Finding a 'New' Needle in the Haystack: Unseen Radio Detection in Large Populations Using Deep Learning

Andrey Gritsenko*, Zifeng Wang*, Tong Jian, Jennifer Dy, Kaushik Chowdhury, and Stratis Ioannidis
Department of Electrical and Computer Engineering
Northeastern University
Boston, USA
Email: {agritsenko, zifengwang, jian, jdy, krc, ioannidis}@ece.neu.edu

***Abstract*—Radio frequency fingerprinting enhances security and privacy of wireless networks and communications by learning and extracting unique characteristics embedded in transmitted signals. Deep learning-based approaches learn radio fingerprints without hand-engineering features. One persisting drawback in deep learning methods is they identify only devices that are previously observed in a training set: if a radio signal from a new, unseen, device is passed through the classifier, the source device will be classified as one of the known devices. We propose a novel approach that facilitates new class detection without retraining a neural network, and perform extensive analysis of the proposed model both in terms of model parameters and real-world datasets. We accomplish this by first breaking down a longer transmission burst into smaller slices, and assessing classifier confidence on a new transmission based on per slice statistics: our approach detects a new device with 76% accuracy, while reducing the classification accuracy of 500 previously seen devices by no more than 10%.**

***Index Terms*—Radio-Frequency Fingerprinting, Transmitter Identification, Novel-Class Detection, Convolutional Neural Networks, Statistical Inference**

I. Introduction

Radio frequency (RF) fingerprinting enhances security and privacy of wireless networks and communications by learning and extracting unique characteristics embedded in transmitted signals. The key idea is that many non-linear artifacts are introduced in the signal by a transmitter's processing chain, which serve as a unique identity for that transmitter [1–5]. Many existing works aim to detect such fingerprints through complex, custom-designed, and protocol-specific feature-extraction models [6–8]. However, recent advances in the area of machine learning provide an alternative way of learning these radio fingerprints without hand-engineering of the features, i.e., through deep neural networks that automatically learn device-specific features [9–11]. One persisting drawback in all these learning-based fingerprinting methods is that they identify only those devices that are previously observed. In other words, if a radio signal from a new, unseen, device is passed through the classifier, the source device will be classified as one of the known devices. Thus, the task of novel device detection is an essential but as yet overlooked problem in context of RF fingerprinting.

Novel device detection falls in the domain of *novelty detection* [12] or *zero-shot* learning [13] in general machine learning literature. Most existing novelty detection methods can be categorized as either kernel density-based, nearest neighbour-based, or reconstruction-based. In kernel density-based methods, the probability density function is estimated using large numbers of kernels distributed over the data space [14, 15]. Modeling the probability distribution over the data has achieved success on small-scale datasets, however, for high-dimensional and large datasets (like the one we use in this paper), it is both computationally expensive and prone to overfitting. Nearest neighbour-based methods rely on the assumption that normal data points have close neighbours in the seen classes, while novel points are located far from those points [16–18]. The definition of distance metric is critical for these methods but it is not well defined for radio signals that carry different contents or are of various lengths. Reconstruction-based methods, which rely heavily on neural networks, construct an encoding-decoding pipeline and compute a novelty score through reconstruction error [19]. One drawback of these methods is that they need to train a separate reconstruction neural network besides the classification pipeline. Moreover, previous works propose various methods to address novelty detection with benchmark image or text datasets only. To the best of our knowledge, we are the first to tackle this problem (a) on a complicated real-world radio fingerprinting dataset (b) with minimal additional computational cost compared to the original classification task.

Our approach in designing the novel device detection method described in this paper is simple: Given a dataset of signals transmitted by a group of known, registered, wireless devices, we design a framework that trains a classifier to correctly identify a source of transmission, if it occurs from one of the registered devices; otherwise, the classifier detects that a novel device is observed. We do this by first breaking down a longer transmission burst composed of a stream of in-phase/quadrature (I/Q) samples into smaller slices. Each slice is classified separately. This allows us to assess classifier

* A.Gritsenko and Z.Wang contributed equally to the paper.

confidence on a new transmission based on per slice statistics: we use these to determine whether the transmitter for the given radio signal is one of the known devices, or it has never been seen before. Crucially, our approach is generic and does not require retraining the classifier.

Our main contributions are: (1) we propose a novel approach to train a neural network that facilitates new class detection, (2) we perform extensive analysis of the proposed model both in terms of model parameters and real-world datasets. Our approach detects a new device with 76% accuracy, while reducing the classification accuracy of 500 previously seen devices by no more than 10%.

The rest of the paper is organized as follows. In Section II, we describe three datasets used in this study, including how we collect, organize and preprocess data. Next, we define general RF fingerprinting problem with both seen and unseen devices precisely in Section III, and present architectures of our classifier as well as the method of classifying known devices. In Section IV, we introduce and elaborate our original algorithm for novel radio device detection. Section V covers in detail the overall experimental setting and the evaluation metrics we use, followed by the extensive analysis on the experiment results in Section VI. There, we also carry out a comprehensive investigation on the influence of different settings on classifiers' performance. Finally, we draw the conclusion of the paper, emphasizing the uniqueness and effectiveness of our method in Section VII.

TABLE I
NOTATION SUMMARY

Variable	Description
$\aleph$	Collection of in-library devices
$\aleph'$	Collection of new, out-of-library, devices
K_d	Number of signals transmitted by device $d \in \aleph \cup \aleph'$
$M^{(k)}$	Length of transmission $k \in [1, K_d]$
L	Slice size (number of I/Q samples)
λ	Parameter regulating the number of slices extracted from a given transmission, $\lambda \in [1, L]$
$n^{(k)}$	Number of slices extracted from transmission k
$p_{ij}^{(k)}$	Probability of slice i from transmission k to be captured from device j, $i \in [1, n^{(k)}], j \in \aleph$
$y^{(k)}$	True label of transmission k, $y^{(k)} \in \{1, \ldots, \lvert\aleph\rvert, new\}$
$\hat{y}^{(k)}$	Predicted label of transmission k
$S_d^{(k)}$	Set of correctly classified slices, extracted from training transmission k of device d
$N^{(k)}$	Number of correctly classified slices, extracted from training transmission k
$r^{(k)}$	Ratio of correctly classified slices, extracted from training transmission k
$\lvert A\rvert$	Cardinality of a set A

TABLE II
DATASET CHARACTERISTICS

	WiFi		ADS-B	
$\lvert\aleph\rvert$	$\lvert\aleph'\rvert$	$\overline{M}$	$\lvert\aleph'\rvert$	$\overline{M}$
500	549	15979	451	9458
250	542	15539	458	9397
50	536	13071	464	9526

For each dataset, we provide the total number of the known, in-library, devices, number of new, out-of-library, devices and the average length of transmitted signals.

