RF Fingerprinting of Software Defined Radios Using Ensemble Learning Models

Arun Kumar K A
Centre for Development of Advanced Computing, Trivandrum, India
Email: arunkumarka@cdac.in

Abstract —Machine Learning (ML) is becoming a transformative technology in wireless communication. The deployment of large scale RF devices particularly in IoT applications escalates security threats and also setting up of secure networks using wireless devices is becoming a big challenge. Along with ensuring security, identifying each RF device in an autonomous network is essential and the RFML (Radio Frequency Machine Learning) can play a crucial role here. This paper focuses on the RF characterization of a set of Software Defined Radios (SDR) using advanced machine learning models. This helps to identify each SDR module in the deployed network which runs only a specific protocol in a particular network. The SDRs will be configured for a particular specification and the test will be conducted. The transmitted data from multiple radio nodes were collected using a reconfigurable radio's receive chain in IQ-format, in the laboratory environment. The RF features like IQ-imbalance, DC-offset and the image leakages in the multicarrier modes were used to set fingerprints for identifying the reconfigurable radios. Two ensemble learning models Random Forest and AdaBoost were used to train and develop predictive models to identify the radio. At a SNR of 30dB Random Forest achieved an accuracy of 85% and AdaBoost achieved an accuracy of 78% with 32K multicarrier data. A maximum recognition rate of 92% is achieved with RF and 83% with AdaBoost.

Index Terms—Machine Learning (ML), RF fingerprinting, software defined radio, SDR, RFML, PYTHON, random forest, and AdaBoost.

I. Introduction

Each and every wireless device has a significant RF characteristic in their transmitted electromagnetic wave which can be used to identify a particular device. RF fingerprinting or wireless fingerprinting is a technique used to classify and identify RF devices from a set of devices. The RF characteristics of wireless devices will vary even though they are designed using same components and from the same vendor. The imperfections and impairments in power amplifiers, mixers, PLL and filters will significantly affect the normal behavior of a wireless device even though they run using the same protocol. Modern ML algorithms can be utilized to exploit these behavior of the RF devices to generate separate fingerprints and we can mark the radios with unique identity.ML based identification of RF devices also play a critical role in ensuring security in wireless networks. The goal of this work is to develop a prediction model which will identify each SDR module from a pool SDRs in a network. The SDRs will be configured to run a single carrier and a two carrier narrowband waveform in laboratory environment and data will be captured. The RF characteristics of the SDRs are learned using its behavior in single carrier and multicarrier environment. Data captured from around ten SDR transmitter modules were used in the training processes.

Two ensemble learning models will be used and two prediction models will be developed and a performance study of these two will be done in detail. Ensemble learning models works by combining the decisions from multiple models. In this work we use Random forest, a Bagging algorithm and AdaBoost which is a Boosting algorithm. The two models will be applied on the transmitted data captured using a wired environment.

II. Datasets Generation and pre-Processing

A. RF Imperfections as a Fingerprint

The RF hardware imperfections will lead to unpredictable DC-offset, IQ-imbalance and image rejection for each RF devices even though they are from the same make. The reasons for DC-offsets are improper RF-Mixing, LO leakage and non-linearity of the components used.

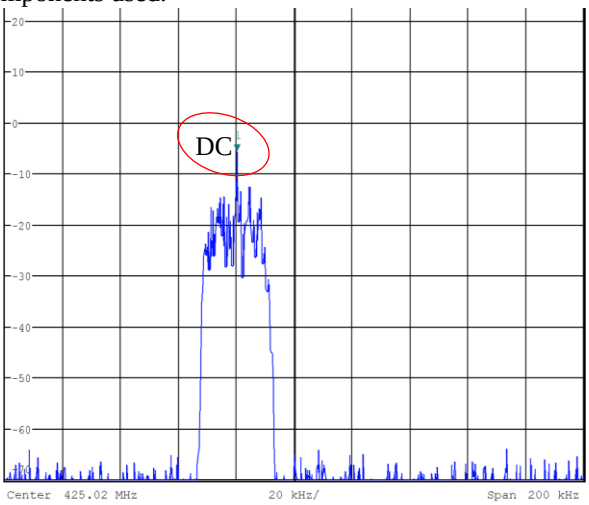

Fig. 1. DC-Offset

Even after implementing DC-cancellation modules in the transmitter there will be DC components in minimum

