terms of precision, recall, and f1-score.

Table 9
Category classification based on subset dataset.

Source : Subset Dataset
Accuracy : 0.988722

	Precision	Recall	f1-score	Support
Macro	0.819231	0.792651	0.789534	2499
Weighted	0.897955	0.892757	0.888707	2499
Label 1	0.991736	1	0.995851	240
Label 2	0.530303	0.251799	0.341463	139
Label 3	0.955763	0.925769	0.940527	1657
Label 4	0.969231	0.84	0.9	150
Label 5	0.649123	0.945687	0.769831	313

5.2. Category classification

Tables 9 and 10 show the category classification based on the subset dataset and full dataset, respectively. In Table 9, the f1-scores for macro f1-score, weighted f1-score, and label 1 to label 5 f1-scores are 78.9534%, 88.8707%, 99.5851%, 34.1463%, 94.0527%,

Table 10
Category classification based on full dataset.

Source : Full Dataset Accuracy : 0.997283				
	Precision	Recall	f1-score	Support
Macro	0.931344	0.915716	0.922369	156354
Weighted	0.949057	0.947363	0.947741	156354
Label 1	0.999389	0.998035	0.998712	14760
Label 2	0.887314	0.848526	0.867487	8853
Label 3	0.967635	0.965106	0.966369	104057
Label 4	0.985411	0.870775	0.924553	9851
Label 5	0.816972	0.89614	0.854726	18833

Table 11
Subcategory classification based on subset dataset.

Source : Subset Dataset Accuracy : 0.954654				
	Precision	Recall	f1-score	Support
Macro	0.570248	0.553568	0.545961	2499
Weighted	0.628303	0.641857	0.626079	2499
Label 1	1	1	1	240
Label 2	0.511364	0.323741	0.396476	139
Label 3	0.259259	0.198113	0.224599	212
Label 4	0.3	0.255605	0.276029	223
Label 5	0.680312	0.687008	0.683643	508
Label 6	0.751625	0.809524	0.779501	714
Label 7	0.900709	0.846667	0.872852	150
Label 8	0.3	0.080357	0.126761	112
Label 9	0.428962	0.781095	0.553792	201

Table 12
Subcategory classification based on full dataset.

Source : Full Dataset Accuracy : 0.97767				
	Precision	Recall	f1-score	Support
Macro	0.700202	0.637441	0.626142	156354
Weighted	0.703244	0.731897	0.691957	156354
Label 1	0.999457	0.997561	0.998508	14760
Label 2	0.890365	0.86321	0.876577	8853
Label 3	0.321678	0.276278	0.297254	13852
Label 4	0.416138	0.051568	0.091764	14001
Label 5	0.792068	0.887703	0.837163	30553
Label 6	0.692747	0.884208	0.776855	45651
Label 7	0.960739	0.876865	0.916888	9851
Label 8	0.590615	0.064981	0.117081	5617
Label 9	0.638015	0.834594	0.723184	13216

90%, and 76.9831%, respectively. In Table 10, the f1-scores for macro f1-score, weighted f1-score, and label 1 to label 5 f1-scores are 92.2369%, 94.7741%, 99.8712%, 86.7487%, 96.6369%, 92.4553%, and 85.4726%, respectively. The accuracy in Table 9 is 98.8722% for the subset while 99.7283% in Table 10 for the full dataset. This is because ANN could extract more hidden knowledge with more data used for training.