II. DATASET

A. Data Collection

We use a database of radio signals recorded in the wild, each signal represented as a sequence of its I and Q components, simply referred to as I/Q samples. The database contains transmissions from two different wireless protocols, specifically, commercial off the shelf (COTS) 802.11b/g/n WiFi and the Automatic Dependent Surveillance-Broadcast (ADS-B) protocol used to transmit signals of airplane status updates. All collected ADS-B signals were transmitted at a center frequency of 1.09 GHz with a sampling rate of 10^8 samples (I/Q pairs) per second. For WiFi data, transmissions with both 2.4 GHz and 5.8 GHz center frequencies were captured at a sampling rate varying between 20-200 Msps.

B. Dataset Structure

The database used in this study contains three datasets of different sizes for each transmission protocol (WiFi and ADS-B). Specifically, we use datasets with $|\aleph| = 500, 250$ and 50 known devices, where $\aleph$ stands for a collection of in-library devices. For all six datasets, $K_d = 176$ transmissions are captured for every in-library device $d \in \aleph$. Additionally, each dataset contains radio signals transmitted by a set $\aleph'$ of new, out-of-library, devices ($|\aleph'| = 542$ for WiFi and $|\aleph'| = 458$ for ADS-B protocols, on average), where $\aleph \cap \aleph' = \varnothing$. Exactly one transmission for each out-of-library device $d \in \aleph'$ was recorded for both wireless protocols. The average length of WiFi transmissions is $\overline{M} = 1.4 \cdot 10^4$ I/Q samples across all datasets, and the average transmission length for ADS-B signals is $\overline{M} = 9.5 \cdot 10^3$ I/Q samples. Table II presents more detailed statistics for all six datasets and both wireless protocols.

To train and evaluate classifiers, we split each dataset into training and test subsets. The training set contains 80% of transmissions captured from in-library devices ($K_{d,train} = 141, d \in \aleph$), while the test set contains all transmissions from out-of-library devices ($K_{d,test} = 1, d \in \aleph'$) in addition to 20% of transmissions from in-library devices ($K_{d,test} = 35, d \in \aleph$).

C. Data Preprocessing

We next describe our preprocessing pipeline, consisting of filtering, equalization, and slicing.

Filtering. WiFi transmissions are recorded in a very complex, multi-device environment, i.e., multiple devices transmitting radio signals at the same time at different frequency bands (WiFi channels). Thus, we first need to extract a single device transmission in order to properly process it. This is accomplished through a filtering process that takes a signal at a given center frequency, for e.g., 2.4 GHz, and moves it to

baseband following which it applies a low pass filter. We refer to this extracted single device sequence of I/Q samples as a *transmission*.

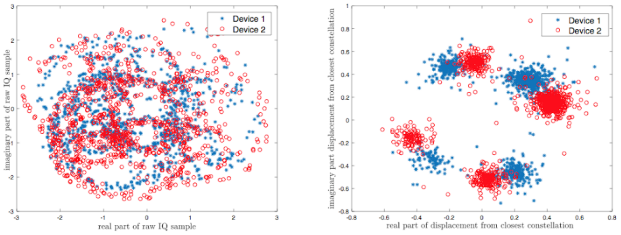
Equalization. Though the resulting filtered WiFi transmission is free from adjacent channel interference and any out-of-band noise, it is still exposed to in-band noise. As a result, instead of learning individual device fingerprints from the radio signal, a classifier may end up learning the so-called *channel conditions*. In order to alleviate the influence of channel conditions on classification accuracy, we partially equalize filtered WiFi transmissions. This equalization process constitutes of three steps:

(a) We use short training sequences for frame synchronization and estimation of coarse frequency and sampling offsets;
(b) We use long training sequences for channel estimation and estimation of fine frequency and sampling offsets; and
(c) We equalize the channel and reintroduce the frequency offsets computed at step (a).

As a result, we obtain a final sequence of partially equalized I/Q samples. Fig. 1 visually illustrates effects of the equalization process on the I/Q samples corresponding to two different devices.

Slicing. As briefly mentioned in the previous section, the provided dataset contains radio transmissions with varying length (number of I/Q samples). Obviously, this type of data cannot be used by the neural network classifier that takes as an input data of fixed length. In order to overcome this issue, we utilize sliding window to disseminate each signal into a sequence of slices of the same length. Slices are then forwarded as an input to a neural network, labeled with the same device ID as the original transmission. The slicing operation is illustrated on Fig. 2a, and Fig. 2b depicts the final structure of the dataset used in the experiments after slicing. Using sliding window with stride equal to 1 and predefined slice size of L I/Q samples, we generate in total $M^{(k)} - L + 1$ slices for a given transmission k of $M^{(k)}$ length.

Slicing is a necessary step for our framework, and it is always performed regardless of the type of data we process. Because filtering is also essential for WiFi data and is implemented for all transmissions of this protocol, we refer to filtered WiFi transmissions as *raw* WiFi in the rest of the paper.



(a) (b)

Fig. 1. Constellation of (a) raw I/Q samples before equalization and (b) demodulated I/Q samples after equalization without frequency and offset corrections for two software-defined radio devices transmitting WiFi signal.

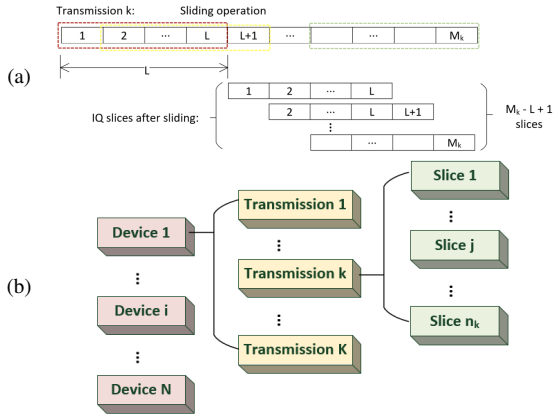


Fig. 2. Breaking down a stream of I/Q samples into discrete sequences through slicing. The total number of all possible overlapping slices is equal to $M^{(k)} - L + 1$ on slicing of a single transmission k of length $M^{(k)}$ samples, with slice size L.

