Article

A Machine Learning Based Two-Stage Wi-Fi Network Intrusion Detection System [†]

Abel A. Reyes , Francisco D. Vaca, Gabriel A. Castro Aguayo, Quamar Niyaz * and Vijay Devabhaktuni

Electrical and Computer Engineering Department, College of Engineering and Sciences, Purdue University Northwest, Hammond, IN 46323, USA; areyesan@pnw.edu (A.A.R.); vacaf@pnw.edu (F.D.V.); gcastroa@pnw.edu (G.A.C.A.); vjdev@pnw.edu (V.D.)

* Correspondence: qniyaz@pnw.edu; Tel.: +1-219-989-2685

[†] This paper is an extended version of our paper published in Francisco D Vaca and Quamar Niyaz, “An Ensemble Learning Based Wi-Fi Network Intrusion Detection System (WNIDS).” In Proceedings of the 17th IEEE International Symposium on Network Computing and Applications (NCA 2018), Cambridge, MA, USA, 1–3 November 2018.

Received: 16 September 2020; Accepted: 11 October 2020; Published: 15 October 2020

Abstract: The growth of wireless networks has been remarkable in the last few years. One of the main reasons for this growth is the massive use of portable and stand-alone devices with wireless network connectivity. These devices have become essential on the daily basis in consumer electronics. As the dependency on wireless networks has increased, the attacks against them over time have increased as well. To detect these attacks, a network intrusion detection system (NIDS) with high accuracy and low detection time is needed. In this work, we propose a machine learning (ML) based wireless network intrusion detection system (WNIDS) for Wi-Fi networks to efficiently detect attacks against them. The proposed WNIDS consists of two stages that work together in a sequence. An ML model is developed for each stage to classify the network records into normal or one of the specific attack classes. We train and validate the ML model for WNIDS using the publicly available Aegean Wi-Fi Intrusion Dataset (AWID). Several feature selection techniques have been considered to identify the best features set for the WNIDS. Our two-stage WNIDS achieves an accuracy of 99.42% for multi-class classification with a reduced set of features. A module for eXplainable Artificial Intelligence (XAI) is implemented as well to understand the influence of features on each type of network traffic records.

Keywords: Wi-Fi networks; network intrusion detection system (NIDS); machine learning (ML); explainable artificial intelligence (XAI)

1. Introduction

In the past few years, the number of smartphones and other portable computing devices (e.g., laptops, tablets) has exceeded traditional computers. There are around 3.5 billion smartphone users worldwide [1]. The computational power and capability of smartphones have increased several fold over the years. They can be used for almost all kinds of tasks performed in traditional PCs such as document sharing, instant messaging, online meetings, video streaming, web browsing, and word processing. In addition, there is a paradigm shift in computing—many software programs are available as cloud-based services nowadays. The increased number of portable devices and their interaction with cloud-based services have increased the Internet traffic significantly. The global Internet traffic is expected to grow from 73 exabytes per month in 2016 to 236 exabytes per month in 2021, of which 53% will be contributed by Wi-Fi networks as portable devices mostly use them to access the Internet [2].

The huge penetration and popularity of Wi-Fi networks also lead to security threats for attacks and intrusions. In 2016, a public Wi-Fi in Tel Aviv (Israel) was taken over by a hacker who discovered a

buffer overflow vulnerability in one of the Wi-Fi access points that would allow him to control it. In the same year, Kaspersky reported that more than 25% of public Wi-Fi worldwide were unsecured with potential threats of data breaches [3]. In 2017, a serious weakness in WPA2 (Wi-Fi Protected Access 2), the protocol used for securing Wi-Fi communication, was discovered that could be exploited using key re-installation attacks (KRACK) [4]. Using this attack, attackers can read the sensitive information transmitted in the network regardless the encryption. Another attack, Kr00k, similar to KRACK was discovered in early 2020 that could decrypt Wi-Fi frames transmitted from vulnerable devices [5].

Almost all organizations nowadays use Wi-Fi networks to provide Internet access to their workforce. The increasing concern among them regarding the security issues in Wi-Fi networks is noticeable. There is an imperative need to develop security tools specific for Wi-Fi networks, such as Wi-Fi Network Intrusion Detection System (WNIDS), based on reliable approaches and techniques to detect intrusions in them with high accuracy and confidence. In general, an IDS is a security tool that identifies any activity as intrusion that violates security policies within systems or networks. For future attack detection, analysis of historical data (e.g., system call, network traffic) in a system or network is performed in an IDS to identify the elements that caused attacks. The IDS must be extensible and adaptable to counter new attacks from the intruders as the nature of technology and techniques used for attacks are constantly evolving. For those reasons, machine learning (ML) based IDSs are well-suited to detect zero-days vulnerabilities using the information gathered in the past with a goal to distinguish normal activities from intrusions. ML-based IDSs are therefore gaining popularity among the research community and security firms. Towards this direction, we propose and implement an ML-based IDS for Wi-Fi networks by significantly extending our previous work [6]. The following are the contributions made in this work:

- A *two-stage* ML-based Wi-Fi network intrusion detection system (WNIDS) is proposed to enhance the detection accuracy using Aegean Wi-Fi Intrusion Dataset (AWID), a publicly available labeled dataset created from real Wi-Fi network traffic traces.
- A significant performance improvement in computational time and detection accuracy of an ML-based IDS is achieved by proper data analysis and selection of features that help understanding the relevant features for the classification task and filtering the noisy features from ML model development. We perform a detailed study of the AWID features and apply various feature selection techniques to improve its classification and computational performance.
- Having better understanding of the selected features allows efficient implementation of the countermeasures against the intrusions; Explainable Artificial Intelligence (XAI) has been used for this purpose.