5.3. Subcategory classification

Tables 11 and 12 show the Subcategory classification based on the subset dataset and full dataset, respectively. As seen in the tables, the accuracy in Table 11 is 95.4654% while 97.767% in Table 12. In Table 11, f1-scores for macro f1-score, weighted f1-score, and labels 1 to 9 f1-scores are 55.5961%, 62.6079%, 100%, 39.6476%, 22.4599%, 27.6029%, 68.3642%, 77.9501%, 87.2852%, 12.6761% and 55.3792%, respectively. In Table 12, f1-scores for macro f1-score, weighted f1-score, and labels 1 to 9 f1-scores are 63.6242%, 69.2957%, 99.8508%, 87.6577%, 29.7254%, 9.1764%, 83.7163%, 77.6855%, 91.6888%, 11.7081%, and 72.3184%, respectively. The results for label 1 are higher than label 2 because there are more data for label 1 than label 2 and the ANN could fit better to the majority class. The results in Table 12 are better than that in Table 11 because there are many parameters (weights between neurons) in ANN, it can better fit the problem with more data used.

In summary, with more data used for training, ANN achieves better results based on the full dataset in Tables 8–10 reinforces the optimal hyperparameter selection from the subset in Tables 7–11.

5.4. Explainability analysis

As discussed previously, explainability is an important factor when it comes to the adaptation of complex machine learning models by end users. MAMID is designed for the smart home scenario where users' trust in the system is crucial for users to adapt this security system in their homes. This section briefly discusses the explainability of the models based on the subset dataset.

The SHAP plots shown in Fig. 9 show the feature importance and summary plot for binary classification based on the subset dataset. In both plots, the features are placed on the Y-Axis according to their importance.

The feature importance plot shows how each feature impacts the detection of the classes: in this case for binary classification Class 1 is normal (blue) and Class 2 is attack (red). The impact of every feature on the detection of the attacks shows that Flow_Duration has the highest impact on the detection of these classes.

The summary plot shows the positive and negative relationships of the predictors with the target variable. The value on the X-Axis in the summary plot shows whether the effect of that value is associated with a higher or lower prediction: the colour shows high (in red) or low (in blue) for that observation. The low level of the Flow_Duration has a low and negative impact on attack detection. Similarly, the feature importance plot and summary plot for category classification are shown in Fig. 10.

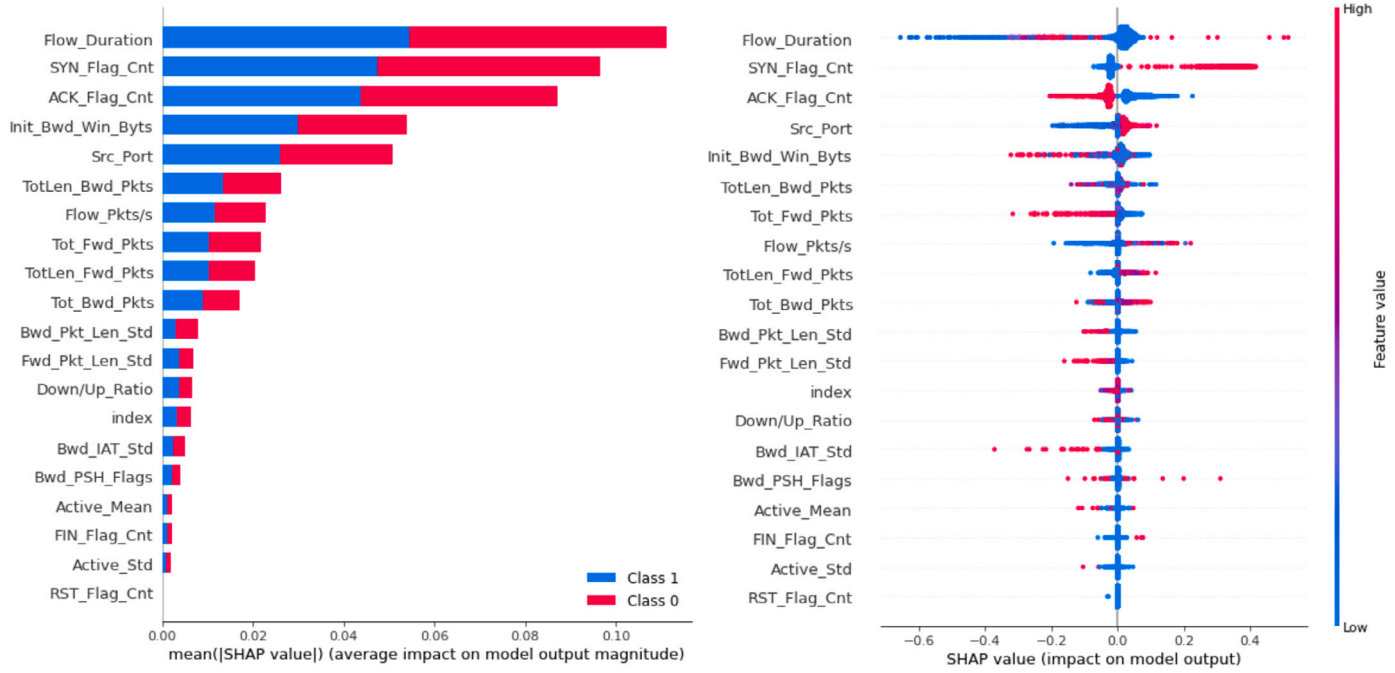
The force plot shown in Fig. 11 shows the effect of different features on a particular predicted value for binary classification based on the subset dataset. The given plot shows the baseline and the average predicted probability for a particular value to be classified as an attack is 0.07. The data sample has low predicted value as the increasing effect of TotLen_Fwd_pkts field is offset by decreasing effect of ACK_Flag_Cnt field.