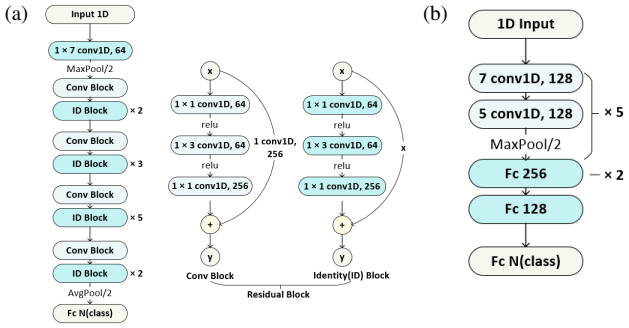


Fig. 3. Architecture of the ResNet1D (a) and AlexNet1D (b) models.

III. Methodology

In this section, we state the general problem of RF fingerprinting in the environment in the presence of new, unseen, devices. Then, we provide a detailed description of two neural network models implemented to solve the task, the training process, and finally the wireless device inference procedure.

Problem Formulation. Given a dataset of radio transmissions, a collection $\aleph$ of known devices, and a set $\aleph'$ of unknown devices, we formulate the following research problem: for each transmission k we need to predict its label $y^{(k)}$ that specifies whether a signal is transmitted by one of the known, in-library, devices ($y^{(k)} \in \{1, \ldots, |\aleph|\} \iff d \in \aleph$) or unseen, out-of-library, device ($y^{(k)} = new \iff d \in \aleph'$). In order to accurately classify the source of a radio transmission, we use a deep neural network (DNN) with multiple convolutional layers.

Structure of Deep Neural Networks. In this paper, we implement and examine the performance of two DNN models. Before we describe in detail architectures of both networks, we first introduce their major building blocks: convolutional, max-pooling and fully-connected, dense, layers. Convolutional layer, as the name suggests, convolves a number of different

filters through the full depth of the input volume. Filters used in the same convolutional layer are of the fixed size, and the output of the layer is a sequence of feature maps with decreased dimension compared to the original input. Max-pooling layer is used to further down-sample the input by dividing it into non-overlapping 'pools' and returning only the maximum value of a pool. Convolutional and max-pooling layers are usually grouped together to perform initial feature extraction, followed by fully-connected layers that learn high-level features and conduct the final prediction. In contrast to convolutional and max-pooling layers, every neuron in a dense layer is connected to every neuron in the previous layer.

ResNet1D. Our first model is a modification of the well-known ResNet-50 architecture [20], i.e., short for Residual Networks. ResNet achieved great success on various computer vision tasks and is now the backbone of many deep learning frameworks. It addresses the so-called vanishing gradient problem [21], commonly encountered in training of very deep neural networks, by means of skip connections inside residual blocks. The skip connections between layers add the outputs from previous layers to the outputs of stacked layers.

Although ResNet is powerful, it does not allow direct application in context of wireless IQ symbol which are of the form of 1D time series, instead of 2D images. In order to take advantage of the representation ability of ResNet for in the domain of wireless, we extend ResNet to ResNet1D by substituting all 2D convolutional layers with their 1D counterparts. Following the structure of the original ResNet-50, our ResNet1D uses identity blocks and convolutional blocks as basic elements. As illustrated on Fig. 3a, identity block consists of size 1×1 1D convolutional layer with 64 filters, size 1×3 1D convolutional layer with 64 filters and size 1×1 1D convolutional layer with 256 filters as the main path and an identity link as the skip connection. A convolutional block shares almost the same structure as identity block, except that the skip connection becomes a size 1D convolutional layer with 256 filters. Compensating for its large number of convolutional layers, ResNet1D features zero padding for cases when the input sequence becomes too small to be processed by consecutive layers.

AlexNet1D. The structure of the second model, hereinafter referred to as AlexNet1D, is inspired by another famous deep neural network with convolutional layers called AlexNet [22], and adapted for 1D time series input in the same manner as ResNet1D described above. This model is much smaller than ResNet1D and contains a total of 10 1D convolutional layers, followed by 3 fully-connected, dense, layers. Convolutional layers are grouped into 5 stacks, each stack containing 2 convolutional layers followed by one max pooling layer. The first convolutional layer in the stack is of size 1×7, while the second one has a size of 1×5. Both types of convolutional layers use 128 filters, with the model architecture depicted in Fig. 3b.

Training. Considering multi-class setting of a given classification problem, the size of the final, output layer of both ResNet1D and AlexNet1D is equal to the number of devices in the training set, i.e. the number of in-library devices, $|\aleph|$. We use softmax function as the activation of the output layer, a conventional approach to predict for multiple classes simultaneously, so that for each given slice i the model outputs a vector p_i of probabilities. Thus, p_{ij} corresponds to the probability of slice i being transmitted by device j, and $\sum_{j=1}^{|\aleph|} p_{ij} = 1$.

Random Slicing. Here, we emphasize that not all $M^{(k)} - L + 1$ slices from the k-th transmission have to be used to successfully train a classifier. In practice, we would like to uniformly select only a small fraction of slices at random and use them as the input for a neural network. In the next paragraph, we describe the motivation for performing random slicing. The actual number of slices $n^{(k)}$ used for each transmission k during training is governed by a hyperparameter λ according to the following formula

$$n^{(k)} = \frac{M^{(k)} - L + 1}{L} \lambda, \tag{1}$$

where $M^{(k)}$ is the total number of I/Q samples in the k-th transmission and L is the slice size. The numerator in (1) is equal to the total number of all possible slices generated with a sliding window stride equal to 1. Therefore, λ ranging in the interval of $[1, L]$ can be intuitively seen as the expected value of how many times each I/Q sample appears during the training.

There are certain advantages of implementing random slicing. First of all, it inherently guarantees input data to be of the fixed size that can be processed by DNN models. Then, it improves robustness and reliability of neural networks through learning shift-invariant RF fingerprints for each device. Moreover, varying parameter λ tends to prevent model overfitting and reduce computational costs. Finally, we naturally aggregate predicted labels for multiple slices to infer device ID of the original radio transmission. This ensemble strategy generally results in overall accuracy boost. In this study, we utilize a majority vote rule to predict a label for a given transmission k:

$$\hat{y}^{(k)} = \underset{i}{\text{mode}}\{\arg\max_j (p_{ij}^{(k)})\}, \tag{2}$$

where $p_{ij}^{(k)}$ is the probability that the i-th slice was extracted from the k-th transmission of the j-th device.

