Tangled Program Graph for Radio-Frequency Fingerprint Identification

Alice Chillet, Baptiste Boyer, Robin Gerzaguet, Karol Desnos, Matthieu Gautier

Tangled Program Graph for Radio-Frequency Fingerprint Identification

Alice CHILLET†, Baptiste BOYER†, Robin GERZAGUET†, Karol DESNOS*, Matthieu GAUTIER†.
† Univ Rennes, CNRS, IRISA, firstname.name@irisa.fr
* Univ Rennes, INSA Rennes, CNRS, IETR - UMR 6164, Rennes, France, firstname.name@insa-rennes.fr

***Abstract*—This paper proposes to use Tangled Program Graph (TPG) for Radio Frequency Fingerprint (RFF) identification. RFF is a unique signature created by electromagnetic distortions of the different radio frequency hardware components in the device. This signature is contained in the signal and may be used as a secure identifier because it can not be easily spoofed. In recent years, RFF identification is mainly based on Deep Learning (DL). TPG is a new machine learning technique based on genetic evolution, which are less complex than DL. In this paper, we propose to use TPG-based classification to achieve a lightweight and accurate RFF identification scheme. Results show that TPGs achieve the same accuracy as a state-of-the-art convolutional neural network with a learning phase duration clearly reduced on the CPU. TPGs are also used to analyse both the impact of the channel and the receiver radio on the accuracy.**

***Index Terms*—Tangled Program Graph, Deep Learning, Radio Frequency Fingerprint, Software Defined Radio.**

I. Introduction

Since the introduction of the Internet of Things (IoT), low power devices have been used in sensitive and secure networks. Hence, the security problem has been the topic of many researches [1]. As a consequence, it is mandatory to ensure robust and lightweight authentication systems.

In most telecommunication standards, identification methods are based on the meta-data of communication protocol that gives an address to enable the authentication. In fact, such meta-data are used for RFID and WiFi authentication, but those solutions could be counterfeited [2]. Moreover, they should also lead to a low transmission overhead especially for IoT scenarios. In recent years, Jagannath et al. propose to identify static devices by their location [3]. Location methods have strong limitations because they are intrinsically sensible to environmental variations and identification accuracy falls in a dynamic context. A secure identification solution should be robust to time and environment changes, especially in a wireless context.

The Radio Frequency Fingerprint (RFF) identification is a solution based on the singularity of the hardware. The hardware components create unique electromagnetic distortions in the transmitted signal [4]. To identify the RFF, the signal should be captured by a radio and be classified among the different potential candidates. A Software Defined Radio (SDR) receiver helps for signal capture as they are able to record large bandwidths and store the raw In Phase - Quadrature (IQ) samples before applying specific post-processing to help signals classification.

978-1-6654-6483-3/23/31.00 ©2023 IEEE

To identify a device with its RFF, two main methods exist: parametric methods, and learning based methods.

- The parametric methods are done in two steps: first, some features that describe the RFF are extracted from the received signal [4]–[6]. In the second step, a classifier such as K-Nearest Neighbors is used to identify the device based on the values of the features. For example, the feature used in [5], [6], is Power Spectral Density (PSD) while [4] uses many metrics to characterise the RFF such as phase and frequency error.
- The supervised Deep Learning (DL) techniques used training phase to learn how to classify radios. In particular Convolutional Neural Networks (CNNs) are used to extract and classify RFFs [7]–[10].

Optionally, the input signal can be pre-process before being fed into the neural network with, for example, channel equalization or domain transforms. Sankhe et al. explore different architectures of CNN, such as AlexNet-inspired, or CNN with less layers, but more parameters [9], [10]. Some Recurrent Neural Network (RNN) has been explored too [7]. In addition, other architectures of DL have been explored such as transformers architecture [11].

If DL techniques are promising and show very good classification results, they also exhibit an important complexity both at training and inference steps, dependent on their architecture [12]. In the IoT context, RFF authentication might be with stringent complexity and energy constraints.

To address the complexity issues, this paper proposes to use Tangled Program Graph (TPG) instead of CNN. TPGs are a recent light-by-construction machine learning technique based on genetic programming principles [13]. Previous works demonstrated that for comparable performance with a State of the Art (SotA) of DL, TPGs inference required 2 to 3 orders of

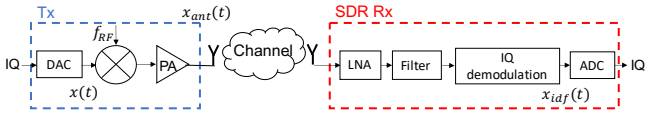

Fig. 1. Transmission and reception chain. LNA stands for Low Noise Amplifier and ADC stands for Analog to Digital Converter.

magnitude less computations, and 3 to 5 orders of magnitude less memory [14].