The remainder of the paper is organized as follows: Section 2 provides an overview of recent research works related to ML-based Wi-Fi NIDS implementation. In Section 3, we discuss the AWID dataset and initial data preparation. Section 4 discusses the approach for our proposed work along with the experimental results. In Section 5, features are investigated for normal traffic and intrusions using XAI. Finally, the paper is concluded with possible future enhancements in Section 6.

2. Related Works

In this section, we review previous works (summarized in Table 1) related to machine learning (ML) based Wi-Fi NIDS development using the AWID dataset released in 2015 by Kolias et al. [7]. The authors developed an NIDS using the dataset as well. Their NIDS achieved an accuracy of 96.20% using J48 as the ML classification algorithm considering all the 154 features in the dataset and 96.26% after reducing the number of features from 154 to 20. Aminanto et al. reported an accuracy of 99.86% in their work after experimenting with three different ML algorithms including Artificial Neural Networks (ANN), Support Vector Machines (SVM), and Decision Tree on the reduced version of AWID dataset named as AWID-CLS-R [8]. They selected the best features using a binary ANN classifier to improve the detection of impersonation attack and normal traffic without considering the other two

classes presented in the dataset. The C4.5 variant of Decision Tree achieved the best performance with a set of 11 features. Udaya et al. improved the detection accuracy from 92.17% to 95.12% on the AWID-CLS-R dataset by implementing several ML algorithms with 111, 41, and 10 features obtained from different feature selection techniques [9]. They reported the highest accuracy with 41 features used with Random Tree as an ML algorithm. The model proposed by Kaleem et al. shows considerable improvement in the accuracy from 97.84% to 99.3% by reducing the number of features in an ANN on AWID-CLS-R data set [10]. However, the proposed model implements a binary classifier for normal and attack classes. Thing used an auto-encoder based deep learning approach on the AWID-CLS-R data set and achieved an accuracy of 98.67% for a multi-class classifier using all the features presented in the dataset [11]. A novel application of ladder network with the deep learning approach for WNIDS was proposed by Ran et al. with the purpose of differentiating difficult classification samples (e.g., impersonation and flooding attacks) from simple ones (e.g., injection attack samples) [12]. The proposed approach achieved 98.54% and 99.25% accuracy for multi-class and binary class classification of the AWID dataset. In another work, Aminanto et al. proposed the implementation of deep feature learning for feature extraction and selection (D-FES) to build a schema of features required for the best performance of an intrusion detection system [13]. The D-FES is based on stacked autoencoder (SAE). They complemented the experiment by trying different ML classifiers (ANN, SVM, and C4.5) and reported an accuracy of 99.97% for an SVM based binary classifier on a set of 21 features. They built the classifier for normal and impersonation attack samples. A quite similar approach was followed by Lee et al. that emphasized three phases in the proposed IDS, feature extraction using SAE, feature selection using Mutual Information and C4.8, and SVM based classification with gradient descent optimization [14]. The goal was to reduce the number of features as much as possible for classification. They ended up with a set of five features and reported an accuracy of 98.22%. However, the work focused only on impersonation attacks. Kim et al. used deep learning and unsupervised clustering to implement an IDS to detect impersonation attacks [15]. Two main tasks were discussed in the experiment, feature extraction through an SAE network with two hidden layers, and unsupervised clustering using the K-means algorithm. An accuracy of 94.81% was reported with a set of 50 features. Wang et al. performed a feature reduction task in [16]. They removed features with zero variance and features with missing values, ended up with a set of 71 features for training the ML model based on Deep Neural Network (DNN) with seven hidden layers. They achieved an average accuracy of 92.49% for 4-class classification.

Table 1. Summary of the related works for Wi-Fi NIDS using an AWID dataset.

Study	ML Algorithm	Features	Classes	Accuracy
Kim [15]	SAE & K-means	50	2	94.81%
Lee [14]	SAE & SVM	5	2	98.22%
Vaca [6]	Random Forest	36	2	99.11%
Ran [12]	Ladder Network	95	2	99.28%
Kaleem [10]	ANN	7	2	99.30%
Sydney [17]	FFDNN	26	2	99.66%
Aminanto [8]	ANN	11	2	99.86%
Aminanto [13]	SVM	21	2	99.97%
Wang [16]	DNN	71	4	92.49%
Udaya [9]	Random Tree	41	4	95.12%
Vaca [6]	Random Forest	36	4	95.88%
Kolias [7]	J48	20	4	96.20%
Ran [12]	Ladder Network	95	4	98.54%
Thing [11]	Deep Learning	154	4	98.67%
Zhou [18]	CSF-BA-Ensemble classifier	8	4	99.50%
Sydney [17]	FFDNN	26	4	99.77%

A few researchers considered different datasets along with the AWID dataset in their works. Sydney et al. used UNSW-NB15 and AWID datasets and extracted a reduced number of features from

them using feature selection techniques [17]. The UNSW-NB15 dataset is a multi-class dataset that contains nine types of intrusions in a traditional network environment [19]. Using the ExtraTree ML algorithm, 22 and 26 features were extracted for the UNSW-NB15 and AWID datasets, respectively. A Feed-Forward Deep Neural Network (FFDNN) based NIDS was implemented. The NIDS achieved accuracy of 77.16% and 87.10% for binary class classifier and multi-class classifier for the UNSW-NB15 dataset, whereas accuracies of 99.66% and 99.77% were reported for binary class and multi-class classifiers for the AWID dataset. Zhou et al. [18] used three different datasets: ISCX-2012 [20], NSL-KDD [21], and AWID in their work. The NIDS for each dataset was built using the same approach—a feature selection method was followed by ensemble learning based classification algorithms. The authors reported the accuracy of 99.90%, 99.89%, and 99.50% for ISCX-2012, NSL-KDD, and AWID datasets, respectively.