Manuscript received October 2, 2021; revised March 12, 2022.
doi:10.12720/jcm.17.4.287-293

level which will not be identical in all the RF-devices even though similar components and same DC-null and phase corrections algorithms were used. [1], [2]. Fig. 1 shows the DC-offset in a narrowband DQPSK waveform captured at the transmitter output.IQ based RF-designs usually use two separate paths, one path for I-signals and other for Q-signals. Performance of the RF-mixers, PLL, gain and phase mismatches in the IQ parallel path will lead to IQ Imbalances. Total IQ imbalance is a combination of offset in phase and amplitude in I and Q path.IQ imbalance cancellation modules will help to suppress the imbalance to a larger extent but still it will not be zero practically. Similar behavior of the RF-devices will result in generation of image components also. All these characteristics can be used to set a fingerprint to RF-devices which can be used to identify each device [3]. Fig. 2 shows the IQ-Constellation of a normal waveform and a waveform with IQ-imbalance.

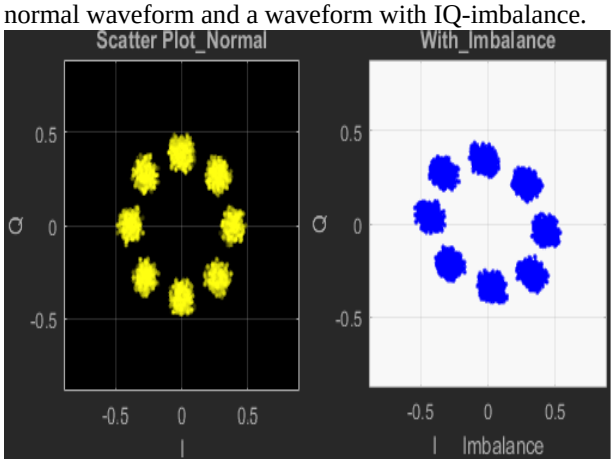

Fig. 2. IQ-Imbalance

B. Data Generation in the Laboratory Environment

For controlled study the data needed for training and testing were captured using a laboratory setup.

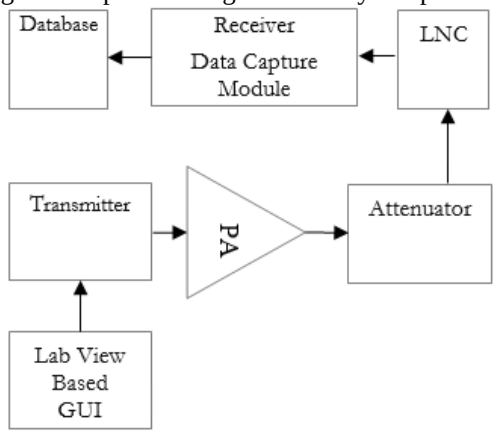

Fig. 3. Data capture module

Fig. 3 shows the block diagram representation of the hardware setup used for data capturing. The transmitter module is a reconfigurable module designed using FPGA so we can integrate multiple protocols and can use for data capturing. For this work we used a narrowband single carrier and a two carrier DQPSK waveform.

Two sets of data were used to learn each transmitter and in total around 10-radio sets were used for the work. This test bed also supports a software based RF-impairments correction using a GUI designed using Lab View and using this impairments compensation values can be inserted in the transmit path. The transmitters are designed to work at 390MHz to 425 MHz and at the data capture end the received data in down converted to an IF of 70MHz and using a Digital Down converter the signal is mapped to baseband. We collected IQ data with a sample size of 32K in which I data is of size 16K and Q is of size 16K.For each transmitter around 32K data set is captured in which 16K is a single carrier signal and 16K is a two-carrier signal. Finally the data looks like 32Kx32K matrix for a single transmitter with each sample size is of 16-bit and total data used for modelling is around 32GB [4], [5]. The block diagram representation of the signal generation and reception is shown in Fig. 4.