The core contributions of this paper are as followed:

- We explore a new machine learning mechanism called TPG for RFF identification.
- We propose some results and comparison between TPG and an SotA-inspired CNN in both accuracy and the duration of the learning phase [10].
- We use and analyse the results obtained with a recent database called WiSig [15] to highlight the impact of the propagation channel and the receiver RFF.

The rest of this paper is organized as follows.

In section II, transmission and RFF models are presented, with a focus on their intricacies and pitfall in an RFF identification context and the classification framework is introduced. Section III introduces TPGs and keys on how classification is made with TPGs are given. In Section IV, we describe the database used for our analysis and describes the key performance results of the paper. Section V, draws some conclusions.

II. RFF CLASSIFICATION

A. Transmission chain model and radio frequency features

The RFF of a transmitter is a unique signature created by the hardware components of the transmission chain. A transmission/reception chain is modelled in Fig. 1. The transmission chain is represented in the left part of the channel block. It takes in input a IQ complex symbol stream. The first block represents the digital signal converter, converted in the analog domain to obtain $x(t)$. Then the local oscillator modulates the signal at the carrier frequency f_{RF}, and the Power Amplifier (PA) amplifies it. All those components distort the signal and create the signature called the RFF of the transmitter denoted $\mathcal{F}_{\mathrm{RFF_{Tx}}}$. The emitted signal could be modelled by:

$$x_{\mathrm{ant}}(t) = \mathcal{F}_{\mathrm{RFF_{Tx}}}\left(x(t)\right), \tag{1}$$

$$x_{\mathrm{ant}}(t) = \mathcal{F}_{\mathrm{PA}} \circ \mathcal{F}_{\mathrm{osc}} \circ \mathcal{F}_{\mathrm{IQ_{mismatch}}} \circ \mathcal{F}_{\mathrm{DAC}}\left(x(t)\right), \tag{2}$$

where $\circ$ represents the function composition.

Equation (2) shows the impact of each component. $\mathcal{F}_{\mathrm{DAC}}$ represents the distortions caused by the Digital to Analog Converter (DAC). $\mathcal{F}_{\mathrm{IQ_{mismatch}}}$ models the impairments induced by the IQ modulation and called IQ Mismatch: the two independent paths of the modulator are not perfectly orthogonal, and this breaks the plane orthogonality. $\mathcal{F}_{\mathrm{osc}}$ represents the impact of the oscillator, used to shift the signal from baseband to its carrier frequency. It will lead to two impairments i) a carrier frequency offset ii) a phase noise. Finally, $\mathcal{F}_{\mathrm{PA}}$ corresponds to the distortion induced by the amplifier non linearity. It can be for instance modeled by a parametric model (such as Rapp model) or a polynomial model. These nested functions show the difficulty to extract features. However, some parametric solutions are proposed by [5], [6] which use the Power Spectral Density (PSD) coefficients to characterise the transmitter, or metrics that characterize the RFF [4].

In Fig. 1, the channel block represents the wireless communication environment as interference signals, the multi-path and fading channel models which could impact the signal. Finally, the receive block represents the receiver with its own components (not detailed in the model) and its own distortion functions. It corresponds to a RFF in the receiver. The received IQ signal used for identification and denoted $x_{\mathrm{idf}}(t)$ can be modeled as:

$$x_{idf}(t) = \mathcal{F}_{\mathrm{RFF_{Rx}}} \circ \mathcal{F}_{\mathrm{channel}} \circ \mathcal{F}_{\mathrm{RFF_{Tx}}}\left(x(t)\right), \tag{3}$$

with $\mathcal{F}_{\mathrm{channel}}$ represents the propagation channel, and $\mathcal{F}_{\mathrm{RFF_{Rx}}}$ represents the distortion function caused by the receiver. The impact of the receiver and channel on the signal complicates the transceiver identification. To overcome this problem, some authors propose to equalize the channel [8], [15]. However, in this paper, we consider that the identification system has no information on the channel and classifies the IQ signals without demodulation and decoding operations.