In our previous work [6], we applied ensemble learning algorithms for classification using AWID-CLS-R dataset and achieved an accuracy of 95.88% for multi-class classification. The accuracy came out 99.11% when all the attack classes were grouped into one attack class. We observed that two of the attack classes (impersonation and injection) were considerably misclassified to each other and led to a comparatively low-value for accuracy. Based on the observation, we propose the application of an additional ML model (in this work) that will be specifically used to distinguish impersonation and injection attack classes, if any records classified into them by the previous model, to improve the performance of our previous WNIDS, i.e., an ML based two-stage WNIDS. A two-stage approach was proposed for flow-based anomalous activity detection by Ullah et al. [22] as well. In their approach, a network trace is passed through two different ML models. The first model identifies if the trace is a normal or an attack, and the second model detects the attack type, if the first model has identified the trace as an attack. The authors considered the attacks in an IoT network for the NIDS implementation and used an IoT Botnet dataset [23]. They reported an accuracy of 99.90% for the multi-class classifier.

Most of the works discussed above implemented some sort of feature selection techniques and achieved a considerably high value for accuracy. However, they failed to offer any explanation for the importance of selected features for each specific traffic class and mostly lacked the reporting of computational time for their ML models, which is an important aspect for real-time intrusion detection in a network. A few works have recently used explainable AI (XAI) for intrusion detection systems. Marino et al. used an adversarial machine learning approach for XAI and evaluated their approach on an NSL-KDD 99 dataset [24]. In [25]; the authors used an XAI approach similar to us; however, they also applied it for an NSL-KDD intrusion dataset. The NSL-KDD dataset addresses different types of network intrusions with a different set of features [21].

3. Overview of AWID and Initial Data Preparation

We use an Aegean Wi-Fi Intrusion Dataset (AWID) to implement our two-stage WNIDS; therefore, a brief overview of the dataset is important. AWID is a publicly available Wi-Fi intrusion dataset created by Kolias et. al using Wi-Fi network frames of normal and intrusion traffic [7]. The authors released the intrusion dataset in different variants. We are using AWID-CLS-R version in our work, a reduced dataset in terms of size and number of records. AWID-CLS-R includes separate sets for training (AWID-CLS-R-Trn) and test (AWID-CLS-R-Tst) purposes. There are four major classes in the dataset: (i) Impersonation, (ii) Injection, (iii) Flooding, and (iv) Normal. The first three classes belong to intrusion attacks and the last one represents normal traffic. The dataset contains a total of 1,795,575 records for training in which 91% of the records are labeled as normal, and 575,643 for testing in which 92% of the records are labeled as normal. A distribution of the records for each class is shown in Table 2.

Table 2. Distribution of records in the AWID-CLS-R dataset for each traffic class.

Traffic Class	AWID-CLS-R-Trn	AWID-CLS-R-Tst
Normal	1,633,190 (91%)	530,785 (92.2%)
Impersonation	48,484 (2.7%)	8097 (1.4%)
Injection	48,522 (2.7%)	20,079 (3.5%)
Flooding	65,379 (3.6%)	16,682 (2.9%)

Each record in the dataset represents a Wi-Fi network frame and contains 154 features extracted from the frame and a class label (e.g., normal, flooding). The data types of features vary from timestamp and numbers to hexadecimal digits and strings. Having a dataset with such a variety of features, it is required to prepare the dataset that includes analysis, transformation, and removal of certain features to make the dataset suitable and compact for ML model development. In the dataset, there are features having the same values for all the records, and we removed those features. In addition, there are several features in the training dataset that have missing values in more than 50% of the records; those features were removed as well. Following that, the number of features reduced from 154 to 36 that includes 23 features with non-zero variance and non-missing values and 13 features with missing values, but in less than 50% of the records. Table 3 shows the list of reduced features in the dataset. For the features having missing values in less than 50% of the training records, we replaced the missing values with the most frequent values found in them. We discarded feature 13 as it is described as the MAC timestamp. Feature 19 is obtained by combining features 20 and 21, so it was also removed. After that, the total number of features was further reduced to 34. Features 29, 30, 31, 32, and 33 are MAC addresses which are represented as hexadecimal strings. They are associated with the Wi-Fi adapters of a sender host, access point, or receiver host and having a unique value for each network device. We replaced the hexadecimal values with 1 and their absence was marked as 0 in those features. The categorical class labels in the dataset were converted to numerical values as well.

Table 3. Features remained in the AWID dataset after initial data preparation.