5.5. Comparison of the proposed system with other related work

The performance of the proposed approach has been compared with recent existing work using the same data and the summary is presented in Table 13. It is evident from the table that only Ullah and Mahmoud (2020) have considered the subcategory level of attack classification for this dataset and the accuracy and F score is 88% with the decision tree method. Most of the work that considered IoTID20 dataset did not classify attacks at the subcategory level. As discussed in the Results section (Section 5), this work performed a comprehensive set of experiments and achieved high accuracy for all three levels of attack classification.

6. Conclusion and future work

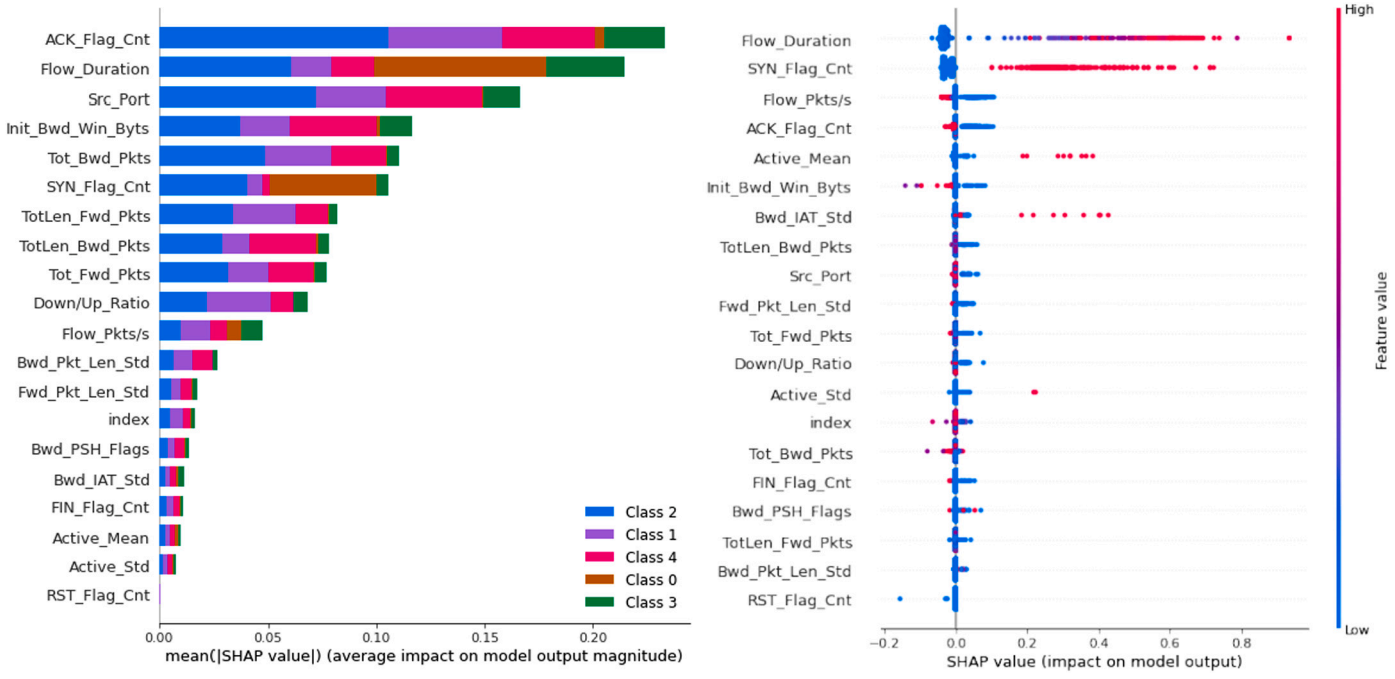
Accurate detection of intrusion is very important in cybersecurity. In this study, ANN is used for intrusion detection based on hyperparameter optimisation. To examine the effectiveness of different hyperparameters for ANN, experiments were conducted, including changing the number of epochs, batch size, number of neurons, optimiser, activation for input and hidden layers, and activation for the output layer. To validate the proposed model, a dataset with three different cases was considered: (1) Label classification (binary), (2) Category classification (5 classes), and (3) Subcategory classification (9 classes). The experimental results revealed that the proposed model with hyperparameter optimisation