B. Classification with Deep Learning

In recent years, DL has been massively used for classification as it could learn automatically how to classify radio transmitter [7]–[10]. Fig. 2 describes the classification procedure. First, the SDR receiver stores the IQ samples at the sampling frequency F_s to populate a labeled database, at the input of the identification stage. The database is then split in two parts 90% for train and 10% for test part. The network chosen in the present paper for comparison has been proposed and studied for RFF identification in [10]. It is a CNN inspired by AlexNet, with 4 convolutional layers, and each layer is composed of two blocks of 128 filters size 7×1 and 5×1 and a maxpooling stage. The chosen activation function is ReLu and the optimizer is Adam. After the 4 convolutional layers, the CNN has 3 fully connected layers with 256 nodes, 128 and the number of class (in this case 6 classes). This architecture is described in Fig. 3 and has 1,232,774 parameters.

In the training part, the network takes 256 IQ samples grouped in batches of size 600 sub-signals in input. The labels of the signals in the batch are calculated and compared with the true labels using cross-entropy as the loss function to apply the back-propagation. This process is repeated for each batch and each epoch.

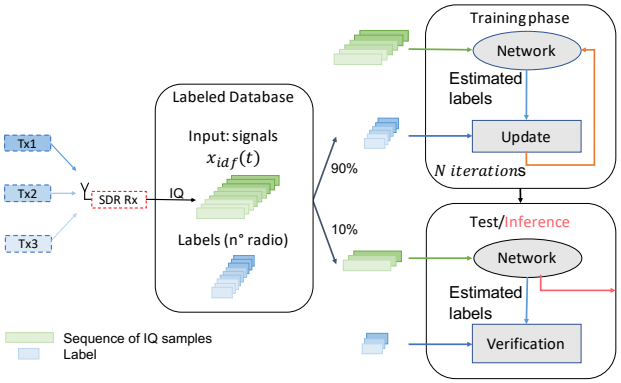

Fig. 2. Process chain with machine learning techniques.

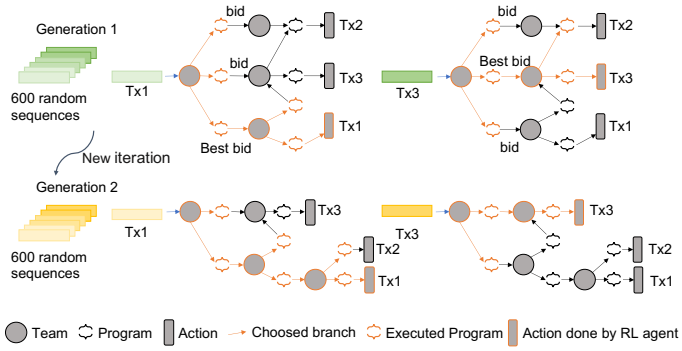

Fig. 4. Semantics of the Tangled Program Graphs (TPGs).

At the end of the learning phase, the network could now take unlabeled IQ samples in the input and estimate the radio label. This is called inference phase, or test if we can compare the results with true labels.

III. Tangled Programm Graph

A. A brief introduction of TPG

Introduced in 2017, TPG is a successful Reinforcement Learning (RL) model [16] that builds on SotA genetic programming techniques. Unlike DNNs whose topology is generally chosen by an expert data scientist, TPGs are grown from scratch for each learning environment, and their topology and computational complexity adapt automatically to the complexity of the learned task. TPGs have proven to be competitive with SotA DNNs, providing several order of magnitude improvements in computational complexity and memory requirements on various use cases, with gains at both training and inference [14].

A TPG is structured as a directed graph whose vertices and edges, called teams and programs, respectively, specify a control flow of an RL agent, and not a data flow as in DNNs. The control flow of the TPG stems from its root vertex, each time a new state of the learning environment is observed. All programs associated to outgoing edges of the root team are executed with the current state of the environment as their input. An example of TPG and the semantics is given in Fig. 4.

A program is a genetically evolved assembly sequence of instructions taking as inputs the different variables exposed by the environment and returning a single value, called a bid, per program. Once all programs have completed their execution, the edge associated to the largest output bid is identified, and the execution of the TPG continues following this edge. Eventually, the edge with the largest bid leads to a leaf vertex, associated to a specific action of the RL agent on the environment.

The TPG is grown from scratch for each learning environment, and complexity is added to the model if it leads to a greater reward. This makes the complexity of the TPGs dependent on the complexity of learned task [16], [17].

The training process for TPG is not based on gradient descent, like DNNs, but on a genetic algorithm. The genetic algorithm is a bio-inspired optimization algorithm. An initial graph is randomly created with different roots where each root represents a different policy. After objective evaluations which affect some reward to each individual policy, the algorithm selects the roots associated to the greatest rewards, and removes the other ones from the graph. At each generation of the training process, new root teams are introduced in the graph by randomly copying and mutating surviving ones.

