Detecting Interference in Wireless Sensor Network Received Samples: A Machine Learning Approach

George D. O'Mahony
Dept. of Electrical and Electronic Engineering, University College Cork
Cork, Ireland
george.omahony@umail.ucc.ie

Philip J. Harris
United Technologies Research Center Ireland (UTRC-I)
Cork, Ireland
harrispj@utrc.utc.com

Colin C. Murphy
Dept. of Electrical and Electronic Engineering, University College Cork
Cork, Ireland
colinmurphy@ucc.ie

***Abstract*—Wireless Sensor Network (WSN) technology has developed substantially over the past decade or so and now numerous solutions exist across a diverse range of innovative applications. The expanding Internet of Things (IoT) sector is becoming an ever more important aspect of modern technology and a key motivator for improving security and privacy in WSNs. Typically, WSN protocols form an integral part of the overall IoT infrastructure by enabling the sensor to access point communication links. These wireless links inherently encompass security challenges, frequently due to external interference and intrusions. As IoT applications incorporate WSNs in their architecture, the incentive to attack and compromise these WSNs escalates. Often, commercial off the shelf devices and standardized open-access protocols combine to achieve specific WSN deployments. Numerous WSN vulnerabilities exist, whilst attack approaches are abundant and change frequently. Thus, to ensure acceptable performance, safety and privacy in many IoT applications, the adopted WSN must be secure. This paper discusses IoT security and privacy, by evaluating a machine learning approach for interference detection focused entirely on analyzing received In-phase (I) and Quadrature-phase (Q) samples. Significantly, once an intrusion is detected, mitigation strategies can be implemented, thus emphasizing the requirement for interference detection. Random Forest is chosen as the machine learning classifier as it consists of a large number of individual decision trees operating as an ensemble. An intrusion detection system (IDS) is developed based on Matlab simulated ZigBee data as an initial insight into whether a real wireless data approach may be viable.**

***Index Terms*—IEEE802.15.4, IoT, Interference, Intrusion, Machine Learning, Random Forest, Security, WSN and ZigBee.**

I. Introduction

The Internet of Things (IoT) and wireless sensor networks (WSNs) continue to evolve into integral components of modern technology and are becoming integrated into safety-critical applications [1]. These technologies are rapidly changing the way people live and, hence, the number of connected devices in the spectrum is growing exponentially [1]. WSNs and the IoT can enhance individual performances when operated and connected together. However, this inherently creates new challenges in terms of spectral coexistence, privacy, safety and threat identification. As application deployments continue to explore new safety-critical areas, the incentive for attackers to access sensitive data, or cause a denial of service, escalates.

Distinctive IoT applications are vast and include everything from smart appliances to wireless body area networks and health care [1], [2]. WSN protocols, like ZigBee, can form an essential part of the IoT architecture, by implementing the communication link between sensing devices and the IoT gateway. WSN versatility is observed in the diverse range of potentially IoT enabled applications, which include aircraft health monitoring [3], space-based applications [4], unmanned aerial vehicles [5], precision agriculture and smart buildings, amongst others [6]. This maturing field of WSNs and IoT applications results in long-lived deployments where resource-constrained low-power embedded devices are tightly coupled to the environment and must execute received instructions and necessary data transmissions.

Therefore, the security and availability of each communication link and the delivery of authentic and confidential packets are essential for the useful operation of WSN and IoT applications. Security is required to maintain services, provide privacy and safety, keep data confidential and ensure efficient battery usage. However, as many protocols and devices are publicly available, certain security vulnerabilities are identifiable. Additionally, typical edge devices are low-power resource-constrained isolated equipment which have difficulty in executing complex algorithms.

This paper works on improving privacy and safety for IoT applications by developing a machine learning based intrusion detection system (IDS) focused on the received WSN signal in-phase (I) and quadrature-phase (Q) samples. This approach is encompassed in the idea that once an interferer is detected it can be mitigated, thus motivating a security algorithm focused on detection. The probability distribution function (PDF) and statistical analysis of the received samples define a set of distinct features. Monte Carlo Matlab simulations provide the necessary ZigBee transmissions with and without added malicious interference which, in this paper, consists of the matched protocol approach [7]. Thus, this simulated approach analyzes whether the identified features and detection strategy are a viable method for real-world WSN signals.