Feature #	Feature's Name	Feature #	Feature's Name
0	frame.time_epoch	18	radiotap.dbm_antsignal
1	frame.time_delta	19	wlan.fc.type_subtype
2	frame.time_delta_displayed	20	wlan.fc.type
3	frame.time_relative	21	wlan.fc.subtype
4	frame.len	22	wlan.fc.ds
5	frame.cap.len	23	wlan.fc.frag
6	radiotap.length	24	wlan.fc.retry
7	radiotap.present.tsft	25	wlan.fc.pwrmtgt
8	radiotap.present.flags	26	wlan.fc.moredata
9	radiotap.present.channel	27	wlan.fc.protected
10	radiotap.present.dbm_antsignal	28	wlan.duration
11	radiotap.present.antenna	29	wlan.ra
12	radiotap.present.rxflags	30	wlan.da
13	radiotap.present.mactime	31	wlan.ta
14	radiotap.datarate	32	wlan.sa
15	radiotap.channel.freq	33	wlan.bssid
16	radiotap.channel.type.cck	34	wlan.frag
17	radiotap.channel.type.ofdm	35	wlan.seq

4. Two-Stage WNIDS Implementation and Experimental Results

A large number of impersonation attack records were incorrectly classified into injection attack records in the 4-class ML model of our previous work [6]. To overcome this limitation, we considered a *two-stage* approach that works in sequence for the proposed WNIDS and developed an ML model for each stage separately. For the first stage, we developed a 3-class classification model where

impersonation and injection attack classes are unified as a single class and the other two classes include the traffic for normal and flooding traffic. To develop the ML model for the first stage, we labeled the records for impersonation and injection records with the same label in training and test sets of reduced AWID. For the second stage, an ML model was developed to classify a record as either impersonation or injection attack. To develop the ML model for this stage, we selected the impersonation and injection records only from the training and test sets of the reduced AWID. In our two-stage WNIDS, a record is processed at the first stage. If the first stage ML model classifies it as normal or flooding, then it is predicted as-is. If the record is classified into the unified class, then it is sent to the second stage ML model to be predicted as impersonation or an injection attack. Algorithm 1 briefly lists the approach used in our two-stage WNIDS. The descriptions for feature selection and ML model development for each stage are as follows:

Algorithm 1: Pseudocode for two-stage WNIDS

Input: A captured Wi-Fi frame

Output: The Wi-Fi frame labeled with normal or specific attack type

```

1 record := a vector created from the frame by extracting features for first stage ML model
2 label := WNIDS_ML_model1 (record) /* label will be "Normal", "Flooding", or "Imper/Inject"
   traffic for first stage ML model */
3 if label == "Imper/Inject" then
4   record1 := a subset of record vector containing features for second stage ML model
5   label = WNIDS_ML_model2 (record1) /* label will be only "Impersonation" or "Injection"
   for second stage */
6 return (frame, label)

```

4.1. Feature Selection

In feature selection, features that contribute significantly in output prediction are identified and selected [26]. It reduces the number of features, model over-fitting, and processing time along with the improvement in model accuracy. As discussed in Section 3, we reduced the number of features from 154 to 34 after initial data preparation. After this, we performed feature selection for ML models for both stages. We used different feature selection techniques as discussed below and compared their performances to select one of them.

- **Recursive Feature Elimination (RFE):** In this technique, we start with the full list of features and then recursively remove the weakest feature up to the point until the performance is not degraded. The technique tends to eliminate dependencies and collinearity that may exist among the features [27].
- **Feature Importance:** Relative importance of each feature for the ML model prediction is computed and ranking is given accordingly. Features are selected based on this ranking [28].
- **Chi-square Test:** Chi-square test is used in statistics to test the independence of two events. In the feature selection task, the goal is to select the features that are highly dependent on the prediction. The higher the chi-square value (or lower p -value), the more the feature is dependent on the response and suitable candidate for ML model development [29].
- **Feature Correlation:** It describes how features are related to each other. If we determine a set of features highly related to each other, i.e., features follow a correlated pattern for their values in the set, then the set can be replaced by only one of its features [30].
- **Particle Swarm Optimization (PSO):** Feature selection is a computationally expensive task for those ML problems in which the datasets have a large number of features. Optimization based feature selection techniques help with finding an optimal set of features within a reasonable time in such cases. Particle swarm optimization (PSO) is a nature inspired swarm intelligence

based optimization algorithm modeled from the movement of a flock of birds [31]. PSO based feature selection techniques have been studied and found to be promising compared to other optimization approaches [32,33]. For this reason, we implemented a PSO-based feature selection technique by adopting the approach discussed in [33]. In this approach, each particle is expressed in a d -dimensional space, where d is the number of features in the dataset. For the implementation, we used Pyswarms, a Python library for PSO [34].

We first performed feature selection for the first stage ML model. We used a Random Forest ML algorithm (as it performed best in our previous work [6]) to test each feature selection method. As observed from Table 4, the best performance was achieved using 19 features selected from feature correlation analysis with the accuracy of 99.46% and 96.19% for training and test sets, respectively. Figure 1 shows the correlation heatmap of features. From the figure, a high correlation is observed among the following groups: {1, 2}, {4, 5}, {6, 7, 8, 9, 10, 11, 12}, {17, 18}, {30, 31, 32, 33}. Since features 6–12 are highly correlated, we considered one feature from the group and discarded the remaining. Similarly, we selected one feature from each feature group {4, 5} and {17, 18} due to their high correlation and only one feature was selected from features 30–33 and the remaining were discarded. We also discarded features 0–3 as they represent time. Table 5 lists the 19 features selected from feature correlation analysis for the first stage ML model.

Figure 1. Correlation heatmap of features for the first stage ML model.