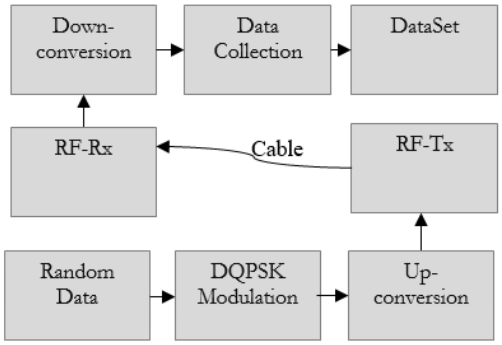

Fig. 4. Wired data generation and preparation

SNRs ranging from -10dB to around 30dB and the performance of the developed model will be studied by using data with low range SNR and high range SNR particularly the accuracy of the model is analyzed [6].

Classifier Architecture description

Ensemble learning models have the capacity to improve overall performance by combining the decisions from multiple models. Ensemble learning algorithms are basically classified into the bagging method, stacking method, boosting method and blending. This work focuses on boosting and bagging technique. Bagging, Bootstrap Aggregation helps to develop prediction models with less variance using decision tree method. Boosting algorithms helps to develop strong predictive models which works by boosting the weights of the observation based on the results of the previous observations.

Fig. 5 shows the advanced ensemble learning algorithms. This paper focus on applying Random Forest and Ada Boost algorithms on the RF data to characterize the transmitter. Random forest works by selecting feature sets randomly and using them at the decision tree nodes [6]. Random forest is a supervised learning technique which sets up the forest using multiple random binary trees which performs good with noisy data [7], [8]. Ada boost works by creating multiple sequential models and

each model will correct the errors from the previous models. Ada boost can improve the accuracy of prediction by combining the predictions of other weaker learners with inaccurate rules.

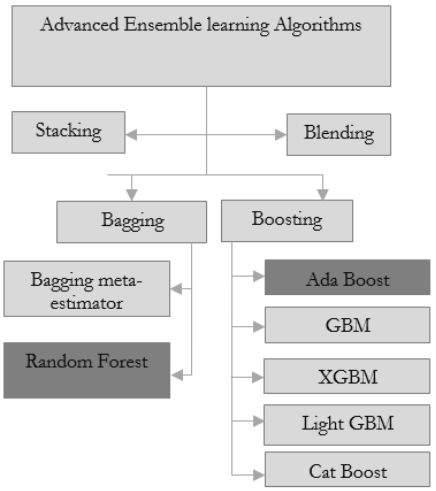

Fig. 5. Ensemble learning algorithms

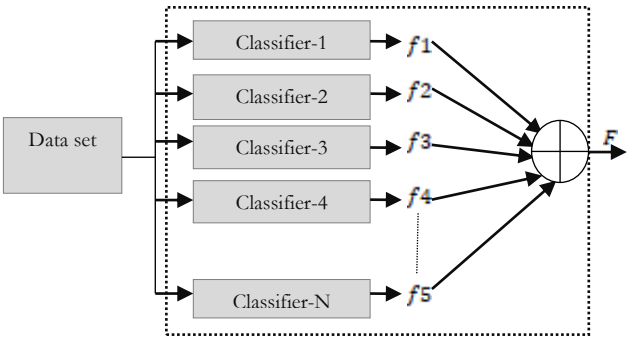

Fig. 6. AdaBoost classifier

The basic theory of Ada boost says that there will be an exponential drop in the training error of final hypothesis to zero if the accuracy of the weak classifier is only slightly better than half [9], [10]. As per the literature survey it is learned that when compared with CNN and RNN, Random Forest and Ada Boost are generally not widely used for RFML. Fig. 6 shows the basic structure of an AdaBoost classifier.

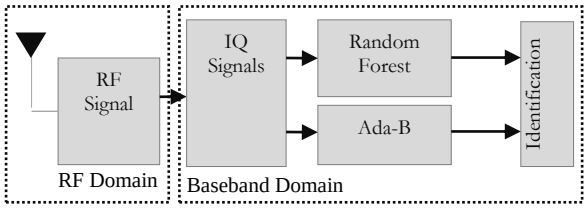

Fig. 7. Basic design flow

The full system implementation involves data collection or preparation and analysis in which the data collection modules are designed using xilinx FPGA and the analysis portion is implemented using PYTHON. Fig. 7 shows the basic block diagram of the work with signal flow indications. Initially the study is performed using two devices with both the algorithms, in which single carrier and two carrier waveforms data were used. The accuracy improvement is also studied by varying the data size. Then ten transmit processing modules were used and the performance is analyzed with multiple data sets at different SNR and data size. The study is categorized into different modes based on number of carriers used, devices used and data size.