(a) Feature importance plot.

(b) Summary plot.

Fig. 9. Shap plots for binary classification based on the subset dataset.



(a) Feature importance plot.

(b) Summary plot.

Fig. 10. Shap plots for category classification based on the subset dataset.

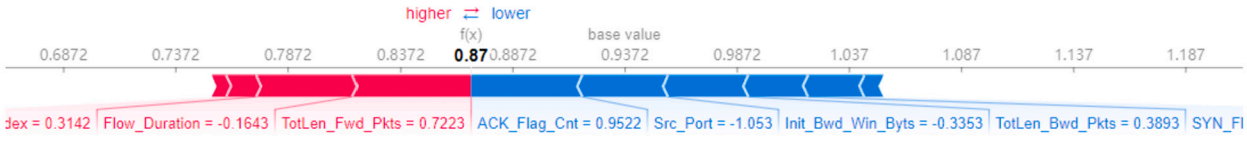


Fig. 11. Force plot for binary classification based on the subset dataset.

Table 13
Comparison With Other Related Work.

Research work	Machine Learning Approaches used	Classification label	Best Result
Ullah and Mahmoud (2020)	Multiple	Binary, Category and Subcategory	Subcategory = 88% F-Score, 88% Accuracy (DT)
Qaddoura et al. (2021)	Single Hidden Layer Feed-Forward Neural Network (SLFN)	Binary	98.42% Accuracy
Farah (2020)	Multiple	Binary and Category	Binary = 0.96 AUC, Category = 0.98 AUC (Ensemble)
Krishnan et al. (2021)	SVC, XGBoost and RF	Binary	99.97 Accuracy (XGBoost)
MAMID (proposed approach)	ANN with optimal Hyperparameters	Binary, Category and Subcategory	99.9%, 99.7%, and 97.7% Accuracy

has high performance for intrusion detection with 99.9%, 99.7%, and 97.7% accuracy for Label, Category, and Subcategory classification. The results also showed that the best hyperparameter of binary classification are [100, 10, 200, Adam, tanh, sigmoid] for [the number of neurons, the batch size, the number of epochs, optimiser, the activation I, the activation II] while [200, 10, 200, Adamax, tanh, softmax] and [100, 10, 200, AdaMax, tanh, softmax] for Category and Subcategory classification.