Table 4. Performance of feature selection techniques for the first stage ML model.

Feature Selection Method	# of Features	Training Accuracy	Test Accuracy
Recursive Feature Elimination	19	98.28%	95.59%
Feature Importance	8	98.87%	95.05%
Chi-square Test	24	98.84%	95.43%
Feature Correlation	19	99.46%	96.19%
Particle Swarm Optimization	23	97.47%	95.77%

Table 5. Features selected for the first stage ML model.

Feature #	Feature's Name	Feature #	Feature's Name
0	frame.len	10	wlan.fc.retry
1	radiotap.present.flags	11	wlan.fc.pwrmtgt
2	radiotap.datarate	12	wlan.fc.moredata
3	radiotap.channel.freq	13	wlan.fc.protected
4	radiotap.channel.type.cck	14	wlan.duration
5	radiotap.dbm antsignal	15	wlan.ra
6	wlan.fc.type	16	wlan.da
7	wlan.fc.subtype	17	wlan.frag
8	wlan.fc.ds	18	wlan.seq
9	wlan.fc.frag		

For the second stage ML model, we performed feature selection from the set of 19 features selected for the first stage. The results for the second stage feature selection are shown in Table 6. It is evident from the table that the best performance is achieved from the Feature Importance method. In Figure 2, the top 10 features with high importance are shown. Out of those 10 features, we selected features with the importance value greater than 1.0 resulted a total of seven features for the second stage: {4, 18, 21, 22, 27, 28, 35} from Table 3 or {0, 5, 7, 8, 13, 14, 18} from Table 5. Table 7 lists the features selected for the second stage ML model.

Figure 2. Top 10 high importance features for the second stage ML model.**Table 6.** Performance of feature selection methods for the second stage ML model.

Feature Selection Method	# of Features	Training Accuracy	Test Accuracy
Recursive Feature Elimination	10	100%	48.38%
Feature Importance	7	100%	99.99%
Chi-square Test	16	97.68%	45.38%
Correlation Analysis	16	99.91%	49.36%
Particle Swarm Optimization	15	99.92%	49.46%

Table 7. Features selected for the second stage ML model.

Feature #	Feature's Name	Feature #	Feature's Name
0	frame.len	4	wlan.fc.protected
1	radiotap.dbm_antsignal	5	wlan.duration
2	wlan.fc.subtype	6	wlan.seq
3	wlan.fc.ds		

4.2. ML Model Development for Two Stages

The algorithms used in ML model development of our two-stage WNIDS are as follows:

- **Bootstrap Aggregation (Bagging):** It is an ensemble method that combines predictions from multiple ML model to make better predictions than the individual models. In bagging, random subsets of data points are assigned to individual ML models during training. When a new data point is evaluated for prediction, the value assigned to it is the most common value predicted by the individuals' models [35].
- **Random Forest:** It is an improved version of decision tree classifier, where several decision trees are constructed during the training time, and each of them uses a random subset of features from the full features set of features during model development. The random samples of features help to guarantee a low variance in the results [36].
- **Extra Trees:** It is also known as Extremely Randomized Trees, and the features and sub-datasets are selected at random, which makes it different from Random Forest. The splits are randomly chosen for each feature, demanding less computational effort compared to Random Forest [37].
- **Extreme Gradient Boosting (XGBoost):** It is an efficient tool implemented based on a Gradient Boosting classifier. Gradient Boosting shows effective performance in the form of an ensemble of weak prediction models. XGBoost offers the same reliability as Gradient Boosting, but with less computational expenses [38].
- **Naive Bayes (NB):** It is based on Bayes' Theorem with a strong assumption of features independence and selects the best hypothesis for a given data based on the probabilities. It uses Gaussian distribution to extend its application for real-valued attributes [39].

All of the above-mentioned algorithms were used for model development for both the stages to classify the records into four classes using the selected features. Accuracy and training time were recorded for the models that helped us to finalize the best performing algorithm for ML models of each stage in the proposed WNIDS. We implemented and evaluated the models in a desktop computer equipped with Intel(R) Core(TM) i5-7400 CPU @ 3.00 GHz and 12 GB RAM.

Table 8 shows the performance of ML algorithms on the training sets of both the stages. It is observed that Random Forest achieved the highest accuracy (99.57%) compared to other classifiers in the first stage. Similarly, the ML algorithm for the second stage was selected after comparing the performance of different ML models. The best performance was achieved using Naive Bayes algorithm with an accuracy of 100% and training time of 0.22 s. After the performance evaluation, we finalized Random Forest and Naive Bayes ML models for the first and second stages, respectively. We tested each ML model with the test sets. Figures 3 and 4 show the confusion matrices for the first stage and second stage ML models on the test sets when they were evaluated independently. The 3-class ML model of first stage achieved an accuracy of 99.41%. The 2-class ML model of second stage that classifies impersonation and injection attacks achieved an accuracy of 99.99% ($\approx 100\%$).

The proposed WNIDS combines the two-stage ML models in a sequence. As described earlier, the first model takes a record as input and classifies it as normal, flooding, or unified impersonation and injection traffic class. If the record is classified as the unified class, then it will be evaluated by the second ML model; otherwise, the prediction will be directly reported as an output of the WNIDS. The record sent to the second ML model will be classified as impersonation or injection class and reported as the output of the WNIDS. The two-stage WNIDS was evaluated with the AWID test dataset.