IV. Detecting unseen devices

As introduced in the previous section, our framework is able to classify radio transmissions from a fixed set of known, in-library, devices. However, in real world applications, we often encounter transmissions from new, unseen devices (out-of-library). If we blindly follow the normal classification strategy by majority vote as stated in (2), a transmission from a new device will be wrongly classified as transmitted by one of the old devices, due to the fixed number of output nodes in a neural network. To address this non-trivial problem, we exploit the *probabilistic nature* of neural networks, as well as our use of slices, to introduce an original novel device detection method.

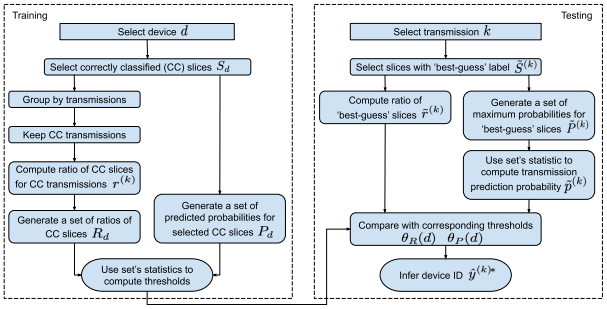


Fig. 4. Pipeline scheme for computing thresholds $\theta_P(d)$ and $\theta_R(d)$ during training, and using these thresholds at test time to infer device ID.

For each transmission k in the test set, we first compute the majority device label $\hat{y}^{(k)}$ via (2). Subsequently, to assess to the confidence of this prediction made by our classifier, we compute two quantities: the *transmission prediction probability* $\tilde{p}^{(k)}$ and the *estimated correct slice ratio* $\tilde{r}^{(k)}$. To finally determine whether an transmission is from a novel device, we compare these two quantities against (device-specific) thresholds $\theta_P(\hat{y}^{(k)})$ and $\theta_R(\hat{y}^{(k)})$, respectively. If both computed quantities $\tilde{p}^{(k)}$ and $\tilde{r}^{(k)}$ are smaller than the corresponding thresholds $\theta_P(\hat{y}^{(k)})$ and $\theta_R(\hat{y}^{(k)})$, we assert that the confidence level of the classifier is low; we thus conclude that a new radio device is detected. Otherwise, we state that a signal is transmitted by the 'best-guess' device $\hat{y}^{(k)}$. In summary, our modified classifier outputs:

$$\hat{y}^{(k)*} = \begin{cases} new, & \tilde{r}^{(k)} < \theta_R(\hat{y}^{(k)}) \wedge \tilde{p}^{(k)} < \theta_P(\hat{y}^{(k)}), \\ \underset{i}{\text{mode}}\{\arg\max_j(p_{ij}^{(k)})\}, & \text{otherwise.} \end{cases} \tag{3}$$

We precisely define the quantities $\tilde{p}^{(k)}$, $\tilde{r}^{(k)}$ and thresholds $\theta_P(\hat{y}^{(k)})$, $\theta_R(\hat{y}^{(k)})$ in the sections below. Our entire inference pipeline is summarized in Fig. 4.

Transmission Prediction Probability $\tilde{p}^{(k)}$. Recall that for every test transmission k, we generate $n^{(k)}$ slices according to (1) and classify them using a trained model. Then, we employ majority vote rule (2) to define the 'best-guess' label $\hat{y}^{(k)}$ of the transmission. In order to evaluate our confidence in this prediction, we define the transmission prediction probability $\tilde{p}^{(k)}$ by following these steps:

1) Collect all slices that classified with the 'best-guess' label

$$\tilde{S}^{(k)} = \{i \mid \arg\max_j(p_{ij}^{(k)}) = \hat{y}^{(k)}\}; \tag{4}$$

2) Record maximum probabilities for slices with the 'best-guess' label

$$\tilde{P}^{(k)} = \{p_{ij}^{(k)} \mid i \in \tilde{S}^{(k)}\}; \tag{5}$$

3) Use statistics , e.g. mean value, of $\tilde{P}^{(k)}$ to calculate the transmission prediction probability as the confidence level in the predicted 'best-guess' label. Formally:

$$\tilde{p}^{(k)} = \chi(\tilde{P}^{(k)}), \tag{6}$$

where χ is a mapping from arbitrary set A to its corresponding statistics (e.g., the mean). We list alternative definitions of χ in Table III and further discuss these choices below, in section "Choice of Statistics".

The intuition behind transmission prediction probability is that in order to ensure both out-of-library device detection and in-library device classification, we need to measure the level of prediction confidence for each transmission. Ideally, for a given test transmission k from an in-library device, we aim for $\hat{y}_i^{(k)} = y^{(k)}$ for all slices i, where $y^{(k)}$ is the true label of transmission k. Moreover, we want the probability for the correct class to be notably greater than the probabilities for the other labels, $p_{iy^{(k)}}^{(k)} \gg p_{ij}^{(k)}$, as this indicates that a classifier is confidently making the correct prediction. On the other hand, $p_{i\hat{y}_i}^{(k)}$ being close to probabilities for other classes implies a classifier is making a decision without a clear winner, probably due to the fact slice i was captured from a new device. As a transmission consists of multiple slices, we introduce transmission prediction probability $\tilde{p}^{(k)}$ to unify prediction probabilities from per slice level to per transmission level. Intuitively, the less $\tilde{p}^{(k)}$ is, the less confident our classifier will be in the prediction result, indicating this transmission is from an unseen device.