The remainder of this paper is organized as follows: Section II discusses previous work in the area. Section III briefly describes the chosen signal model and its security aspects. Section IV defines the features used to develop the machine learning model. Section V depicts the Random Forest method and why it was chosen, while section VI specifies the results and section VII concludes this paper.

II. Related Work

Using machine learning techniques for classification and intrusion detection in WSNs is the main focus of this paper and covers a variety of literature. In [8] the throughput, packet drop ratio, and the packet average delay of sensor nodes are used in a Bayesian classification to identify anomalous nodes. Different techniques are compared in their ability to identify WSN outliers in [9]. Machine learning in separate areas of WSNs is discussed in [10], where security and anomaly detection are identified as viable use cases. Using decision trees as an intrusion detection method is provided in [11], where the main advantages include having the highest detection performance, can construct and interpret the model easily and works well with large datasets. Notably, Random Forest was highlighted as outperforming other classifiers in terms of identifying whether data traffic is normal or under attack when using the NSL-KDD data set [12]. These techniques have also previously been shown to detect jamming in global positioning system (GPS) signals [13]. The work in this paper distinguishes itself by focusing on the received I/Q samples and neglecting network-level information.

III. Signal Model & Security

The chosen signal model is the de-facto standard for low-rate wireless personal area networks (LR-WPAN), ZigBee. Almost all available commercial and research sensor nodes are equipped with ZigBee transceiver chips [14] and it is currently deployed in both simple monitoring and critical applications. The standard and its associated devices are essential for the all-inclusive IoT architecture as, typically, ZigBee can be used in IoT applications as the communication link from the sensing platform to an IoT gateway or access point, as visualized in Fig. 1. The operating topology is either star, mesh or peer-to-peer and, in each case, is self-organizing, self-repairing, dynamic and can exploit clustering approaches [15]. ZigBee's physical (PHY) and medium access control (MAC) layers are derived from the IEEE 802.15.4 protocol, which has specifications as per Table I. Here, the ZigBee operating frequency range is the unlicensed industrial, scientific and medical (ISM) 2.4 GHz radio frequency band, in which ZigBee coexists with various other protocols, for example, WiFi and Bluetooth. The 16 relevant channel center frequencies are provided in (1), where F_c is the center frequency and i is the channel number. ZigBee uses carrier sense multiple access with collision avoidance (CSMA-CA) to access the channel and signals are transmitted using direct sequence spread spectrum (DSSS) and offset quadrature phase-shift keying (O-QPSK).

DSSS splits every byte into two 4-bit symbols, where each symbol is spread to a 32-chip (bit) pseudo-noise (PN) sequence from a predefined mapping table, which adds resilience to noise. O-QPSK ensures bit transmissions for the I and Q components occur at different time instants, as the I and Q components are mutually offset by half a chip duration. The signals are pulse shaped to ideally achieve the desirable property of zero inter-symbol-interference at the maximum effect points. This transmission process in the ISM band is visualized in Fig. 2, by employing a Tektronix real-time spectrum analyzer and its associated digital phosphor technology (DPX), which performs hardware digital signal processing and rasterizing of samples into pixel information.

TABLE I
IEEE 802.15.4 (ZigBee) PHY & MAC Specifications

Parameter	2.4 GHz Frequency Band Value	
Number of Channels	16	
Channel Spacing / Width	5 MHz	2 MHz
Channel Range	2.405 → 2.4835 GHz	
Data | Symbol Rate	250 kb/s	62.5 $ksymbols/s$
Byte Spreading	DSSS	
Chip Rate	2 $Mchips/s$	
Modulation	O-QPSK	
Pulse Shaping	Half Sine/Normal Raised Cosine	
Maximum Packet Length	133 bytes	
Channel Access	CSMA/CA & CCA	