Although the proposed model is only tested for intrusion detection, it can also be applied to other problems (e.g. malicious website domain detection) and also other areas (e.g. disease detection and anomaly detection). Although the prediction accuracy rates are improved as compared to other work, there are still limitations of this model. ANN models are generally resource intensive and this model is not an exception. A method is proposed to search for optimal hyperparameters considering small data to minimise the overhead. Still, the model is computationally expensive to train which is a trade-off for high accuracy.

This work is completed using the recent IoT dataset, and future work involves the investigation of the effects of wrong labels and other data noise issues. In the future, the ANN structure, including the automated selection of the right hyperparameters to prevent over-fitting problems, needs to be investigated. Different feature selection or feature extraction options will also be considered as an extension of this work.

Funding

This research received no funding.

CRediT authorship contribution statement

Shaleeza Sohail: Conceptualization, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Writing – original draft, Writing – review & editing. **Zongwen Fan:** Conceptualization, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Writing – original draft, Writing – review & editing. **Xin Gu:** Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Fariza Sabrina:** Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing.

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.

Code availability

All code included in this study is available from the first author upon reasonable request.

Data availability

The dataset used for experimentation in this paper is available online. We have shared the link to the dataset in the paper.

References

Afaq, S., & Rao, S. (2020). Significance of epochs on training a neural network. *International Journal of Scientific and Technology Research*, *9*, 485–488.

Choraś, M., & Pawlicki, M. (2021). Intrusion detection approach based on optimised artificial neural network. *Neurocomputing*, *452*, 705–715. https://doi.org/10.1016/j.neucom.2020.07.138. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925231220319032.

Dolan, A., Ray, I., & Majumdar, S. (2020). Proactively extracting iot device capabilities: An application to smart homes. In A. Singhal, & J. Vaidya (Eds.), *Data and applications security and privacy XXXIV* (pp. 42–63). Cham: Springer International Publishing.

Doshi, S. (2019). Various optimization algorithms for training neural network. Retrieved from https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6.

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. *Journal of Biomedical Informatics*, *35*(5–6), 352–359.

Farah, A. (2020). *Cross dataset evaluation for IoT network intrusion detection* (Ph.D. thesis).

Gaikwad, D., & Thool, R. C. (2015). Intrusion detection system using bagging ensemble method of machine learning. In *2015 international conference on computing communication control and automation* (pp. 291–295). IEEE.

Hayou, S., Doucet, A., & Rousseau, J. (2019). On the impact of the activation function on deep neural networks training. Retrieved from arXiv:1902.06853.

Hodo, E., Bellekens, X., Hamilton, A., Dubouilh, P.-L., Iorkyase, E., Tachtatzis, C., & Atkinson, R. (2016). Threat analysis of iot networks using artificial neural network intrusion detection system. In *2016 international symposium on networks, computers and communications (ISNCC)* (pp. 1–6).

Huang, S.-W., Tsai, H.-P., Hung, S.-J., Ko, W.-C., & Wang, J.-R. (2020). Assessing the risk of dengue severity using demographic information and laboratory test results with machine learning. *PLOS Neglected Tropical Diseases*, *14*, Article e0008960. https://doi.org/10.1371/journal.pntd.0008960.

Islam, F. B., Akter, R., Kim, D.-S., & Lee, J.-M. (2020). *Deep learning based network intrusion detection for industrial internet of things* (pp. 418–421). Retrieved from https://journal-home.s3.ap-northeast-2.amazonaws.com/site/2020kics/presentation/0669.pdf.

Jabbar, M., Aluvalu, R., et al. (2017). Rfaode: A novel ensemble intrusion detection system. *Procedia Computer Science*, *115*, 226–234.

Jabbar, M. A., Aluvalu, R., & Reddy, S. S. S. (2017). Cluster based ensemble classification for intrusion detection system. In *Proceedings of the 9th international conference on machine learning and computing* (pp. 253–257).