TABLE I: TEST MODES

Mode	Device used	Carrier	Data size
Mode-1	2	1	1K,32K
Mode-2	2	2	1K,32K
Mode-3	10	1	1K,32K
Mode-4	10	2	1K,32K

Table I shows the modes of testing with different configurations and this terminology will be used in next discussions. As an initial study only two TPU devices were used and tested with minimum data and then the data size is increased by a specific step to analyze the performance like accuracy and precision. This study is repeated for data with different SNR also. In mode-3 and mode-4 ten devices where used with larger and smaller data sets with different SNR also. Another objective of this work is to study whether learning the transmitters with narrow band multicarrier data have any influence in the overall prediction performance of the system.

III. DATA PREPARATION

Fig. 8 shows the test bed used for data capture and transmission, in which four transmit processing units (TPU) were shown and in total ten TPUs were used. Power amplifier and data capturing modules were also marked in the figure. Before starting transmission the power amplifier is linearized using cartesian loop feedback technique with CMX998. It is ensured that the Phase offset and DC leakage correction values are similar for all the TPUs.

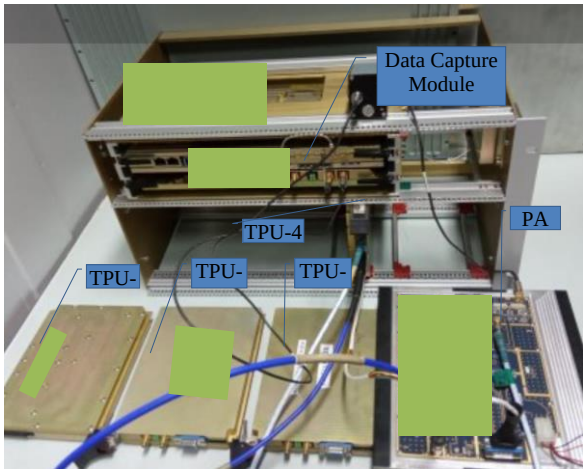

Fig. 8. Test bed

For data capturing each TPUs will be inserted in the allotted slot and connected with the power amplifier and at a time only one TPU will be used even though multiple slots are available. For data capturing the TPUs are

configured to operate in 390MHz-425 MHz band with transmitting power varying from 1W to 40W.

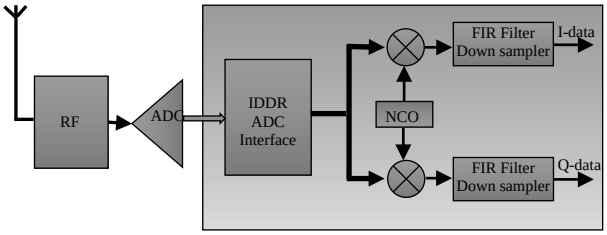

Fig. 9. Data capture Module-FPGA design

Fig. 9 shows the block diagram representation of the data capture module implemented in the FPGAs. The data preparation module includes AD9268 analog to digital converter and the Digital Down Converter (DDC) module. The DDC is designed and implemented in xilinx FPGA using system generator and Matlab. The DDC is designed to receive IF data over wide band from 65 MHz to 75 MHz which is generated by the low noise controller (LNC) module. The DDC is designed using DDS, CIC filter and a set of FIR filters. The data capture module involves a xilinx FPGA with a block RAM memory module implemented in it with required interfaces. A ping-pong buffer architecture is designed with 64K block rams. An automatic data acquisition module is designed using PYTHON and Tcl which will automatically configure FPGA, capture and load the data in empty buffers. The automatic data acquisition unit is basically a VIVADO automation module which automatically configure the FPGA and capture the data in specific intervals and save the data in .csv format with specific labels and file names. The labelling is used in the training and prediction phase. The captured data is stored in the required format and fed to the python designs. Data captured configuration are:

- Frequency: 390,392.5,395, 398,421,420,425 MHz
- Power Amplifier: 30, 34,40,44,46 dBm
- Temperature: 35, 40, 50,65,75,80 ºC