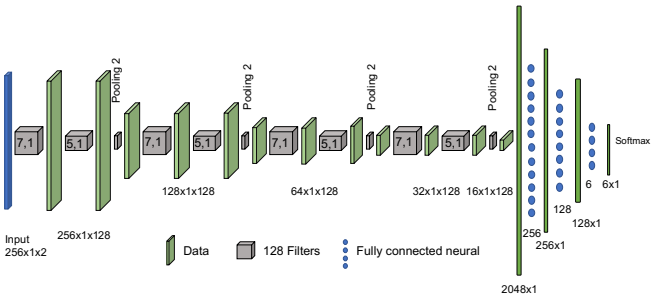

Fig. 3. Process chain with machine learning techniques.

B. TPGs for RFF classification

Despite being initially proposed for RL, TPGs are also used for classification. In this case, an Action represents a class membership decision. For example, TPG-based classification apply on the CIFAR-10 dataset achieves interesting results [13].

TPG-based classification leads to similar framework as the one described in Fig. 2. The network is the TPG, the update phase is done by a genetic algorithm and an iteration corresponds to a generation. The inputs of TPG for each prediction are a set of 256 IQ samples as it is done with DL. At each generation, each root of the TPG takes 600 random

sequences of 256 samples in input. The reward is based on the F1 score which is calculated on the 600 sequences as follows:

$$\text{F1} = \mathop{\mathbb{E}}_{c\in\mathcal{C}}\left(\frac{2}{\frac{1}{P(c)}+\frac{1}{R(c)}}\right), \quad (4)$$
$$\text{with } \begin{cases} P(c) = \frac{tp(c)}{tp(c)+fp(c)} \\ R(c) = \frac{tp(c)}{tp(c)+fn(c)} \end{cases}$$

where $\mathbb{E}[\cdot]$ stands for the expectancy operator applied here on all the classes $c \in \mathcal{C}$. $P(c)$ is called the precision for the class c and is a function of the number of true positives $tp(c)$ and false positives $fp(c)$. $R(c)$ is the recall for the class c and is function of $tp(c)$ and the false negatives $fn(c)$.

The genetic update of TPG could create a solution where one class is not classified [13]. Because the global accuracy or F1 score is given as a reward, this may hide disparities between classes, with a class being perfectly detected all the time, and another never. That's why the TPG update, in classification case, changes to conserve at least one sub-graph per class [18]. In the implementation used throughout our experiments, the natural selection process has been modified as follows. When selecting the n best roots that survive for the next generation of the training, $p\%$ of the roots are selected based on their averaged F1 score on all m classes, while the other $(100-p)\%$ are selected for their F1 score on a single class. In section IV, $p = 10\%$ is used.

IV. Experiments and Results

A. Database

To experiment the capacity of TPG for device classification the database choice is noteworthy. Many public databases exist such as Oracle [9], WiSig [15] and the one proposed in [19]. The other database often used in RFF identification is the DARPA database [7], [8], [20], [21] but this database has no public access. Building a database of IQ signals from scratch is a difficult task because many parameters could affect the detection reliability. In particular, the SotA shows it is difficult to handle the dynamic channel.

In this paper, we use WiSig because of the following reasons. WiSig is a recent database that is constructed with many signals and with a lot of information on how signals have been captured as transmitters positioning and type of radio used. They provide a large-scale WiFi dataset captured by 41 USRPs with 20 MHz bandwidth from different references. The signals come from 174 WiFi transmitters over four different captures performed over a month. The authors have created different databases with many transmitters (150), many receivers (32), many signals (1000 for each transmitter). For our experiments, we choose the ManySig dataset with 6 transmitters and 12 receivers. We have reproduced the locations of the transmitters and receivers to study the impact of the channel and this can be observed in Fig. 5.

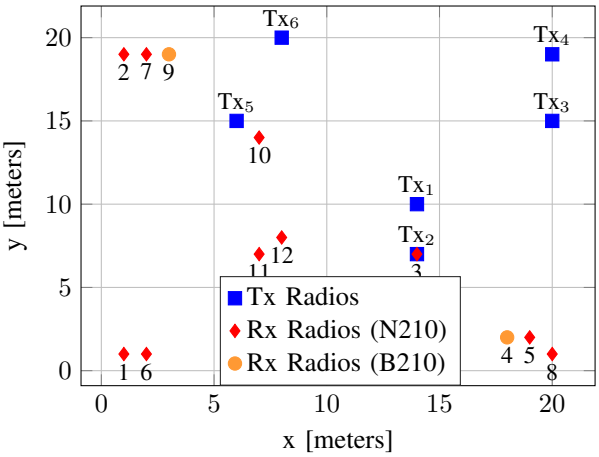

Fig. 5. Locations of Tx and Rx in the Orbit grid, for ManySig dataset.

