Understanding WiFi Cross-Technology Interference Detection in the Real World

Teemu Pulkkinen†*, Jukka K. Nurminen* and Petteri Nurmi*
†Ekahau Oy, Finland
*Department of Computer Science, University of Helsinki, Finland
†first.last@ekahau.com *first.[initial.]last@helsinki.fi

***Abstract*—WiFi networks are increasingly subjected to *cross-technology interference* with emerging IoT and even mobile communication solutions all crowding the 2.4 GHz ISM band where WiFi networks conventionally operate. Due to the diversity of interference sources, maintaining high level of network performance is becoming increasing difficult. Recently, *deep learning based interference detection* has been proposed as a potentially powerful way to identify sources of interference and to provide feedback on how to mitigate their effects. The performance of such approaches has been shown to be impressive in controlled evaluations; however, little information exists on how they generalize to the complexity of everyday environments. In this paper, we contribute by conducting a comprehensive performance evaluation of deep learning based interference detection. In our evaluation, we consider five orthogonal but complementary metrics: correctness, overfitting, robustness, efficiency, and interpretability. Our results show that, while deep learning indeed has excellent correctness (i.e., detection accuracy), it can be prone to noise in measurements (e.g., struggle when transmission power is dynamically adjusted) and suffers from poor interpretability. To compensate for weaknesses of deep learning, as our second contribution we propose a novel signal modeling approach for interference detection and compare it against the deep learning. Our results demonstrate that, in terms of errors, there are some differences across the two approaches, with signal modeling being better at identifying technologies that rely on frequency hopping or that have dynamic spectrum signatures but suffering in other cases. Based on our results, we draw guidelines for improving interference detection performance.**

I. Introduction

IEEE 802.11 technology, better known as WiFi, has achieved tremendous success and has become the de-facto solution for most communications. Most WiFi networks operate within the 2.4GHz ISM (industrial, scientific and medical) frequency band that is fast becoming increasing crowded, making WiFi prone for cross-technology interference. Indeed, besides traditional sources of interference – such as microwave ovens and proprietary short-range wireless technologies used in baby monitors, gaming consoles, video cameras, and so forth – new devices that operate within the ISM band are continually emerging. For example, most IoT systems, including smart homes and smart industries, operate using ISM frequency for communication and emerging mobile technologies, such as LTE-U, take advantage of ISM bands to increase capacity. Addressing interference is critical particularly for enterprise settings where it can result in bandwidth issues and hamper performance of operation-critical systems.

The increased significance of ISM band interference has fueled both academic and commercial interest in developing methods for identifying and classifying sources of cross-technology interference. For example, Airshark uses features extracted from spectrum sweeps to train a classifier that can identify different types of interference sources [1] and recently several solutions that take advantage of deep learning have been proposed [2]–[4]. Simultaneously, a wide range of commercial systems have emerged, such as Cisco CleanAir [5], AirMagnet Spectrum XT [6], and netAlly EtherScope nXG [7]. Besides IEEE 802.11, interference detection has also been widely explored in emerging IoT environments [8]–[13].

Commercial interference detection systems need to be capable of operating robustly across diverse environments, and often with little or no domain knowledge. This can be highly challenging as the RF environments often contain considerable variation across different deployment sites [14]. Despite the significant commercial and academic interest in interference detection, currently little information exists on how well interference detection techniques generalise to the complexity of everyday environments. Indeed, existing research has largely focused on evaluating performance in individual test environments [1]–[3] without systematically assessing how differences in training and testing environments affect the performance of these techniques.

In this paper, we contribute by conducting a comprehensive performance evaluation of deep learning based interference detection. In our evaluation, we consider five orthogonal but complementary metrics: correctness, overfitting, robustness, efficiency, and interpretability. We carry out our evaluation using measurements from two environments, a RF-shielded enclosure with minimal external interference, and a typical office environment. We systematically investigate effects of environmental variations by injecting noise into the measurements and measuring performance variations. Our method offers us a controlled way to assess how similar training and testing environments need to be for the system to operate robustly. Our results show that, while deep learning indeed has excellent correctness (i.e., detection accuracy), it can be prone to noise in measurements (e.g., struggle when transmission power is dynamically adjusted) and suffers from poor interpretability. To compensate for weaknesses of deep learning, as our second contribution we propose a novel signal modeling approach for interference detection and compare it against the deep learning. Our results demonstrate that, in terms of errors, there are some differences across the two approaches, with signal modeling being better at identifying technologies that rely on frequency hopping or that have dynamic spectrum signatures but suffering in other cases. Based on our results, we draw guidelines for improving interference detection performance.