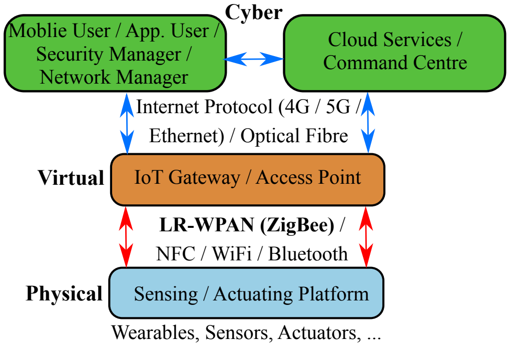
Cyber
Moblie User / App. User / Security Manager / Network Manager
Cloud Services / Command Centre
Internet Protocol (4G / 5G / Ethernet) / Optical Fibre
Virtual
IoT Gateway / Access Point
LR-WPAN (ZigBee) / NFC / WiFi / Bluetooth
Physical
Sensing / Actuating Platform
Wearables, Sensors, Actuators, ...

Fig. 1. IoT Architecture showing the use of ZigBee

$$F_c = 2405 + 5(i-11)MHz,\ for\ i = 11, 12, ...26 \tag{1}$$

WSN security can, generally, be described in terms of requirements, vulnerabilities, attacks and defenses. WSNs need to provide confidentiality, data and origin integrity, services when required, robustness against various impairments and ensure energy is conserved, as most WSN devices contain a finite energy supply. These requirements, typically, ensure the secrecy and authenticity of important transmitted data. Guaranteeing these requirements can be challenging as known WSN security vulnerabilities exist. The open interface of the wireless channel and the public domain character of many WSN protocols enable various forms of attacks. Deployed WSN devices, typically, have low processing power, memory and speed, which all impede the use of conventional security protocols. Regularly, deployed WSN nodes are left unattended in hostile or remote environments, where it is difficult to guarantee continued surveillance and devices can be physically available to potential attackers. The availability of advanced hardware at more affordable rates allows potential intruders to execute diverse attack strategies, which are numerous [6] and can occur across the entire protocol stack. Examples include jamming, spoofing, sinkhole, replaying packets and

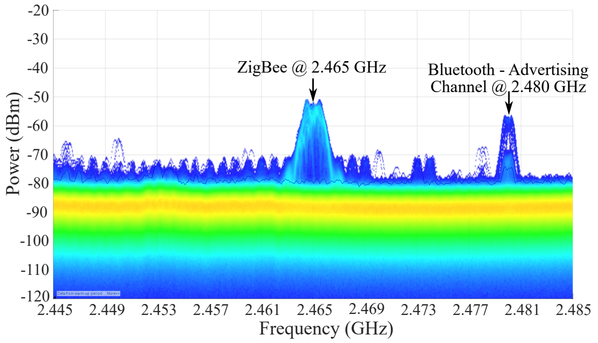

Fig. 2. DPX visualization of a transmitted ZigBee signal in the ISM spectrum

many more. The diverse range of deployments and applications incentivize intruders and typical defenses include cryptography, frame check sequences and DSSS, among others [6]. Here, a defense strategy focused on detection is developed, which utilizes machine learning and a feature set based on the received IQ samples.

IV. Features Explained

Here, the extracted features are briefly described as they are the data analysis foundations on which the subsequent machine learning model is built. The features are based on the received IQ samples and describe ZigBee signals in terms of error-free transmission with no interferer present and erroneous reception caused by an interference signal. In practice, these IQ samples are accessible using a software-defined radio or, possibly, in the device's debug mode if, otherwise, unavailable. The features were developed through Matlab Monte Carlo simulations involving a maximum likelihood decoder where bit errors ≥ 1 led to a packet error. Bit errors were calculated using known transmitted packets, a maximum likelihood decoder and leveraged the correlation equation (2) to identify errors between what was received and transmitted. Each time a packet error occurred, the samples were stored for post-processing. Matched signal interference was the adopted attack as it causes much more damage when compared to conventional continuous-wave jamming and is protocol specific, which causes the spectral image to remain as expected [7]. Each transmission included additive noise, which satisfied a Gaussian distribution, and each interference signal incorporated a random phase offset to aid in resembling real-world transceiver conditions.