Estimated Correct Slice Ratio $\tilde{r}^{(k)}$. In order to complement the transmission prediction probability and measure the prediction confidence from another perspective, we compute the ratio of estimated correctly classified slices $\tilde{r}^{(k)}$ for transmission k in the test set as follows:

1) Collect all slices that classified with the 'best-guess' label in set $\tilde{S}^{(k)}$ as in (4);
2) Compute the estimated correct slice ratio with the 'best-guess' label to the total number of slices:

$$\tilde{r}^{(k)} = \frac{|\tilde{S}^{(k)}|}{n^{(k)}}. \tag{7}$$

Consider that for some transmission k of a known device $d \in \aleph$ in the test set containing $n^{(k)}$ slices, we obtain a corresponding set $s_d^{(k)}$ of slices with the predicted true label. In theory, we desire to have the vast majority of slices correctly classified, $|s_d^{(k)}| \gg |s_d^{(k)\prime}|$, in order to infer correct label for transmission k ($s_d^{(k)\prime}$ is a set of wrongly classified slices). On the other hand, when we process a signal transmitted by a new device $d \in \aleph'$, we still get a majority vote winner from a set of old devices, $\hat{y}^{(k)} \in \aleph$. However, we expect the proportion of 'best-guess' voters not to be significantly higher than for any other device. Ideally, we would want votes to be distributed almost equally among class labels. In this case, small proportion of 'best-guess' votes will be a sign that we should not be confident in the classifier prediction at the level of the whole transmission k, though individual slice predictions might be highly reliable ($p_{ij}^{(k)} \approx 1$). Therefore, estimated correct slice ratio captures the confidence level of prediction as well.

After calculating $\tilde{p}^{(k)}$ and $\tilde{r}^{(k)}$, we get two measures of prediction confidence from different perspectives. Recall from 3 that we determine that transmission k is from an unseen device when $\tilde{p}^{(k)}$ and $\tilde{r}^{(k)}$ are small enough, as determined by

probability threshold $\theta_P(\hat{y}^{(k)})$ and *ratio threshold* $\theta_R(\hat{y}^{(k)})$. The latter are (a) device-dependent, i.e., a different threshold is used for each majority device, and (b) learned: we obtain them in a data-driven way, by studying device statistics on the training set. We describe how to do so below.

Probability Threshold $\theta_P(d)$. The core idea is to have a certain probability threshold calculated from the training set to distinguish unreliable predictions, thereby detecting a new device. To get this threshold, we follow these steps:

1) Define all correctly classified slices, generated for transmission k of device d, as a set:
$$S_d^{(k)} = \{i \mid \arg\max_j(p_{ij}^{(k)}) = d\}; \tag{8}$$
2) Take the union of all sets of correctly classified slices across all transmissions of a given device $d \in \aleph$ in a training set:
$$S_d = \bigcup_{k=1}^{K_d} S_d^{(k)}, \tag{9}$$
where K_d is the number of transmissions captured from device d;
3) Collect all the corresponding maximum probabilities in a set:
$$P_d = \{p_{id} \mid i \in S_d\}; \tag{10}$$
4) Calculate the statistics of P_d as the probability threshold:
$$\theta_P(d) = \chi(P_d), \tag{11}$$
where, as in (6), χ is a statistic (e.g., the mean), among the ones described in Table III.

Ratio Threshold $\theta_R(d)$. Similarly, we also construct a threshold for estimated correct slice ratio to compare against in the following steps:

1) Compute the ratio of correctly classified slices for transmission k from device d in the training set as:
$$r^{(k)} = \frac{|S_d^{(k)}|}{n^{(k)}}, \tag{12}$$
where $S_d^{(k)}$ is defined in (8);
2) For each device $d \in \aleph$ we collect a set R_d of ratios $r^{(k)}$, computed for each correctly predicted transmission k:
$$R_d = \{r^{(k)} \mid \hat{y}^{(k)} = y^{(k)}\}; \tag{13}$$
3) Calculate the statistics of R_d as the ratio threshold:
$$\theta_R(d) = \chi(R_d). \tag{14}$$
Again, possible definitions of χ can be found in Table III.

Choice of Statistics. As mentioned above, we can compute thresholds $\theta_P(d)$ and $\theta_R(d)$ during training using the mean values of sets P_d and R_d, respectively. Similarly, we can compute $\tilde{p}^{(k)}$ as the mean value of $\tilde{P}^{(k)}$ at test time. Beyond the mean value, we also explore several other statistics χ in our experiments, including the minimum value and three lower confidence bounds, as listed in Table III. When applied to thresholds, each computed quantity in the table implies a certain level of confidence and defines the lower bound for probability or ratio that have to be predicted with a 'best-guess' label in order to assure that an in-library device is spotted. While the first four statistics are rather common, the last one is specially designed to tolerate outliers of heavy-tailed real-world distributions.

TABLE III
SET STATISTICS USED FOR NEW DEVICE DETECTION

Notation	Map $A \mapsto \chi(A)$
avg	$\chi(A) = \bar{A} = \frac{1}{\lvert A\rvert}\sum_{a\in A} a$
min	$\chi(A) = \lfloor A \rfloor = \min(A)$
lcb1	$\chi(A) = \bar{A} - std(A)$
lcb2	$\chi(A) = \bar{A} - 2 \cdot std(A)$
lcb3	$\chi(A) = \frac{\bar{A} - 2\cdot std(A) + \lfloor A \rfloor}{2}$

Statistic functions $\chi : 2^{\mathbb{R}} \to \mathbb{R}$. Here, $A \subset \mathbb{R}$ represents a finite set and in practice can be substituted, e.g., by a set P_d of predicted probabilities for all correctly classified slices for device $d \in \aleph$, a set R_d of ratios of correctly classified slice across all correctly labeled transmissions for device $d \in \aleph$, or a set $\tilde{P}^{(k)}$ of predicted probabilities for slices with 'best-guess' label for transmission k in the test set. For a given set A, $\bar{A}$ is the mean value of the set, $\min(A)$ is its minimum value, and $std(A)$ is its standard deviation. *lcb* stands for the 'lower confidence bound'.

To determine the best choice of statistics χ, applied for $\tilde{p}^{(k)}$ and both thresholds $\theta_P(d), \theta_R(d)$, we test our classifiers on all possible combinations in Table III. To limit the search space, first, we use the same function χ for both prediction and ratio thresholds $\theta_P(d)$ and $\theta_R(d)$. Second, we omit *avg* statistics when computing thresholds, as this statistic is too optimistic and results in thresholds $\theta_P(d)$ and $\theta_R(d)$ being much higher than corresponding quantities $\tilde{p}^{(k)}$ and $\tilde{r}^{(k)}$, even when computed for transmissions captured from in-library devices. As a result, we evaluate our models on a Cartesian product of four thresholding methods employed during training and five statistics employed for a set of slice predictions at test time.