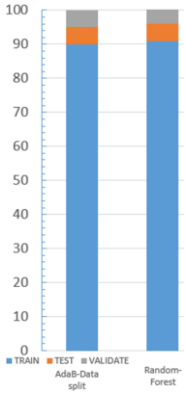

Fig. 10. Data split

The data sets were split into random train and test splits using the sklearn train_test_split API [11], [12]. In the data set around 91% of the data is assigned for training, 5% for validation and 4% testing in case of Random Forest. For AdaBoost 90% data is used for training, 5% for validation and 5% for testing. This split is selected based on best result achieved, other combinations were also designed to understand the performance of the algorithm. Fig. 10 shows the bar plot representation of data distribution for training, test and validation. The performance of the algorithms for different modes were detailed using accuracy curves and confusion matrix plots. In all modes a 25 kHz tetra waveform is used which uses DQPK modulation with a symbol rate of 18kbps.The key module in the transmitter side is a FPGA in which random bit generator module is implemented, so that random IQ samples can be used for training. In mode-1 only two transmit processing modules (TPUs) were used with data size gradually increased from 1k to 32k and each IQ-samples were 16-bit wide.

IV. Results and Observations Discussion

Both Random Forest and AdaBoost classifiers are designed using Python which is an open source package with machine learning libraries. Both the Random Forest classifies and the AdaBoost classifier were modeled using the sklearn API [13]. The crucial parameters which decides the performance of the Random forest model are the number of trees (n_estimators), the criteria to measure the quality of a split (criterion), maximum depth of the tree (max_depth) and the minimum number of samples required to split (min_samples_split) [14]. The Random forest predication model is designed using both the Gini and Entropy criteria to study the performance. Base estimator, number of estimators, learning rare and the boosting algorithm are the critical parameters to be considered while modeling the AdaBoost predictor

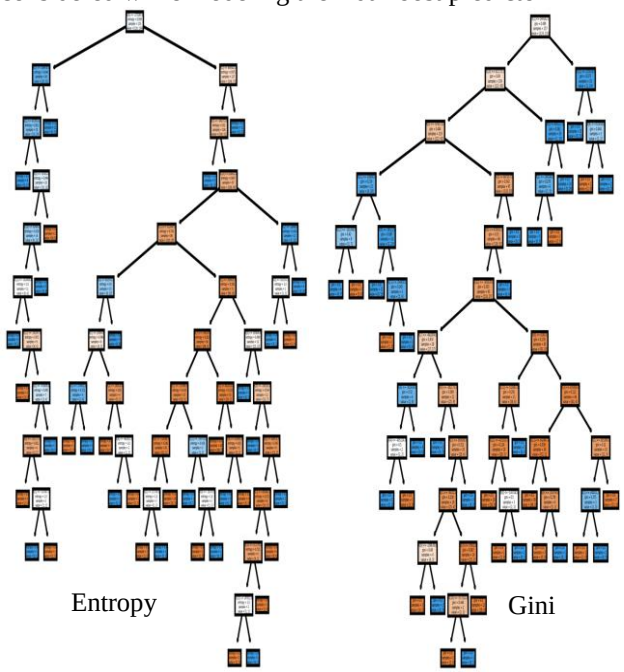

Fig. 11. Decision tree

Fig. 11 shows the decision tree plot of the random forest classifier for Gini criteria and Entropy criteria to indicate the branch flow.

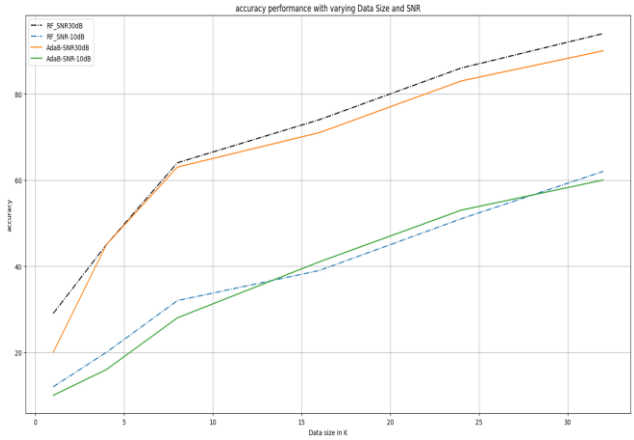

Fig. 12. Accuracy plot for 2-TPUs

Fig. 12 shows the accuracy of the algorithms with two modules while increasing data size form 1K to 32K at a SNR of -10dB to 30dB. Mode 2 also gave significant performance improvement for both algorithms. In both RF and AdaBoost the validation accuracy and training accuracy increased gradually with increase in data size irrespective of SNR, but the maximum accuracy achieved is lower for lower SNR data.