Each transceiver represented in blue square in Fig. 5 has emitted 1000 signals of 256 IQ samples. All transmitters are Atheros AR5212 AR5213. The receivers are dispatched in the room so the propagation channel may differ from one radio to another. The database is split in two parts 90% (5400 signals) for training and 10% (600 signals) for the test. Both datasets are balanced as they contain signals from all transceiver radios with a balanced ratio.

B. Timing and accuracy comparison in a favorable scenario

In this section, performances of both TPG and CNN are compared. Both algorithms are trained on the CPU of a core Intel i7-8850H @2.60GHz with 6 cores and 12 threads and with SSE4.2 and AVX2 extensions. The CNN is also trained on a GPU NVIDIA Quadro P1000. The TPG is not implemented on the GPU as its non-symmetric structure is not suitable for such architecture. The WiSig database offers many degrees of freedom such as: day of capture data, receiver radio (positions and references). In this first experiment, data are received on day 1 by the radio Rx1 for both training and test phase. It corresponds to a favorable scenario for training and identification because the receiver is the same for all signals and the relative position is different for all transmitters moreover the testing scenario is the same as the learning one.

Tables I and II give the confusion matrices obtained with the TPG network [18] and the AlexNet-inspired CNN respectively. The rows of the confusion matrix are the true labels while the columns are the labels estimated by the network. The numbers represent the percentage obtained for each case. Those matrices show the capacity of TPG to correctly learn to labelled radio. Table I allows to validate the correct functioning of TPG. Both confusion matrices are very similar.

TABLE I
CONFUSION MATRIX OBTAINED WITH TPG FOR TRAINING AND TEST IN SAME CONDITIONS

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	96.0	1.0	0.0	1.0	0.0	2.0
Tx_2	0.0	93.0	0.0	7.0	0.0	0.0
Tx_3	0.0	3.0	95.0	0.0	0.0	2.0
Tx_4	1.0	3.0	0.0	96.0	0.0	0.0
Tx_5	0.0	0.0	1.0	0.0	99.0	0.0
Tx_6	0.0	0.0	0.0	0.0	0.0	100.0

TABLE II
CONFUSION MATRIX OBTAINED WITH CNN FOR TRAINING AND TEST IN SAME CONDITIONS

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	100.0	0.0	0.0	0.0	0.0	0.0
Tx_2	0.0	87.5	0.0	12.5	0.0	0.0
Tx_3	0.0	30.0	70.0	0.0	0.0	0.0
Tx_4	0.0	0.0	0.0	100.0	0.0	0.0
Tx_5	12.5	0.0	0.0	0.0	87.5	0.0
Tx_6	0.0	0.0	0.0	0.0	0.0	100.0

To compare those results in terms of timing, Fig. 6 gives F1 scores during the training phase as a function of time. The yellow triangles represent the F1 score of TPG during the training phase on the CPU. The blue triangles represent the evolution of the F1 score for the CNN learning phase on the CPU while the blue squares correspond to the F1 score of the CNN using the GPU. When considering CPU, the TPG exhibits an important speed-up when compared to the CNN. Its speed is very close to a CNN training on a GPU with two advantages (i) the learning can be done on a platform without the specific GPU accelerator with similar speed (ii) the energy consumption is reduced as only the CPU is used for the TPG.

This analysis shows similar performance between TPG and DL. In the rest of the paper TPGs are used to deeper analyse the database and propose an interpretation of the impact of the propagation channel and the receiver. The analysis and the interpretation in the rest of the paper are only done for TPG but the same comportment has been observed for the CNN based solutions.

C. Mitigation of environment impact using augmentation

The impact of environmental change on classification accuracy is now evaluated. In this part, the only changing factor is the day of the emission. The locations of radios do not change, so the propagation channel should not change either. However, the radios are not in controlled room, so 3 factors can affect the RFF:

- The environment channel: the radios are not in an anechoïc chamber and interference signals may alter the quality of the labeled database.

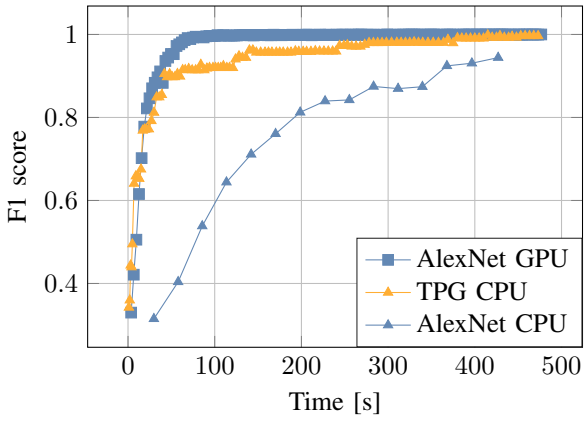

Fig. 6. Time evolution of the F1 score of the different networks on different hardware.