$$Corr_{12} = \sum_{n=0}^{N-1} b_1[n] b_2[n] \tag{2}$$

Simulated samples are either analyzed directly or used to produce a PDF and then evaluated. The expected PDF, when interference is low and packets are error-free, is a unimodal shape with a low degree of variance. This changes when interference is injected into the channel as the PDF begins to resemble a bimodal shape with larger variance and becomes more evident as the interference power increases. From the PDF four distinct features are calculated; the area between bins -2 to +2, averaged area of the bins -128 to -3 and +3 to 127, the number of non-zero bins and the maximum peak. Each area is calculated using the Matlab *trapz* function, which is provided in (3), where the spacing is constant, due to PDF construction, $f(x)$ is the PDF function and N is the corresponding number of bins. As interference power increases the maximum peak decreases and the number of non-zero bins increases.

$$\int_a^b f(x)dx \approx \frac{b-a}{2N} \sum_{n=1}^{N} (f(x_n) + f(x_{n+1})) \tag{3}$$

The samples can be analyzed directly as IQ components to develop the remaining features. These signal characteristics include the sample variance/standard deviation, the entropy of the signal, the mean value and the absolute maximum value in the received sample set. As the interference power increases, the signal details shift to a bimodal, high variance construction and so (4) can be used to calculate the sample variance, where μ is the mean and, as a result, (5) calculates the standard deviation. The mean and absolute maximum value in the received samples both increase as the interferer becomes more prominent, while the entropy of the samples (6) decreases as the noise-like error-free signal becomes encompassed by a more dominant interferer.

$$V = \frac{1}{N-1} \sum_{i=1}^{N} |A_i - \mu|^2 \tag{4}$$

$$s = \sqrt{V} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} A_i} \tag{5}$$

$$S = -\sum P_i \log_2 P_i \tag{6}$$

V. Machine Learning Algorithm: Random Forest

In this paper, the detection of interference in the IoT sensor to IoT gateway communication link is defined as a classification problem. This approach suits the overall concept of interference detection as, here, the goal is to determine whether what is observed in the received signal is due to an intruder, or not. A decision tree algorithm seems appropriate as, typically, this concept is near the top of the classifier hierarchy [16]. Decision trees can be explained by focusing on an individual structure in Fig. 3, where each tree is constructed as a series of binary intermediate nodes, each successively choosing the attribute and associated threshold that provides the best split of the sample subset. Here, the input is deconstructed into a feature set that is used by the individual decision trees to split the observed input into groups that are as different from each other as possible, while the members of each group are as similar as possible. In terms of this paper, the groups would be error-free ZigBee signals and received signals with interference causing a range of bit errors. The features outlined in section IV try to define these distinct groups with as much mutual separation as possible.

The Random Forest algorithm [17] is a supervised machine learning approach and is chosen as it consists of a large number of individual decision trees that operate as an ensemble. This simple yet powerful ensemble concept forms the fundamental theory upon which the Random Forrest algorithm depends. The ”wisdom of crowds” approach specifies that the collective consensus of a group of individuals is usually more valuable than that of any singular entity. Thus, this algorithm operates by combining a large collection of relatively uncorrelated models, sub-optimal decision trees, as a committee to produce a composite decision of higher quality that will outperform any of the individual constituent models. Decision-making depends on a diverse group rather than a predominantly homogeneous approach. Essentially, each individual tree is unique and specifies a vote and the output with the most votes is the overall prediction. This is visualized in Fig. 3, where 7 trees predict interference and 2 trees predict a clean signal, therefore the decision is that interference is present.

This idea depends on having low correlation between individual trees, as this protects each tree from their individual error [16]. Uncorrelated decision trees are ensured by two methods: bagging (bootstrap aggregating) and feature randomness. The former exploits each decision tree’s high sensitivity to the training data used and the latter ensures each tree can only pick from a random subset of available features. Random forest allows each individual decision tree to be constructed by choosing a random subset of the training samples. Applying replacement allows examples to be repeated to maintain the sample size N, while, concurrently, allowing for a unique tree to be modeled. Thus, as each sample-set is randomly chosen from the total training sample set, the corresponding decision trees, known as weak-learners, contain different variations of the original classification, which reduces variance and helps to avoid over-fitting. The random sample-set and feature set allows for the creation of uncorrelated trees that protect each other from their own errors and, once a set of decision trees has been computed, a new sample can be classified by performing a majority voting scheme, as visualized in Fig. 3.