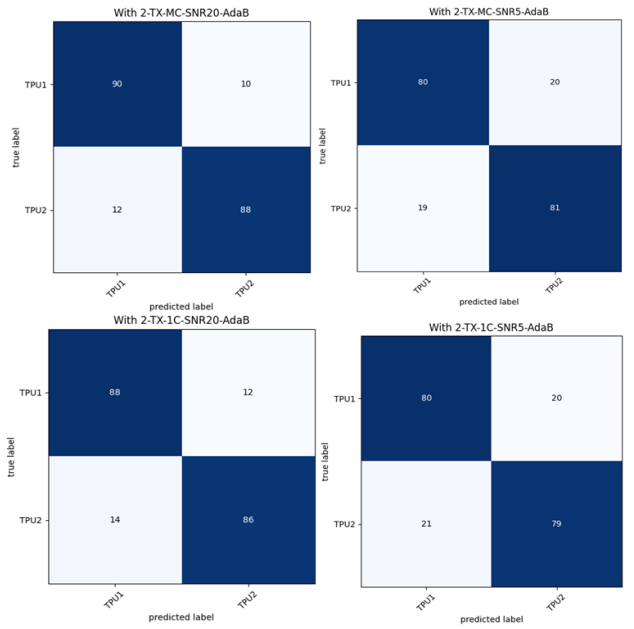

Fig. 13. Confusion matrix plot-AdaBoost mode-1

In two TPU case data with single carrier and multicarrier is also used with data size 32K with 20dB and 5dB SNR and the confusion matrix plot of the same in shows in Fig. 13 and Fig. 14 respectively. When two-carrier data is used with two TPU there is greater increase in the accuracy for both the algorithms. For multicarrier signal with higher SNR there is significant increase in the predication performance when compared with single carrier mode.

As the key objective of this work is to identify each SDR module from set of ten modules the study is conducted by using ten SDR modules with 32K data in single carrier and multicarrier environment with different levels of SNR varying from-10dB to 30dB in 5dB step.