Figure 5 shows the confusion matrix of 4-class classification by the two-stage WNIDS. The well-known classification metrics for the same is shown in Figure 6. From the figure, it is evident that the proposed model performs exceptionally well for classifying impersonation, injection, and normal records with an f-score of almost 1.0. The WNIDS achieved an accuracy of **99.42%** and outperforming all the works reported in Table 1 for 4-class classification except [17,18]. However, the authors in [17] did not report computational time and f-measure values for each traffic class. In addition, we achieved the performance with 19 features compared to their 26 features. In Figure 5, it is observed that a significant number of flooding attack records are incorrectly classified into normal due to which the f-score for flooding attack is around 0.8 in Figure 6. Although flooding attack affects the availability of the network, it is not as critical as impersonation or injection attacks. The latter ones compromise confidentiality and integrity of data. Although the approach in [18] achieved an accuracy of 99.5% with a small set of features, our model performed very well for impersonation and injection attack classification. The two-stage WNIDS achieved detection accuracy of 99.86% and 99.5% for injection and impersonation attacks, whereas the approach in [18] achieved 100% and 93% for them.

Table 8. Performance of ML algorithms on the training set at each stage.

ML Algorithm	First, Stage ML Model		Second Stage ML Model	
	Accuracy	Training Time	Accuracy	Training Time
Bagging	99.41%	4854.25 s	99.99%	54.63 s
Random Forest	99.57%	2168.85 s	99.99%	30.94 s
Extra Trees	99.55%	1802.91 s	99.99%	29.78 s
XGBoost	99.49%	854.50 s	99.99%	4.35 s
Naive Bayes	87.85%	320.79 s	100.00%	0.22 s

Figure 3. Confusion matrix of the first stage model.

Figure 4. Confusion matrix of the second stage ML model.

Figure 5. Confusion matrix of the two-stage WNIDS.

Figure 6. Performance metrics of the two-stage WNIDS.

5. Explainable Artificial Intelligence (XAI)

Several ML algorithms work efficiently to create models for predicting the outcome with high accuracy. However, the models look like a black box from outside without providing enough information to understand the reasons behind a prediction.

Explainable Artificial Intelligence (XAI) is based on the desire to have a computer system working as it is expected, but also have a transparent explanation for its decisions on what it is producing [40]. In other words, XAI is a field that aims to explain how the black box of an AI system is making decisions to produce the outputs. In this work, we implement XAI to get a better understanding of the classification process of WNIDS for normal and intrusion traffic, i.e., identifying the most influential features that classify a record into a specific class. To implement XAI, we rely on the SHapley Additive exPlanations (SHAP) [41] method available for explaining the outputs of ensemble tree models—the one that we used for the first stage ML model. SHAP library offers a variety of tools for global and local explanation with graphical outputs to get a better insight into the ML black box.

In our previous work [6], we used a Random Forest based model for 4-class WNIDS. Figure 7 shows the confusion matrix for the model. The accuracy reported for this ML model for the AWID test dataset was 95.88%. As seen in the confusion matrix, the majority of the impersonation records were incorrectly classified as injection records. Similarly, many flooding and injection records were labeled as normal. Having this outcome, we decided to analyze the important features that identify records for a particular class with the help of SHAP. We found a global explanation that includes all the records from the dataset for each feature. Figure 8 shows the influence of each feature for predicting records of a particular class. For example, the records predicted as normal are mostly influenced by feature 0, 2, 6, 7, and 14 (listed in Table 5). Similarly, the most influential features for injection and impersonation records are {7, 8, 13, 14} and {18, 7, 0, 5}, respectively. In addition to understanding the global explanation, a local analysis was performed for a few records using SHAP. Figure 9 shows the probability for a particular record to be predicted as either one of the attack or normal class. It is observed from the figures that the features interact with each other, pushing right or left the probability value until the probability reaches a final value to classify the record as one

of the classes. Figure 9a shows the local explanation for a record with a flooding label. From top to bottom, the figure shows how likely it is for that record to be predicted as flooding, impersonation, injection, or normal. The interaction of values for each feature gave zero probability to be predicted as impersonation, injection, or normal, but a probability of 1.0 for flooding attack. A similar inference can be drawn for the records of other classes from Figure 9b–d.

Figure 7. Confusion matrix of Random Forest based model for 4-class WNIDS in [6].

Figure 8. Influence of features on predicting the records for each class.

Figure 9. Local explanation using XAI-SHAP.

Although the influence of features on the prediction of each class can be seen in Figure 8, we provide a deeper inspection of the global impact of feature values on the prediction of each type of records in Figure 10. For example, the top five features that have a strong impact on flooding records are {6, 0, 5, 2, 16} (listed in Table 5), in which features 6, 0, and 2 have low values and features 5 and 16 have high values as shown in Figure 10a. These characteristics are common for almost all the records labeled as flooding. Figure 10b–d provide similar information for other classes. Table 9 lists the five most influential features and their values for the classification of each class based on Figures 8 and 10. It is important to mention that the features reported as influential for impersonation and injection attacks by the XAI analysis are included in the second stage ML model as shown in Table 7.

Figure 10. Global impact of feature values on the prediction of each type of records.

Table 9. Top 5 influential features and their values for each traffic class.

Class Name	Feature	Value	Impact
Flooding	wlan.fc.type (6)	High	Negative Low
	frame.len (0)	Low	Positive Low
	radiotap.dbm_antsignal (5)	High	Positive High
	radiotap.datarate (2)	Low	Positive Low
	wlan.da (16)	High	Positive Low

Table 9. Cont.