TABLE III
CONFUSION MATRIX OBTAINED WITH TPG FOR TEST DONE ON DIFFERENT DAYS

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	77.4	20.0	0.0	1.0	0.4	1.2
Tx_2	2.7	40.8	0.0	56.5	0.0	0.0
Tx_3	6.2	29.7	7.5	0.0	25.3	31.4
Tx_4	2.1	17.3	1.1	79.5	0.0	0.0
Tx_5	0.4	0.0	0.4	0.0	99.2	0.0
Tx_6	0.6	3.1	56.4	3.7	4.3	31.9

- The environmental conditions: the ambient factors as humidity or temperature are not controlled in the room and can impact the performance of the components and changed the distortions.
- The RFF modifications over time: the days of capture signals are distributed over one month so the component degradation could impact the RFF of the devices.

Table III gives the average accuracy for a test, realised on signals from days 2, 3 and 4, whereas the training is realised with signals from only day 1.

The confusion matrix shows how difficult it is to generalize the training with other environmental conditions. This problem has been yet shown in [7] for CNN where the authors proposed to realize a data augmentation to present many different environmental conditions to the network. With this augmentation, the network should learn the RFF without the implication of environment. The augmentation could be realized physically or virtually. For physical augmentation, the number of experiments is increased to create more environmental conditions. For virtual augmentation, diversity is added on signals with an environmental model. Here, the database offers the possibility to physically augment the training dataset thanks to the different days captured. Data from day 2, 3 and 4 is used for training and the test is realized on day 1.

TABLE IV
CONFUSION MATRIX OBTAINED WITH TPG FOR AUGMENTED DAYS TRAINING AND TEST ON DAY 1

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	100.0	0.0	0.0	0.0	0.0	0.0
Tx_2	1.0	60.0	21.0	18.0	0.0	0.0
Tx_3	3.0	1.0	17.0	1.0	7.0	71.0
Tx_4	0.0	0.0	1.0	99.0	0.0	0.0
Tx_5	0.0	0.0	1.0	0.0	99.0	0.0
Tx_6	0.0	11.0	0.0	0.0	0.0	89.0

Table IV gives the accuracy results achieved with data augmentation. It shows that TPG is able to generalize the identification and a better accuracy is achieved for the new day. The comparison with Table III shows that physical data augmentation is interesting for RFF classification, especially with environments where variations may occur.

D. Influence of the training conditions

In this section, the impact of both the propagation channel and the receiver is analyzed. The WiSig database is well documented and contained information on the kind of radio used, later denoted by reference. The key point is that all the radio receivers are the same reference (SDR N210) except two radios: radios number 4 (Rx4) and 9 (Rx9), represented by orange circles in Fig. 5. To stress the impact of the radios, all training phases are realized with receiver Rx1.

1) Influence of the channel: The first analysis is done with signals from radio Rx6. Radios Rx1 and Rx6 are one meter distance and they are the same reference. Hence, the channel propagation has changed because of the distance between receivers and the RFF of the receiver has changed a little because of the singularity of the two systems. Table V shows that our emitters are, on average, correctly identified but the detection performance has been altered compared to the ideal case exposed in Section V-B. Two conclusions can be drawn: Two receivers radios from the same reference and with closed position could be swapped during training and test phases with an accuracy penalty with respect to the ideal case. It also proves that a difference propagation channel between two devices from the same reference affects the results or, in other words, that the network learns a part of the propagation channel.

We now propose to realise the same analysis with the test done on signals from radio Rx7. This radio is the same reference as Rx1 and Rx6 but it is localized at the opposite of the room. Table VI shows that three emitters are correctly identified and the average identification accuracy decreases in comparison with the results achieved by Rx6. The main difference between the two radios is the location. So in a dynamic context, for which the channel propagation changes between training and test steps, the identification capacity decreases. To mitigate this phenomenon, we propose a channel augmentation using different radios with the same reference. The confusion matrix VII shows the result of a training realized on signals from Rx1, 2 and 3 and test done on signals from Rx5, 6, 7 and 8. The results are clearly better with the augmentation. The TPG is able to identify transmitters with other receivers in different locations but same reference, when the training phase is done with a diversity of radios and locations. This result would be even better by enhancing the augmentation with data from Rx1 moving at more locations.