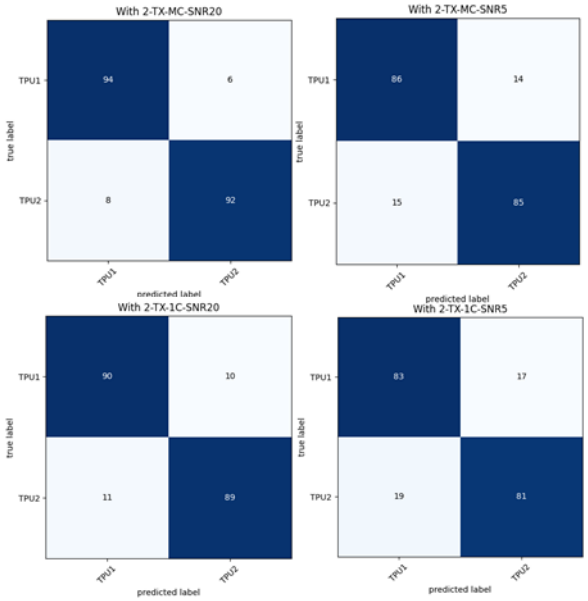

Fig. 14. Confusion matrix plot-RF mode-1

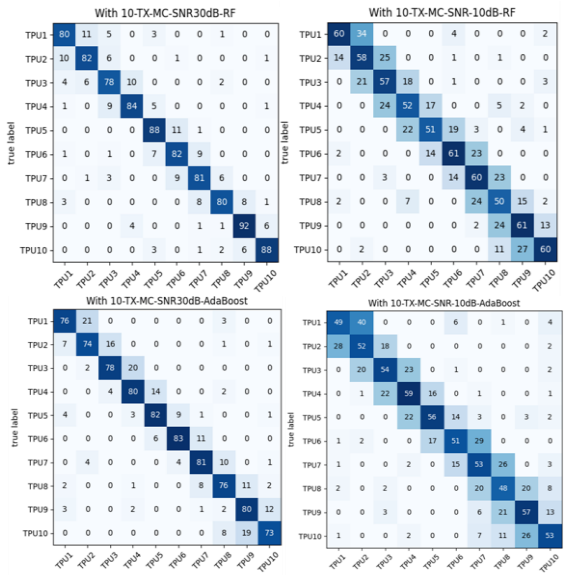

Fig. 15. Confusion matrix plot-RF-AdaBoost mode-4

Fig. 15 shows the confusion matrix plot for random forest and AdaBoost in mode 4 with SNR 30dB and -10dB.So only mode 3 and mode 4 will be discussed in detail with different SNRs. In mode 4 random forest achieved an accuracy of 85% with SNR 30dB, while AdaBoost achieved around 78% accuracy with similar conditions. Similarly in mode 4 both random forest and AdaBoost algorithms achieved an accuracy of 66% and 59% respectively with SNR -10dB. In mode 3 which uses a single carrier narrow band waveform there is a reduction in performance with same SNR and data size for both algorithms, when compared with mode 4 which uses a multicarrier waveform.

It is observed that when multicarrier data is used there is an improvement in the accuracy for both models in

which data captured from ten TPUs were used. Fig. 16 shows the accuracy performance plot of Random forest and AdaBoost for different SNR values and data size. In the figure dotted lines shows the performance of the Random forest model and the solid line represents the performance of AdaBoost. One interesting observation is that random forest achieved more accuracy than AdaBoost even though the data size is low with higher SNR. The height recognition rate for RF is 92% at 30 dB SNR for TPU9 and the lowest is 78% for TPU3.Similarly the lowest and height recognition rate for RF reduced to 51% and 61 % at -10dB SNR respectively. The height recognition rate for AdaBoost is 83% at 30 dB SNR for TPU6 and the lowest is 73% for TPU10.Similarly the lowest and height recognition rate for AdaBoost reduced to 48% and 59 % at -10dB SNR respectively. In both the algorithms out of ten devices no device is fully recognized correctly even at 30dB SNR. Table II shows the performance table of Random forest for two device predication model at 20dB and 5dB SNR. For multicarrier samples with higher SNR both accuracy and precision is more than 90% which is lesser for single carrier samples.

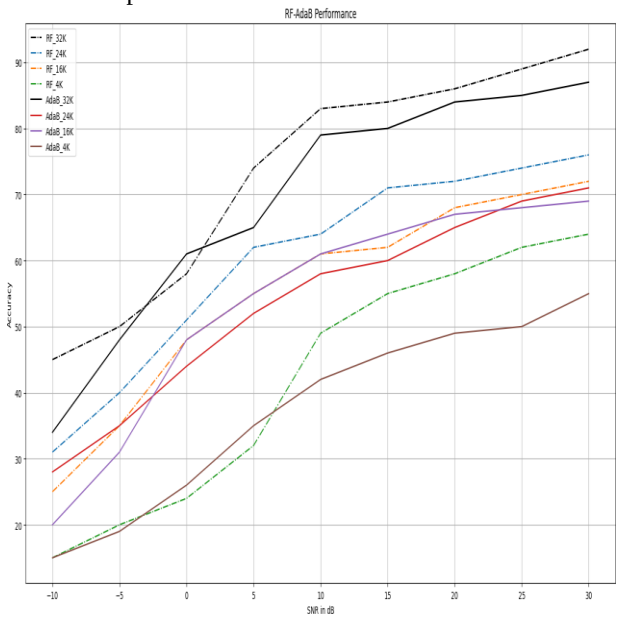

Fig. 16. Accuracy performance-RF and AdaBoost

TABLE II: PERFORMANCE TABLE-RF-2DEVICE-1C/MC

Parameters	1C=20/5dB (%)	MC=20/5dB (%)
Accuracy	90/83	93/85
Precision	89/81	92/86

Table III shows the performance table of AdaBoost for two device predication model at 20dB and 5dB SNR. Here also the performance improved for multicarrier data but less when compared with Random Forest.