Class Name	Feature	Value	Impact
Impersonation	wlan.seq (18)	High	Negative High
	wlan.fc.subtype (7)	Low/High	Positive High
	radiotap.dbm_antsignal (5)	high	Positive High
	frame.len (0)	Low	Positive High
	wlan.fc.pwrmtgt (11)	High	Positive Low
Injection	wlan.duration (14)	High	Positive High
	wlan.fc.protected (13)	High	Positive High
	wlan.fc.subtype (7)	Low	Positive High
	wlan.fc.ds (8)	Low/High	Positive High
	frame.len (0)	Low	Negative High
Normal	frame.len (0)	Low	Positive High
	wlan.fc.subtype (7)	Low/High	Positive High
	wlan.duration (14)	High	Positive Low
	radiotap.datarate (2)	High	Positive Low
	wlan.fc.type (6)	High	Positive Low

6. Conclusions and Future Work

In this work, a machine learning based two-stage WNIDS was implemented to improve the detection accuracy for impersonation and injection attacks in a Wi-Fi network. Two separate ML models that work together in a sequence were developed for the WNIDS. The first stage ML model is a 3-class classification model based on Random Forest that identifies a Wi-Fi network frame (record) as flooding, normal, or unified injection and impersonation class. The second stage ML model is based on Naive Bayes that classifies the frame records, identified as a unified class by the first stage ML model, into impersonation or injection class. The combination of these two-stage ML models were used to classify the frame records into one of the 4-classes and achieved an accuracy of **99.42%** on the AWID-CLS-R test dataset. Nineteen features, selected after the evaluation of several feature selection methods, have been used for the WNIDS implementation. Out of those 19 features, only seven features have been used in the second stage ML model. XAI was implemented to have an insight for the decisions made by the first stage ML model, mostly for the cases where the records were predicted as impersonation or injection. The features that significantly contribute to their prediction were determined. This set of features almost matched the one identified by the feature selection method for the second stage ML model. In the proposed WNIDS, a number of flooding attack records in the test dataset have been identified as normal records. Although this limitation of the WNIDS may affect the availability of the network in the case of flooding attack, confidentiality, and integrity of user information will remain intact. The future work will attempt to overcome this limitation by further improving the performance of WNIDS. A real-world deployment of the proposed WNIDS in an organizational Wi-Fi network will be investigated as well.

Author Contributions: Conceptualization, Q.N., A.A.R. and F.D.V.; Methodology, A.A.R., F.D.V. and Q.N.; Software, A.A.R., F.D.V. and G.A.C.A.; Validation, A.A.R. and G.A.C.A.; Writing—original draft preparation, A.A.R.; Writing—review and editing, Q.N., G.A.C.A. and V.D.; Supervision, Q.N. and V.D.; Resources, V.D. and Q.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. O'Dea, S. Number of Smartphone Users Worldwide from 2016 to 2021. 2020. Available online: <https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/> (accessed on 10 September 2020).

2. Cisco. Cisco VNI Global—2021 Forecast Highlights. Available online: https://www.cisco.com/c/dam/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2021_Forecast_Highlights.pdf (accessed on 10 September 2020).
3. Biggest Wi-Fi Hacks of Recent Times—Lessons Learnt? Available online: <https://www.titanhq.com/biggest-wi-fi-hacks-of-2016-lessons-learnt/> (accessed on 10 September 2020).
4. Vanhoef, M.; Piessens, F. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. In Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS), Dallas, TX, USA, 30 October–3 November 2017.
5. Cermak, M.; Stefan Svorencik, S.; Lipovsky, R. KR00K-CVE-2019-15126, Serious Vulnerability Deep Inside Your Wi-Fi Encryption. 2020. Available online: https://www.welivesecurity.com/wp-content/uploads/2020/02/ESET_Kr00k.pdf (accessed on 10 September 2020).
6. Vaca, F.D.; Niyaz, Q. An ensemble learning based wi-fi network intrusion detection system (wnids). In Proceedings of the 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 1–3 November 2018; pp. 1–5.
7. Koliass, C.; Kambourakis, G.; Stavrou, A.; Gritzalis, S. Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Commun. Surv. Tutor.* **2015**, *18*, 184–208. [CrossRef]
8. Aminanto, M.E.; Tanuwidjaja, H.; Yoo, P.D.; Kim, K. Weighted feature selection techniques for detecting impersonation attack in Wi-Fi networks. In Proceedings of the Symposium on Cryptography and Information Security (SCIS), Naha, Japan, 24–27 January 2017; pp. 1–8.
9. Thanthrige, U.S.K.P.M.; Samarabandu, J.; Wang, X. Machine learning techniques for intrusion detection on public dataset. In Proceedings of the 2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Vancouver, BC, Canada, 15–18 May 2016; pp. 1–4.
10. Kaleem, D.; Ferens, K. A cognitive multi-agent model to detect malicious threats. In Proceedings of the 2017 International Conference on Applied Cognitive Computing (ACC'17), Las Vegas, NV, USA, 17–20 July 2017.
11. Thing, V.L. IEEE 802.11 network anomaly detection and attack classification: A deep learning approach. In Proceedings of the 2017 IEEE Wireless Communications and Networking Conference (WCNC), San Francisco, CA, USA, 19–22 March 2017; pp. 1–6.
12. Ran, J.; Ji, Y.; Tang, B. A Semi-Supervised learning approach to IEEE 802.11 network anomaly detection. In Proceedings of the 2019 IEEE 89th Vehicular Technology Conference (VTC 2019-Spring), Kuala Lumpur, Malaysia, 28 April–1 May 2019; pp. 1–5.
13. Aminanto, M.E.; Choi, R.; Tanuwidjaja, H.C.; Yoo, P.D.; Kim, K. Deep abstraction and weighted feature selection for Wi-Fi impersonation detection. *IEEE Trans. Inf. Forensics Secur.* **2017**, *13*, 621–636. [CrossRef]
14. Lee, S.J.; Yoo, P.D.; Asyhari, A.T.; Jhi, Y.; Chermak, L.; Yeun, C.Y.; Taha, K. IMPACT: Impersonation attack detection via edge computing using deep autoencoder and feature abstraction. *IEEE Access* **2020**, *8*, 65520–65529. [CrossRef]
15. Kim, K.; Aminanto, M.E.; Tanuwidjaja, H.C. Deep Feature Learning. In *Network Intrusion Detection using Deep Learning. SpringerBriefs on Cyber Security Systems and Networks*; Springer: Singapore, 2018; pp. 47–68.
16. Wang, S.; Li, B.; Yang, M.; Yan, Z. Intrusion Detection for WiFi Network: A Deep Learning Approach. In *Wireless Internet. WICON 2018. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*; Chen, J.L., Pang, A.C., Deng, D.J., Lin, C.C., Eds.; Springer: Cham, Switzerland, 2018; pp. 95–104.
17. Kasongo, S.M.; Sun, Y. A deep learning method with wrapper based feature extraction for wireless intrusion detection system. *Comput. Secur.* **2020**, *92*, 101752. [CrossRef]
18. Zhou, Y.; Cheng, G.; Jiang, S.; Dai, M. Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Comput. Netw.* **2020**, *174*, 107247. [CrossRef]
19. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015; pp. 1–6.
20. Tavallaee, M. An Adaptive Hybrid Intrusion Detection System. Ph.D. Thesis, Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada, 2011.
21. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.