Combining Thresholds. In (3), we define two conditions necessary for new device detection: both $\tilde{r}^{(k)}$ and $\tilde{p}^{(k)}$ should be smaller than the corresponding thresholds $\theta_R(\hat{y}^{(k)})$ and $\theta_P(\hat{y}^{(k)})$. It has to be noted here, that in general it is possible to detect a new device even when only one condition is satisfied. Moreover, a different logical relation can be used to combine two conditions, e.g., OR ($\vee$) instead of AND ($\wedge$). However, preliminary experiments have shown that our classifiers perform the best when both conditions are combined via a conjunction, therefore, we use the logical operator AND in all our experiments.

V. EXPERIMENTAL SETTING

We divide our dataset of wireless transmissions, generated by $|\aleph|$ known, in-library, devices and $|\aleph'|$ unseen, out-of-library, devices into training and test subsets, as described in Section II-B. Then, we slice transmission k into $n^{(k)}$ slices according to (1), using predefined slice size L and parameter λ (provided in the next subsection). Slices of transmissions in the training set are used as an input to train our deep learning models (AlexNet1D and ResNet1D). We use the resulting

trained model to detect whether a transmission in the test set is from a new device (out-of-library) or not using 3; in the latter case, we predict the (in-library) source of transmission via the majority rule. We expect the model to be capable of both correctly classifying signals transmitted by old devices, as well as precisely detecting if a transmission comes from a new device. To reach this goal, we test several combinations of statistics χ for each model, and report the one that achieves superior performance.

Parameters. For all experiments described in this paper, we set the slice size to $L = 512$ when processing raw WiFi transmissions with the ResNet1D model and $L = 198$ with the AlexNet1D model. We use $L = 198$ for equalized WiFi for both models, and $L = 1024$ for ADS-B transmissions in the AlexNet1D model and $L = 512$ in ResNet1D. We also set $\lambda = 10$, which means that each I/Q sample in the training subset has been seen 10 times, on average, given the randomness of the process. Models for each dataset are trained till convergence, i.e. until validation accuracy stagnates for 3 consecutive epochs. Finally, we use a batch size of 256 for all experiments.

Performance Evaluation. Given a current setting of old and new devices in the test set, we use five different metrics to reflect the ability of our trained models to correctly classify existing devices and detect new ones.

The first three metrics are designed to evaluate classifier's performance with respect to test transmissions by old, in-library devices. The first is the **O**ld device transmission **C**orrectly **C**lassified (OCC) ratio:

$$OCC = \frac{|\{k \mid \hat{y}^{(k)} = y^{(k)}, \hat{y}^{(k)}, y^{(k)} \in \aleph\}|}{K_{\aleph}}, \qquad (15)$$

capturing the accuracy of classification. The **O**ld device transmission **W**rongly classified as **I**n-library (OWI) ratio:

$$OWI = \frac{|\{k \mid \hat{y}^{(k)} \neq y^{(k)}, \hat{y}^{(k)}, y^{(k)} \in \aleph\}|}{K_{\aleph}}, \qquad (16)$$

captures signals that were correctly detected as old, but are attributed to the wrong source. The **O**ld device transmission **W**rongly classified as **O**ut-of-library (OWO) ratio:

$$OWO = \frac{|\{k \mid \hat{y}^{(k)} \neq y^{(k)}, \hat{y}^{(k)} \in \aleph', y^{(k)} \in \aleph\}|}{K_{\aleph}}, \qquad (17)$$

captures transmissions that are in fact due to old devices, but are wrongly detected as new. For OCC, OWI, and OWO, $k \in [1, K_{\aleph}]$, and $K_{\aleph} = \sum_{d \in \aleph} K_d$.

In addition, we compute two metrics for transmissions captured from new, out-of-library, devices: the **N**ew device transmission **C**orrectly **D**etected as transmitted by an unseen device (NCD) ratio:

$$NCD = \frac{|\{k \mid \hat{y}^{(k)} = y^{(k)}, \hat{y}^{(k)}, y^{(k)} \in \aleph'\}|}{K_{\aleph'}}, \qquad (18)$$

and the **N**ew device transmission **W**rongly **D**etected as transmitted by a known device (NWD), i.e.,

$$NWD = \frac{|\{k \mid \hat{y}^{(k)} \neq y^{(k)}, \hat{y}^{(k)} \in \aleph, y^{(k)} \in \aleph'\}|}{K_{\aleph'}}, \qquad (19)$$

where $k \in [1, K_{\aleph'}]$, and $K_{\aleph'} = \sum_{d \in \aleph'} K_d$.

Software. Data preprocessing (equalization) has been performed with GNU Radio Toolkit [23]. Classifiers, training and prediction are implemented in Python using Keras with TensorFlow backend and NVIDIA CUDA support.

Hardware. All experiments are carried out on an NVIDIA DGX-1 workstation with 4 Tesla V100 GPUs with 16 GB memory each and 512 GB RAM.

TABLE IV
DEVICE CLASSIFICATION ACCURACY WITHOUT NEW DEVICE DETECTION

#	AlexNet1D			ResNet1D		
Dev.	WiFi (raw)	WiFi (eq)	ADS-B	WiFi (raw)	WiFi (eq)	ADS-B
500	0.46	0.43	0.83	**0.61**	0.56	**0.90**
250	0.40	0.48	0.88	**0.63**	0.55	**0.90**
50	0.58	**0.64**	**0.92**	0.63	**0.64**	0.86

When new device detection is not used, accuracy coincides with OCC (15). Best results are highlighted in bold. For WiFi protocol we also compare raw (filtered) WiFi transmissions with their equalized versions.

VI. RESULTS

In this section, we discuss results obtained for our two deep neural network models, ResNet1D and AlexNet1D, on three datasets of different size for both WiFi and ADS-B protocols and different choices of statistics χ for thresholds $\theta_R(d)$ and $\theta_P(d)$ and transmission prediction probability $\tilde{p}^{(k)}$).

Accuracy without New Device Detection. In order to evaluate the influence of the out-of-library device detection procedure, we first present results on the same data featuring only in-library device classification. Results are summarized in Table IV and the best performance for each wireless protocol is highlighted in bold. As it can be observed, ResNet1D demonstrates superior performance in the majority of cases.

Performance with New Device Detection. The results from both AlexNet1D and ResNet1D models on three datasets (raw WiFi, equalized WiFi, and ADS-B) with $|\aleph| = 500$ in-library devices are depicted on Fig. 5. We report 5 different metrics (OCC, OWI, OWO, NCD, NWD) for each combination of transmission probability prediction $\tilde{p}^{(k)}$ and probability and ratio thresholds $\theta_P(d), \theta_R(d)$. However, in general, we aim to primarily maximize the model performance with respect to the correct new device detection and in-library device classification. Therefore, we focus on OCC and NCD below.