Here, Random Forest, specifically decision trees, suit the identified problem for many reasons. This algorithm was used to develop an interference detection scheme in GPS signals [13], it is cited as being suitable for classification and intrusion detection [11], is fast, scalable, robust to noise, does not overfit [18] and, importantly, can work with large datasets. As Monte Carlo experimentation, either through simulations and/or live data, is required for WSN transmission analysis, the chosen algorithm must be capable of working with large example datasets. Multiple iterations (likely in the tens of thousands) are required to sufficiently model the wireless channel, as typical channels and environments change regularly. WSNs are commonly deployed in environments where the spectrum changes rapidly due to the number of connected devices, demand, packet size or services in operation and the physical channel changes due to varying fading levels, obstacles, path losses, and spurious interference. Furthermore, employing machine learning techniques on low power embedded systems by exploiting low-power micro-controllers is becoming more achievable in IoT applications [2], [19], meaning optimizing machine learning algorithms for WSN nodes is possible. Hence, developing this type of algorithm for use in a WSN is an achievable task and is becoming more relevant as training begins to shift from the data centers to the edge nodes.

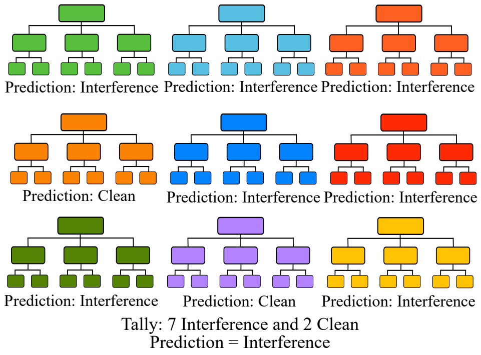

Fig. 3. Visualization of a Random Forest Model making a Prediction

VI. Results

Here, Matlab is used as the development tool, which utilizes the necessary machine learning libraries, to design the IoT WSN interference detection algorithm. Specifically, the Breiman and Cutler Random Forest method [17] has been selected as the classification approach for the advantages described in section V and for previous work in detecting interference is DSSS real-world GPS signals [13]. The Matlab ’TreeBagger’ class is used with all necessary settings to implement the Breiman and Cutler method. Fig. 4 provides the basic approach of the designed algorithm, where a Matlab simulated ZigBee signal, with(out) added interference, is received, deconstructed based on the defined feature set and classified by the designed procedure. To achieve this IDS, data is required to train, evaluate and test the algorithm. Monte Carlo simulations, utilizing ZigBee transmissions, additive noise and interference, supplied the data. Training data is stored in a NxD matrix and the associated annotation vector is of size Nx1, where N is the number of training examples and D is the number of dedicated features for each example. Clearly, each column contains a specific feature and each row corresponds to a data-point example. Hence, each row needs to be annotated as either ’error-free’ or ’erroneous’, where errors are induced by applying a matched protocol attack [7].

The number of features, as per section IV, is nine, which may need optimization and/or expansion at a later stage, and the training data size is 70% of all available data points. Presently, some features may be related but, as each tree takes a random subset of features, the data is simulated and this is the first iteration of the algorithm, this fact is kept for a later optimization and analysis stage. For error-free data, 20,000 simulations were executed and deconstructed into the specific features, thus, providing 14,000 (70%) examples for training. Erroneous data includes jamming-to-signal-ratios (JSR) decreasing in steps of 1 dB from 40 dB to -15 dB. As the probability of error (P_e) increases with JSR, the number of simulations executed rises as the JSR decreases. Once a packet error occurs, the data is deconstructed and logged.