22. Ullah, I.; Mahmoud, Q.H. A Two-Level Flow-Based Anomalous Activity Detection System for IoT Networks. *Electronics* **2020**, *9*, 530. [CrossRef]
23. Ullah, I.; Mahmoud, Q.H. IoT-Botnet Dataset 2020. 2020. Available online: <https://sites.google.com/view/iotbotnetdataset> (accessed on 10 September 2020).
24. Marino, D.L.; Wickramasinghe, C.S.; Manic, M. An adversarial approach for explainable ai in intrusion detection systems. In Proceedings of the IECON 2018–44th Annual Conference of the IEEE Industrial Electronics Society, Washington, DC, USA, 21–23 October 2018; pp. 3237–3243.
25. Wang, M.; Zheng, K.; Yang, Y.; Wang, X. An Explainable Machine Learning Framework for Intrusion Detection Systems. *IEEE Access* **2020**, *8*, 73127–73141. [CrossRef]
26. Brownlee, J. Feature Selection in Python with Scikit-Learn. 2014. Available online: <https://machinelearningmastery.com/feature-selection-in-python-with-scikit-learn/> (accessed on 10 September 2020).
27. Recursive Feature Elimination. Available online: https://scikit-learn.org/stable/modules/feature_selection.html (accessed on 10 September 2020).
28. Feature Importances with Forests of Trees. Available online: https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html (accessed on 10 September 2020).
29. Gajawada, S.K. Chi-Square Test for Feature Selection in Machine Learning. 2014. Available online: <https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223> (accessed on 10 September 2020).
30. Shaikh, R. Feature Selection Techniques in Machine Learning with Python. 2018. Available online: <https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e> (accessed on 10 September 2020).
31. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
32. Xue, B.; Zhang, M.; Browne, W.N. New fitness functions in binary particle swarm optimisation for feature selection. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, Australia, 10–15 June 2012; pp. 1–8.
33. Vieira, S.M.; Mendonça, L.F.; Farinha, G.J.; Sousa, J.M. Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients. *Appl. Soft Comput.* **2013**, *13*, 3494–3504. [CrossRef]
34. Miranda, L. PySwarms: A Particle Swarm Optimization Library in Python. 2017. Available online: <https://lvmiranda921.github.io/projects/2017/08/11/pyswarms/> (accessed on 10 September 2020).
35. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
36. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
37. Ceballos, F. An Intuitive Explanation of Random Forest and Extra Trees Classifiers. 2019. Available online: <https://towardsdatascience.com/an-intuitive-explanation-of-random-forest-and-extra-trees-classifiers-8507ac21d54b> (accessed on 10 September 2020).
38. Introduction to Boosted Trees. Available online: <https://xgboost.readthedocs.io/en/latest/tutorials/model.html> (accessed on 10 September 2020).
39. Brownlee, J. Naive Bayes for Machine Learning. 2016. Available online: <https://machinelearningmastery.com/naive-bayes-for-machine-learning/> (accessed on 10 September 2020).
40. Schmelzer, R. Understanding Explainable AI. 2019. Available online: <https://www.forbes.com/sites/cognitiveworld/2019/07/23/understanding-explainable-ai/#74bdb29d7c9e> (accessed on 10 September 2020).
41. Schmelzer, R. Shap. 2018. Available online: <https://shap.readthedocs.io/> (accessed on 10 September 2020).

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).