Optimal Statistics. As it can be noticed from Figure 5, the best performance with respect to OCC and NCD jointly, for both AlexNet1D and ResNet1D models, and all three data types, is achieved when the *lcb1* statistic is employed for threshold computing during training. However, the statistics used at test time to compute transmission prediction probability $\tilde{p}^{(k)}$ exhibit inconsistent behavior across datasets. This might be a result of much higher variance of the test data, probably due to the fact that for each new device only a single transmission was processed.

To define the best choice of a tuple $(\theta, \tilde{p}^{(k)})$, a combination of two statistics: the former θ is utilized to compute

Fig. 5. Performance of AlexNet1D (a,b,c) and ResNet1D (d,e,f) models for different combinations of statistics used for thresholds $\theta_P(d), \theta_R(d)$ and transmission prediction probability (TPP) $\tilde{p}^{(k)}$. Presented results are for datasets with $|\aleph| = 500$ in-library devices: WiFi dataset (raw transmissions (a,d) and their equalized versions (b,e)), and ADS-B dataset (c,f). Models are evaluated on five metrics: OCC, OWI, and OWO for old device transmissions, and NCD and NWD for new device transmissions.

ratio and probability thresholds $\theta_P(d), \theta_R(d)$ during training, and the latter is used at test time to compute transmission prediction probability, we compute a Pareto curve (Fig. 6) showing the ratio of correctly classified transmissions from old devices (OCC) and the ratio of transmission from new devices correctly detected as out-of-library (NCD). Again, we choose OCC and NCD as the metrics that we are most interested to optimize. Numerically, we represent the optimal tuple $(\theta, \tilde{p}^{(k)})$ as a pair that minimizes the distance to the output of the ideal classifier:

$$(\theta, \tilde{p}^{(k)})^* = \arg\min_{\theta, \tilde{p}^{(k)}} \sqrt{\left(\frac{acc - OCC}{acc}\right)^2 + (1 - NCD)^2}, \tag{20}$$

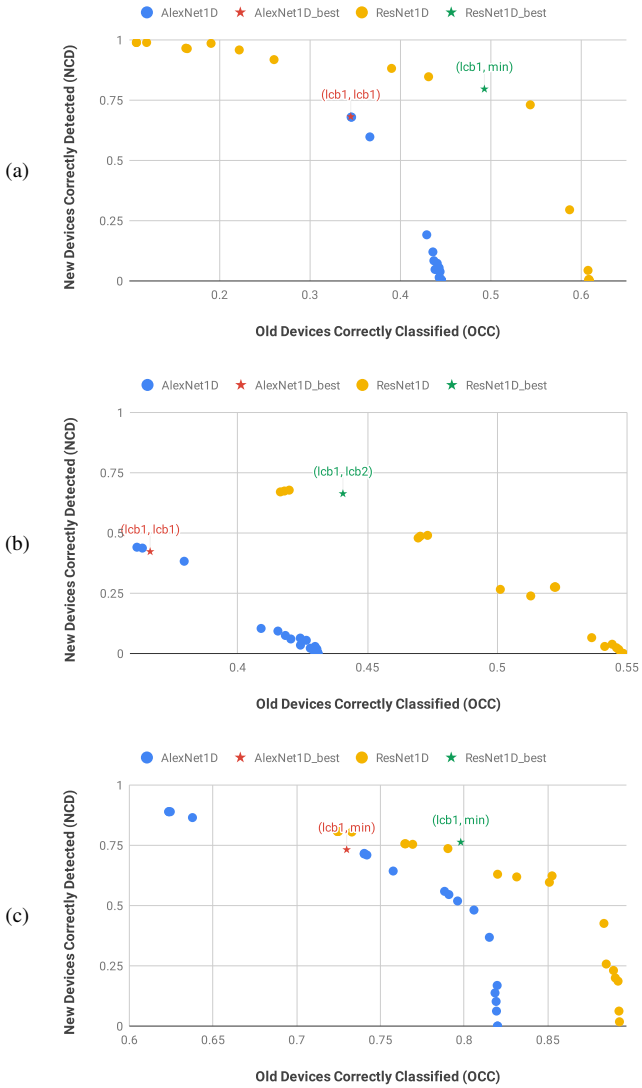


Fig. 6. OCC vs. NCD Pareto curves for WiFi (raw (a) and equalized (b)), and ADS-B (c) datasets with 500 in-library devices. Each point on the plots represents performance of a corresponding model achieved for a given combination of a thresholding method used during training to compute thresholds $\theta_P(d), \theta_R(d)$, and statistics used at test time to compute $\tilde{p}^{(k)}$.

where acc is the highest classification accuracy achieved for a given dataset, wireless protocol and classifier without implementing new device detection (see Table IV).

Performance Analysis. Table V summarizes performance of both AlexNet1D and ResNet1D models on all three datasets and both wireless protocols by reporting the optimal OCC/NCD pairs. In comparison to Table IV, we observe that incorporating new device detection results in a 12.5% decrease, on average, depending on the number of transmissions correctly classified with an old device label, regardless of the wireless protocol and initial performance. However, this moderate sacrifice of the in-library device classification accuracy is compensated with a rather precise new device detection ability: on average, 67.9% out of 542 unseen WiFi devices, and 75% out of 458 unseen ADS-B devices are correctly detected. Only once (AlexNet1D model trained on partially equalized WiFi dataset with 500 in-library devices) accuracy of the proposed new device detection method does not exceed 50% threshold, which is the expected performance of a random classifier.

TABLE V
DEVICE CLASSIFICATION ACCURACY WITH NEW DEVICE DETECTION

#	AlexNet1D			ResNet1D		
Dev.	WiFi (raw)	WiFi (eq)	ADS-B	WiFi (raw)	WiFi (eq)	ADS-B
500	0.35/0.68	0.36/0.44	0.73/0.73	**0.49/0.80**	0.44/0.66	**0.80/0.76**
250	0.30/0.73	0.40/0.65	**0.71/0.81**	**0.55/0.75**	0.45/0.61	0.70/0.78
50	0.39/0.77	**0.52/0.65**	**0.83/0.76**	**0.46/0.73**	0.53/0.57	0.77/0.70

For each model, accuracy is represented by two numbers: the first number being the ratio of transmissions captured from known devices that were correctly classified (OCC), and the second being the ratio of transmissions from unseen devices being correctly identified as such (NCD). Equation (20) is used to define the best results (highlighted in bold).