2) Influence of the receiver RFF: Finally, we realize the test on the signals from radio Rx9 again with a training phase on Rx1. Rx9 is close to Rx7 so we can expect similar confusion matrix as in Table VI". The main difference between Rx7 and Rx9 is the reference of radio. Rx9 is B210 when Rx7, 1 and 6 are N210. Table VIII shows the results obtained with this configuration and shows the incapacity to correctly identify the radios and in particular a strong performance penalty with respect to Table VI.

Some key assets can be drawn here: even a slight modification of the propagation channel or the environment propagation may lead to an important drop of the detection accuracy. Physical data augmentation is thus required to keep good generalization properties. Besides, we prove here that the receiver RFF has a tremendous impact on the capacity to accurately classify a transmitter, even more than the propagation channel itself. It shows the necessity to propose diverse and extensive dataset that can be physically augmented and with a strong variety of propagation channels, environment characteristics and strong diversity in both transmitter and receiver references.

V. CONCLUSION

This paper proposes to use a new machine learning technique called TPG to identify devices with RFF recognition. The results show a fast F1 score progression of TPG during the training phase on the CPU. The progression is very close to F1 score progression of SotA CNN on the GPU. In the second part, TPGs are used to assess a deep analysis of the chosen database and interpretations of the impact of the propagation channel and the receiver are proposed. The analysis concludes with the negative impact of changing captured conditions between training and test phase for identification. The paper presents the interest of physical data augmentation to be able to identify the transmitters in different situations. The augmentation has to be on different days and proposes different realistic configuration that we can have in inference phase.

Acknowledgements

This work is funded by DGA and Brittany region under the Creach Lab founding and by the French National Research Agency (ANR) under the grant number ANR-22-CE25-

TABLE V
CONFUSION MATRIX OBTAINED WITH TPG FOR TEST DONE ON RX 6

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	75.0	8.5	1.1	8.1	2.7	4.6
Tx_2	4.0	57.6	0.4	36.5	1.3	0.1
Tx_3	1.2	11.9	0.6	0.5	5.4	80.4
Tx_4	3.0	7.9	0.5	86.7	1.1	0.8
Tx_5	1.4	0.1	3.8	0.5	94.2	0.0
Tx_6	7.9	51.1	9.8	0.1	31.0	0.1

TABLE VI
CONFUSION MATRIX OBTAINED WITH TPG FOR TEST DONE ON RX 7

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	83.2	8.7	0.9	4.6	1.1	1.5
Tx_2	2.5	62.5	0.2	34.3	0.3	0.2
Tx_3	19.9	3.1	0.0	75.4	0.7	0.9
Tx_4	0.3	99.3	0.0	0.3	0.1	0.0
Tx_5	0.2	5.5	0.3	1.6	92.4	0.0
Tx_6	24.3	31.3	0.8	41.1	2.5	0.0

TABLE VII
CONFUSION MATRIX OBTAINED WITH TPG AND AUGMENTED RECEIVERS TRAINING AND TEST ON RECEIVERS 5 TO 8

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	76.6	4.7	7.0	9.3	0.0	2.4
Tx_2	7.9	65.3	10.6	14.2	0.3	1.7
Tx_3	18.5	2.0	15.1	34.9	11.8	17.7
Tx_4	5.6	69.9	5.6	17.7	0.2	1.0
Tx_5	0.8	0.0	4.3	1.6	87.8	5.5
Tx_6	7.7	0.3	11.3	1.2	2.6	77.0

TABLE VIII
CONFUSION MATRIX OBTAINED WITH TPG FOR TEST DONE ON RX 9

True \ Guess	Tx_1	Tx_2	Tx_3	Tx_4	Tx_5	Tx_6
Tx_1	15.4	12.2	24.4	19.6	16.0	12.5
Tx_2	11.5	7.6	33.1	23.3	16.4	8.0
Tx_3	4.4	4.8	69.9	11.9	7.9	1.1
Tx_4	7.9	10.6	46.0	15.5	13.3	6.6
Tx_5	6.0	7.1	41.2	22.1	19.3	4.2
Tx_6	1.5	2.9	64.5	14.0	11.4	5.7

0007-01 (RedInBlack project) and ANR-22-CE25-0005-01 (FOUTICS project).

REFERENCES

[1] F. A. Alaba, M. Othman, I. A. T. Hashem, and F. Alotaibi, "Internet of things security: A survey," *Journal of Network and Computer Applications*, 2017.

[2] Y. Sheng, K. Tan, G. Chen, D. Kotz, and A. Campbell, "Detecting 802.11 mac layer spoofing using received signal strength," in *2008 IEEE INFOCOM The 27th Conference on Computer Communications*.