TABLE III: PERFORMANCE TABLE-ADB-2DEVICE-1C/MC

Parameters	1C=20/5dB (%)	MC=20/5dB (%)
Accuracy	88/80	90/80
Precision	86/79	88/79

Precision = TP/ (TP+FP)
Accuracy = TP+TN/ (TP+FP+TN+FN)

TABLE IV: PERFORMANCE TABLE-RF/ADB-10DEVICE

Parameters	RF=30/-10dB	AdB=30/-10dB (%)
Accuracy	85/66	78/59
Precision	83/59	77/53

Overall performance of both algorithms reduced drastically when ten devices were used. Table IV shows the performance table of Random Forest and AdaBoost for ten device predication model at 30dB and -10dB SNR.

V. CONCLUSION

The main objective of the paper is to develop a prediction model which can identify and detect a particular SDR module from a set SDRs working as a single network. The SDRs were designed to form a network using a particular protocol and each SDRs were supposed to manage and control eight to ten hand held radios and sometimes need to manage group calls with more than 25 radios. Around 32GB of data is captured from these ten radios with different SNR values and the performance of the Random forest and AdaBoost algorithms were studied. Random forest achieved a best accuracy of 85% when compared with AdaBoost. The novelty in this work is only custom made modules were used for data capturing and validation, and also this paper focused on ensemble learning models like RF and AdaBoost which are rarely used in RFML domain. The next phase study will be conducted by including more radios and capturing more data in wireless mode.

REFERENCES

[1] L. J. Wong, W. H. Clark, B. Flowers, R. M. Buehrer, A. J. Michaels, and W. C. Headley, "The rfml ecosystem: A look at the unique challenges of applying deep learning to radio frequency applications," arXiv preprint arXiv:2010.00432, 2020.

[2] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. R. Chowdhury, "ORACLE: Optimized radio classification through convolutional neural networks," in *Proc. IEEE INFOCOM*, 2019, pp. 370–378.

[3] K. Sankhe, "No radio left behind: Radio fingerprinting through deep learning of physical-layer hardware impairments," *IEEE Trans. Cognitive Commun. Netw.*, vol. 6, no. 1, pp. 165–178, Mar. 2020.

[4] D. Roy, T. Mukherjee, M. Chatterjee, and E. Pasiliao, "RF Transmitter Fingerprinting Exploiting Spatio-Temporal Properties in Raw Signal Data," in *Proc. 18th IEEE International Conference on Machine Learning And Applications*, 2019, pp. 89-96.

[5] N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More is better: Data augmentation for

channel-resilient RF fingerprinting," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 66-72, October 2020.
[6] T. Jian, *et al*., "Deep Learning for RF fingerprinting: A massive experimental study," *IEEE Internet of Things Magazine*, vol. 3, no. 1, pp. 50-57, March 2020.
[7] L. Breiman and L. Machine. (2001). Random Forests. [Online]. 45. pp. 5–32 Available: https://doi.org/10.1023/A:1010933404324
[8] S. Imtiaz, G. Koudouridis, H. Ghauch, *et al*., "Random forests for resource allocation in 5G cloud radio access networks based on position information," *J Wireless Com Network*, 2018.
[9] A. H. Wahla, L. Chen, Y. Wang, R. Chen, and F. Wu, "Automatic wireless signal classification in multimedia internet of things: An adaptive boosting enabled approach," *IEEE Access*, vol. 7, pp. 160334-160344, 2019.
[10] X. Liu, G. Chuai, and W. Gao, *et al*., "GA-AdaBoostSVM classifier empowered wireless network diagnosis," *J Wireless Com Network*, 2018.
[11] F. Chollet, *et al*. Keras: The Python Deep Learning library. [Online]. Available: https://keras.io
[12] M. Abadi, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, 2016.
[13] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al*., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[14] A. B. Shaik and S. Srinivasan, "A brief survey on random forest ensembles in classification model," in *Proc. International Conference on Innovative Computing and Communications*, 2019, pp. 253-260.

Arun Kumar K A was born in Kerala, India, in 1985. He received his B.Tech degree from the University of Kerala, in 2007 and M.Tech degree from Cochin University of Science and Technology in 2009.He is currently working as Principal Engineer in Centre for Development of Advanced Computing (CDAC). His area of interests include signal processing, wireless communication, FPGA and Machine Learning.