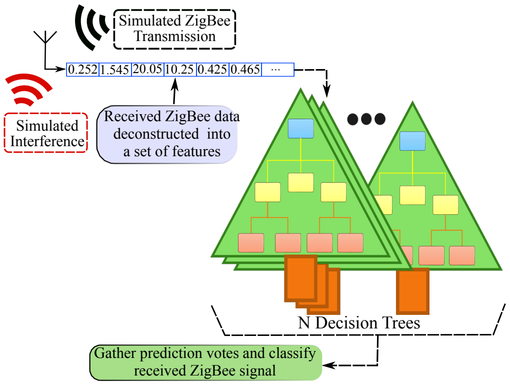
Simulated ZigBee Transmission
0.252
1.545
20.05
10.25
0.425
0.465
···
Simulated Interference
Received ZigBee data deconstructed into a set of features
N Decision Trees
Gather prediction votes and classify received ZigBee signal

Fig. 4. Data flow diagram representing how the ZigBee data is collected, deconstructed and classified by the Random Forest algorithm

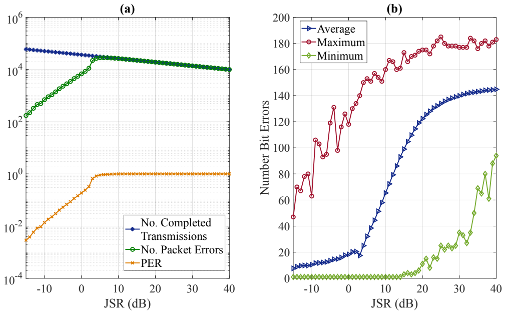

Fig. 5. The logarithmic approach to data collection and associated BER

Thus, 70% of the packet error data at each available data point is used for training. The number of executed simulations is in a logarithmic scale from 10,000 at 40dB JSR to 60,000 at -15dB JSR and visualized in Fig. 5 (a), where the performed number of trials and resulting packet errors are provided. A successful attack produces bit errors in a packet, as it results in either rejection or non-reception and requires retransmission. Here, a constant packet length of 40 bytes is used with an attack packet of matching length for a tight comparison across the JSR range. The designed algorithm attempts to identify why packets contain errors by simply taking a snapshot of the received IQ samples. However, as it is based on simulated results, the feature set contains differences from error-free data even at a JSR value of -15 dB. This concept is visualized by analyzing the out-of-bag (OOB) error, which is a method of measuring the prediction error of a decision tree algorithm, utilizing bagging to sub-sample data samples used for training. OOB is the mean prediction error on each training sample x_i, using only trees that did not have x_i in their bootstrap sample.

The OOB is provided in Fig. 6 for four cases including a two-class case for error-free and erroneous, an extended three-class case to separate the erroneous stage into PER regions of ≥ 0.32 and ≤ 0.32 and an erroneous case above and below a JSR of 5dB, which were identifiable during feature extraction. The PER regions relate to the decreasing slope towards low levels of PERs in Fig. 5, relating to, typically, unintentional interference. Finally, a four-class case is presented based on the packet/bit errors in Fig. 5. A PER of $\leq 10\%$ and bit errors ≤ 15 defines a region where unintentional interference or high channel noise may exist, a PER from $11\% \rightarrow 32\%$ and bit errors from $15 \rightarrow 20$ defines a subtle jamming or signal collision region and above these resides a high impact jamming region. Fig. 6 specifies that the OOB decreases with the number of trees and this OOB is much smaller for the two-class case. However, having such small differences between 'good' and 'bad' signals is, typically, not the best approach to ensure low false positives and high true positives. Also, as bit errors are sporadic when interference is supplied, visualized in Fig. 5 (b), being able to identify different zones is advantageous. Therefore, the algorithm's ability to define multiple cases is beneficial, as the high and medium jamming regions have a higher separation from error-free signals.

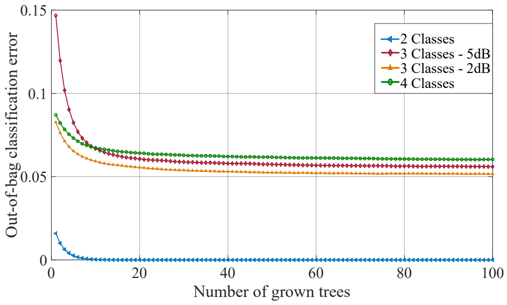

Fig. 6. Designed Random Forest Algorithm: Out of bag errors

TABLE II
DESIGNED RF ALGORITHM: SPECIFICATIONS

	Predictor Depth	No. Trees	Train Time	Prediction Time
Validation Set	5	55	41.44 s	191 ms
Training Set	5	55	169.205 s	420 ms