[3] A. Jagannath, J. Jagannath, and P. S. P. V. Kumar, "A Comprehensive Survey on Radio Frequency (RF) Fingerprinting: Traditional Approaches, Deep Learning, and Open Challenges," 2022.

[4] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *2008 Proceedings of the 14th ACM international conference on Mobile computing and networking - MobiCom 08*. San Francisco, California, USA: ACM Press, p. 116.

[5] S. U. Rehman, K. Sowerby, and C. Coghill, "Analysis of receiver front end on the performance of RF fingerprinting," in *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sydney, Australia, pp. 2494–2499.

[6] Y. Lin, J. Jia, S. Wang, B. Ge, and S. Mao, "Wireless Device Identification Based on Radio Frequency Fingerprint Features," in *2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, pp. 1–6.

[7] N. Soltani, K. Sankhe, J. Dy, S. Ioannidis, and K. Chowdhury, "More Is Better: Data Augmentation for Channel-Resilient RF Fingerprinting," *in 2020 IEEE Communications Magazine*, vol. 58, no. 10, pp. 66–72.

[8] A. Al-Shawabka, F. Restuccia, S. D'Oro, T. Jian, B. Costa Rendon, N. Soltani, J. Dy, S. Ioannidis, K. Chowdhury, and T. Melodia, "Exposing the Fingerprint: Dissecting the Impact of the Wireless Channel on Radio Fingerprinting," in *2020 INFOCOM IEEE Conference on Computer Communications*, Toronto, ON, Canada, pp. 646–655.

[9] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized Radio clAssification through Convolutional neuraL nEtworks," in *2019 INFOCOM IEEE Conference on Computer Communications*, Paris, France, pp. 370–378.

[10] K. Sankhe, M. Belgiovine, F. Zhou, L. Angioloni, F. Restuccia, S. D'Oro, T. Melodia, S. Ioannidis, and K. Chowdhury, "*No Radio Left Behind:* Radio Fingerprinting Through Deep Learning of Physical-Layer Hardware Impairments," *in 2020 IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 1, pp. 165–178.

[11] G. Shen, J. Zhang, A. Marshall, M. Valkama, and J. Cavallaro, "Radio Frequency Fingerprint Identification for Security in Low-Cost IoT Devices," *arXiv:2111.14275 [eess]*, 2021.

[12] A. Zhu, J. C. Pedro, and T. J. Brazil, "Dynamic deviation reduction-based volterra behavioral modeling of RF power amplifiers," *IEEE Transactions on Microwave Theory and Techniques*, vol. 54, no. 12, pp. 4323–4332, 2006.

[13] R. J. Smith, R. Amaral, and M. I. Heywood, "Evolving simple solutions to the cifar-10 benchmark using tangled program graphs," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2061–2068.

[14] S. Kelly, "Scaling Genetic Programming to Challenging Reinforcement Tasks through Emergent Modularity," 2018.

[15] S. Hanna, S. Karunaratne, and D. Cabric, "WiSig: A Large-Scale WiFi Signal Dataset for Receiver and Channel Agnostic RF Fingerprinting," arXiv, Tech. Rep. arXiv:2112.15363, 2022.

[16] S. Kelly and M. I. Heywood, "Emergent tangled graph representations for atari game playing agents," in *2017 European Conference on Genetic Programming*, pp. 64–79.

[17] S. Kelly, J. Newsted, W. Banzhaf, and C. Gondro, "A modular memory framework for time series prediction," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, ser. GECCO '20. New York, NY, USA: Association for Computing Machinery, pp. 949–957.

[18] K. Desnos, N. Sourbier, P.-Y. Raumer, O. Gesny, and M. Pelcat, "Gegelati: Lightweight artificial intelligence through generic and evolvable tangled program graphs," in *2021 Workshop on Design and Architectures for Signal and Image Processing (DASIP)*, ser. International Conference Proceedings Series (ICPS). Budapest, Hungary: ACM.

[19] C. Morin, L. Cardoso, J. Hoydis, J.-M. Gorce, and T. Vial, "Transmitter Classification With Supervised Deep Learning," *arXiv:1905.07923 [cs, eess]*, 2019.

[20] J. Robinson, S. Kuzdeba, J. Stankowicz, and J. M. Carmack, "Dilated Causal Convolutional Model For RF Fingerprinting," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC) IEEE*, Las Vegas, NV, USA, pp. 0157–0162.

[21] J. Robinson and S. Kuzdeba, "RiftNet: Radio Frequency Classification for Large Populations," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, Las Vegas, NV, USA, pp. 1–6.