Kantipudi, M. P., Aluvalu, R., & Velamuri, S. (2022). An intelligent approach of intrusion detection in mobile crowd sourcing systems in the context of iot based smart city. *Smart Science*, 1–7.

Koroniotis, N., Moustafa, N., Sitnikova, E., & Turnbull, B. (2019). Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Generation Computer Systems*, *100*, 779–796. https://doi.org/10.1016/j.future.2019.05.041. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167739X18327687.

Krishnan, S., Neyaz, A., & Liu, Q. (2021). Iot network attack detection using supervised machine learning. *International Journal of Artificial Intelligence and Expert Systems*, *10*, 18–32. Retrieved from https://www.cscjournals.org/library/manuscriptinfo.php?mc=IJAE-201.

Lee, I. (2020). *Securing the Internet of things: Concepts, methodologies, tools, and applications*. USA: IGI Global.

Lundberg, S., & Lee, S.-I. (2017). *A unified approach to interpreting model predictions* (pp. 1–10).

Maniriho, P., Niyigaba, E., Bizimana, Z., Twiringiyimana, V., Mahoro, L. J., & Ahmad, T. (2020). Anomaly-based intrusion detection approach for iot networks using machine learning. In *2020 international conference on computer engineering, network, and intelligent multimedia (CENIM)* (pp. 303–308).

Mehlig, B. (2019). *Artificial neural networks*. arXiv e-prints. arXiv–1901.

Moustafa, N., & Slay, J. (2015). Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)* (pp. 1–6).

Qaddoura, R., Al-Zoubi, A. M., Almomani, I., & Faris, H. (2021). A multi-stage classification approach for iot intrusion detection based on clustering with oversampling. *Applied Sciences*, *11*. Retrieved from https://www.mdpi.com/2076-3417/11/7/3022.

Smith, S. L., Kindermans, P., & Le, Q. V. (2017). Don't decay the learning rate, increase the batch size. CoRR, arXiv:1711.00489 [abs]. Retrieved from http://arxiv.org/abs/1711.00489. arXiv:1711.00489.

Subba, B., Biswas, S., & Karmakar, S. (2016). A neural network based system for intrusion detection and attack classification. In *2016 twenty second national conference on communication (NCC)* (pp. 1–6).

Taher, K. A., Mohammed Yasin Jisan, B., & Rahman, M. M. (2019). Network intrusion detection using supervised machine learning technique with feature selection. In *2019 international conference on robotics, electrical and signal processing techniques (ICREST)* (pp. 643–646).

Thamilarasu, G., & Chawla, S. (2019). Towards deep-learning-driven intrusion detection for the internet of things. *Sensors*, *19*. https://doi.org/10.3390/s19091977. Retrieved from https://www.mdpi.com/1424-8220/19/9/1977.

Tsaih, R.-H., Huang, S.-Y., Lian, M.-C., & Huang, Y. (2018). Ann mechanism for network traffic anomaly detection in the concept drifting environment. In *2018 IEEE conference on dependable and secure computing (DSC)* (pp. 1–6).

Ullah, I., & Mahmoud, Q. (2020). A scheme for generating a dataset for anomalous activity detection in iot networks. In *Canadian conference on AI*.

Woźniak, M., Zielonka, A., Sikora, A., Piran, M. J., & Alamri, A. (2020). 6g-enabled iot home environment control using fuzzy rules. *IEEE Internet of Things Journal*, *8*, 5442–5452.

Yang, Y., Wu, L., Yin, G., Li, L., & Zhao, H. (2017). A survey on security and privacy issues in internet-of-things. *IEEE Internet of Things Journal*, *4*, 1250–1258. https://doi.org/10.1109/JIOT.2017.2694844.

Zhang, G. (2000). Neural networks for classification: A survey. *IEEE Transactions on Systems, Man and Cybernetics. Part C, Applications and Reviews*, *30*, 451–462.