The four-class case was validated using available validation and testing data to determine the optimal metrics, including the number of decision trees, feature depth and minimum percentage error. Validation data contained 20% of all available data and included varying the maximum feature depth from one to nine and the number of decision trees from one to one hundred and thirty-nine. These results are supplied in Fig. 7, which specifies the lowest error level for the four-class case at $\approx 5.87\%$ using fifty-five decision trees and a maximum feature depth of five. The training time and average prediction time are supplied in Fig. 8 and Fig. 9, respectively, while Table II supplies the final algorithm metrics. These results highlight an optimization prospect in terms of prediction time and provide motivation for a real-world signal approach.

VII. CONCLUSION

This paper used simulated ZigBee transmissions to demonstrate the suitability of using received IQ samples and a machine learning algorithm for malicious interference detection in WSNs for IoT applications. The concept was designed using a Random Forest approach and malicious matched protocol interference. The procedure is built upon deconstructing received signals into a set of features entirely based on the received I/Q

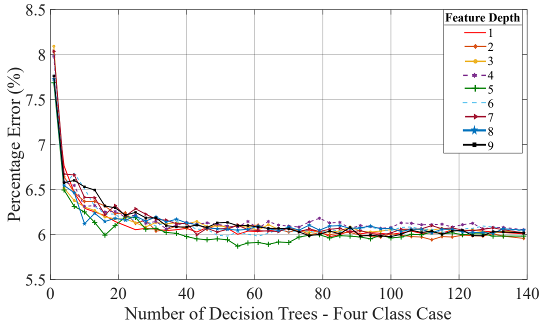

Fig. 7. Designed Four-Class Case - Validation: Error

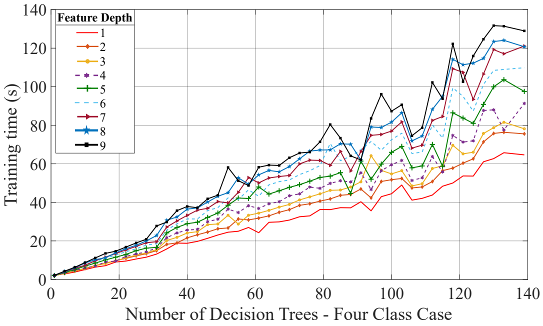

Fig. 8. Designed Four-Class Case - Validation: Training Time

samples. This work on Monte Carlo Matlab simulated ZigBee transmissions provided substantial evidence that the developed feature set and machine learning approach could be adapted to real-world wireless signals. The classifier allocates observed (received) signals into groups, either error-free, erroneous or a sub-set of erroneous. Thus, in future work, as attack strategies change, the potential exists that the number of observed groups will expand to provide both interference detection and attack classification. Live real-world signals are the next stage of expansion as this approach needs to be tested and trained using real-world live WSN signals in various environments and channels. As machine learning has previous applications on low-power embedded devices, this approach is a viable and achievable security enhancement. Once this algorithm is adapted to real-world signals and various attack approaches, the overall approach requires optimization for use on low-power WSN edge devices. In addition, the implementation results and trade-offs for using this approach on resource-constrained devices need to be determined and analyzed.

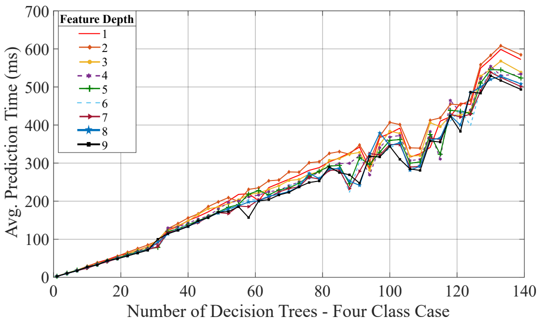

Fig. 9. Designed Four-Class Case - Validation: Average Prediction Time

Acknowledgment

This work has been jointly funded by the Irish Research Council (IRC) and United Technologies Research Center Ireland (UTRC-I) under the post-graduate Enterprise Partnership Scheme 2016, award number EPSPG/2016/66. Authors would like to thank both funding bodies for their continued support.