Influence of Equalization. Interestingly, we observed a phenomenon while analyzing results of raw WiFi transmissions and their equalized versions. Specifically, for all combinations of tuples $(\theta, \tilde{p}^{(k)})$, both ResNet1D and AlexNet1D models constantly show much lower novel device detection accuracy for preprocessed data, comparing to the original WiFi transmissions: 60% vs. 74%, on average. This can be clearly observed through the visual comparison of Fig. 6a and Fig. 6b. Meanwhile, it has to be mentioned, that percentage of transmissions from new devices correctly detected as such for unprocessed WiFi transmissions is on par with ADS-B transmissions (76%, on average).

Scalability. Additionally, we examine the scalability of the task, i.e. the influence of the size of the training set on the performance of the model. Surprisingly, there is no direct relation between the novel device detection accuracy and the number of known devices in the training set.

VII. CONCLUSION

In this paper, we propose a framework for a novel class detection in the domain of radio frequency fingerprinting. The core of the proposed method is to slice radio transmissions into smaller parts, compute prediction heuristics for slices, and then use this heuristic to infer class label of the original transmission. We tested our method on six real-world datasets of different size and transmission protocols. To the best of our knowledge, this is the first paper to describe experiments on novel device detection and achieve high accuracy on such big datasets (e.g., 500 known + 549 unseen devices for WiFi protocol).

Based on the analysis of the results, we conclude that the performance of the framework for new device detection (NCD) does not depend significantly on the number of in-library devices in the training set, achieving 74.3% accuracy for raw WiFi, and 75.8% for ADS-B, on average. Rather, it depends on the prediction ability of the original classifier. Though partial equalization of WiFi transmissions might improve device

prediction (OCC) in some cases, it always has negative impact on the novel device detection, attaining 59.9%, on average.

Results presented in this paper is the first attempt to address novel class detection in the domain of radio frequency transmissions, to the best of our knowledge. Statistical methods used to compute transmission prediction probability $\tilde{p}^{(k)}$, as well as probability and ratio thresholds $\theta_P(d)$ and $\theta_R(d)$, can be further refined to capture class boundaries in highly non-linear label space. Already, in this paper we show that simple statistical approaches are capable of attaining up to 75% in the 'new device correctly detected' metric (NCD) while giving only 8% drop in the 'old device correctly classified' metric (OCC), even without model retraining.

ACKNOWLEDGMENT

This work is supported by the Defense Advanced Research Projects Agency (DARPA) under the Radio-Frequency Machine Learning Systems (RFMLS) program contract N00164-18-R-WQ80 and partially supported by National Institutes of Health (NIH) grant NIH/NHLBI U01HL089856. We are grateful to Paul Tilghman, Esko Jaska, and Kunal Sankhe for their insightful comments and suggestions.

REFERENCES

[1] O. Ureten and N. Serinken, "Wireless security through RF fingerprinting," *Canadian Journal of Electrical and Computer Engineering*, vol. 32, no. 1, pp. 27–33, 2007.

[2] W. C. Suski II, M. A. Temple, M. J. Mendenhall, and R. F. Mills, "Radio frequency fingerprinting commercial communication devices to enhance electronic security," *International Journal of Electronic Security and Digital Forensics*, vol. 1, no. 3, pp. 301–322, 2008.

[3] V. Lakafosis, A. Traille, H. Lee, E. Gebara, and M. M. Tentzeris, "RF fingerprinting physical objects for anticounterfeiting applications," *IEEE Transactions on Microwave Theory and Techniques*, vol. 59, no. 2, pp. 504–514, 2011.

[4] Y. Shi and M. A. Jensen, "Improved radiometric identification of wireless devices using mimo transmission," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 4, pp. 1346–1354, Dec 2011.

[5] S. U. Rehman, K. W. Sowerby, and C. Coghill, "Analysis of impersonation attacks on systems using rf fingerprinting and low-end receivers," *Journal of Computer and System Sciences*, vol. 80, no. 3, pp. 591 – 601, 2014.

[6] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, 2008, pp. 116–127.

[7] S. U. Rehman, K. W. Sowerby, S. Alam, I. T. Ardekani, and D. Komosny, "Effect of channel impairments on radiometric fingerprinting," in *2015 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Dec 2015, pp. 415–420.

[8] T. D. Vo-Huu, T. D. Vo-Huu, and G. Noubir, "Fingerprinting Wi-Fi devices using software defined radios," in *Proceedings of the 9th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2016, pp. 3–14.

[9] K. Merchant, S. Revay, G. Stantchev, and B. Nousain, "Deep learning for RF device fingerprinting in cognitive communication networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 160–167, 2018.

[10] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury, "Deep learning convolutional neural networks for radio identification," *IEEE Consumer Electronics Magazine*, vol. 56, no. 9, pp. 146–152, 2018.

[11] K. Sankhe, M. Belgiovine, F. Zhou, S. Riyaz, S. Ioannidis, and K. Chowdhury, "ORACLE: Optimized Radio clAssification through Convolutional neuraL nEtworks," in *IEEE International Conference on Computer Communications*, 2019.

[12] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[13] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning-the good, the bad and the ugly," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4582–4591.

[14] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment, 2006, pp. 187–198.

[15] Y. Bengio, H. Larochelle, and P. Vincent, "Non-local manifold parzen windows," in *Advances in neural information processing systems*, 2006, pp. 115–122.

[16] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2002, pp. 15–27.

[17] V. Hautamaki, I. Karkkainen, and P. Franti, "Outlier detection using k-nearest neighbour graph," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 3. IEEE, 2004, pp. 430–433.

[18] J. Zhang and H. Wang, "Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance," *Knowledge and information systems*, vol. 10, no. 3, pp. 333–355, 2006.

[19] M. Markou and S. Singh, "Novelty detection: a reviewpart 2:: neural network based approaches," *Signal processing*, vol. 83, no. 12, pp. 2499–2521, 2003.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[21] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, vol. 9, 2010, pp. 249–256.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1097–1105.

[23] M. Müller. (2018, Aug.) Gnu radio v3.7.13.4 (press release). [Online]. Available: https://www.gnuradio.org/news/2018-07-15-gnu-radio-v3-7-13-4-release/