References

[1] F. Wu, T. Wu, and M. R. Yuce, “Design and Implementation of a Wearable Sensor Network System for IoT-Connected Safety and Health Applications,” *IEEE 5th World Forum Internet Things*, pp. 87–90, 2019.

[2] V. M. Suresh, R. Sidhu, P. Karkare, A. Patil, Z. Lei, and A. Basu, “Powering the IoT through embedded machine learning and LoRa,” in *IEEE World Forum Internet Things, WF-IoT 2018*. IEEE, 2018, pp. 349–354.

[3] R. K. Yedavalli and R. K. Belapurkar, “Application of wireless sensor networks to aircraft control and health management systems,” *J. Control Theory Appl.*, vol. 9, no. 1, pp. 28–33, 2011.

[4] T. Vladimirova, C. P. Bridges, J. R. Paul, S. A. Malik, and M. N. Sweeting, “Space-based wireless sensor networks: Design issues,” *IEEE Aerosp. Conf.*, pp. 1–14, 2010.

[5] C. Dragana, G. Stamatescu, L. Ichim, and D. Popescu, “Interlinking Unmanned Aerial Vehicles with Wireless Sensor Networks for Improved Large Area Monitoring,” in *Int. Conf. Control. Decis. Inf. Technol.*, 2017, pp. 359–364.

[6] G. D. O Mahony, P. J. Harris, and C. C. Murphy, “Investigating the Prevalent Security Techniques in Wireless Sensor Network Protocols,” in *30th IEEE Irish Signals Syst. Conf.*, 2019, pp. 1–6.

[7] ——, “Analyzing the Vulnerability of Wireless Sensor Networks to a Malicious Matched Protocol Attack,” in *52nd IEEE Int. Carnahan Conf. Secur. Technol.*, 2018.

[8] Z. Xiao, C. Liu, and C. Chen, “An anomaly detection scheme based on machine learning for WSN,” in *1st Int. Conf. Inf. Sci. Eng. ICISE*. IEEE, 2009, pp. 3959–3962.

[9] H. Ayadi, A. Zouinkhi, B. Boussaid, and M. N. Abdelkrim, “A machine learning methods: Outlier detection in WSN,” in *Int. Conf. Sci. Tech. Autom. Control Comput. Eng.* IEEE, 2015, pp. 722–727.

[10] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, “Machine learning in wireless sensor networks: Algorithms, strategies, and applications,” *IEEE Commun. Surv. Tutorials*, vol. 16, no. 4, pp. 1996–2018, 2014.

[11] K. A. Jalil, M. H. Kamarudin, and M. N. Masrek, “Comparison of machine learning algorithms performance in detecting network intrusion,” in *Int. Conf. Netw. Inf. Technol.*, 2010, pp. 221–226.

[12] M. C. Belavagi and B. Muniyal, “Performance Evaluation of Supervised Machine Learning Algorithms for Intrusion Detection,” *Procedia Comput. Sci.*, vol. 89, pp. 117–123, 2016.

[13] G. D. O’Mahony, S. O’Mahony, J. T. Curran, and C. C. Murphy, “Developing a low-cost platform for GNSS interference detection,” in *Eur. Navig. Conf.*, 2015, pp. 1–8.

[14] B. Stelte and G. D. Rodosek, “Thwarting attacks on ZigBee - Removal of the KillerBee stinger,” in *Proc. 9th Int. Conf. Netw. Serv. Manag.*, 2013, pp. 219–226.

[15] I. Tomi and J. A. McCann, “A Survey of Potential Security Issues in Existing Wireless Sensor Network Protocols,” *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1910–1923, 2017.

[16] T. Yiu, “Understanding Random Forest.” [Online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[17] L. Breiman, “Random Forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[18] A. Singh, N. Thakur, and A. Sharma, “A review of supervised machine learning algorithms,” in *Int. Conf. Comput. Sustain. Glob. Dev.*, 2016, pp. 1310–1315.

[19] C. Leech, Y. P. Raykov, E. Ozer, and G. V. Merrett, “Real-time room occupancy estimation with Bayesian machine learning using a single PIR sensor and microcontroller,” in *IEEE Sensors Appl. Symp.* IEEE, 2017.