Summary of Contributions

- **Systematic evaluation.** We critically assess deep learning based interference detection using measurements from a noise-free environment and a typical everyday environment using five different criteria.
- **Novel insights.** By systematically injecting noise into the measurements, we evaluate how resilient deep learning based approaches are, demonstrating that they can easily break as the environments differ.
- **Novel approach.** We develop a signal modeling based interference detection technique, which offers better robustness than deep learning, and provides more interpretable results with only slightly decreased accuracy.
- **Novel Guidelines**. Based on our results, we draw guidelines for the use of different interference techniques and their use in commercial deployments.

II. Measurement Setup

We evaluate the effectiveness of deep learning based interference detection using extensive measurements collected from two environments: a controlled testbed with minimal external interference and a typical office environment. Our testbed was carefully crafted to ensure we can control the interference sources and avoid external artefacts influencing the measurements. We next describe our measurements, the overall measurement setup, and the devices that were used to create interference.

Measurement Setup. We collected measurements in two settings: a testbed where external interference was minimized, and an office environment. Our testbed environment consists of a Ramsey Electronics STE3600 RF shielded enclosure, illustrated in Fig. 1. The use of RF shielded enclosure ensured control over the radio spectrum and allowed minimizing noise for the interference measurements. In our experiments, we systematically inject noise into the measurements to assess how differences in testing and training environments affect interference detection performance. The controlled setup ensures we can better separate the effects of interference and noise. Our second environment consists of a typical office environment. In both setups the sources of interference were kept 1 meter apart from the spectrum analyzer to ensure consistent RF decay while having differing noise characteristics.

Sources of Interference. We consider eight different sources of interference, consisting of seven everyday devices and a baseline consisting of background noise. The devices considered in the evaluation were chosen as representative examples of common everyday objects that cover different types of connectivity patterns. Specifically, we consider devices with constant frequency and those performing frequency hopping,

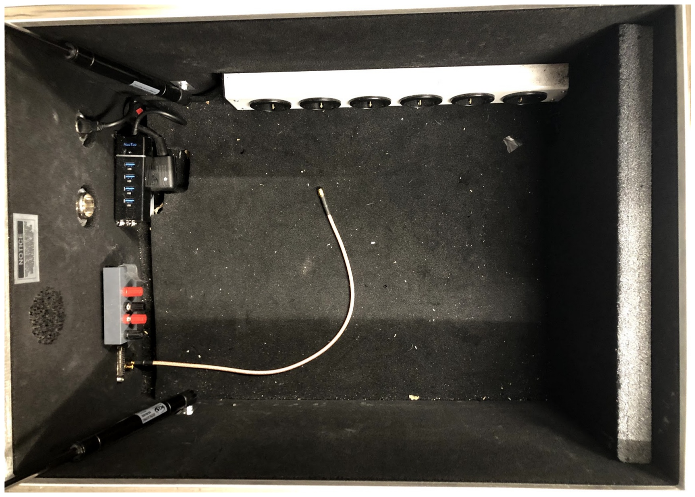

Fig. 1. RF shielded enclosure where training set measurements were performed.

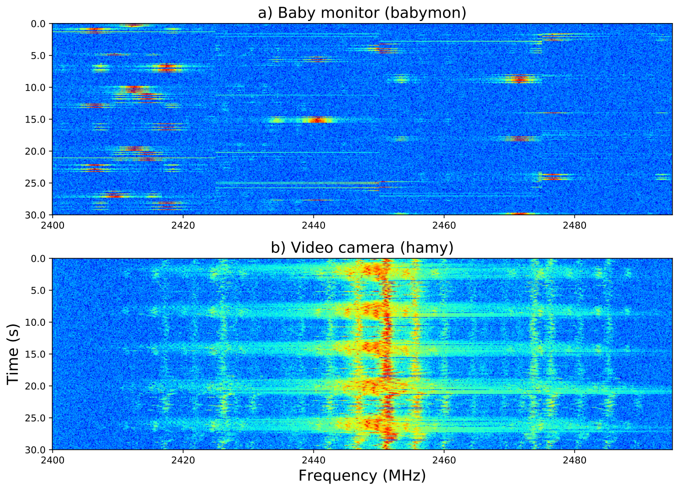

Fig. 2. Spectrogram of training data for two of the target devices.

and devices having constant communication frequency and those that only transmit periodically. The devices considered in the experiments are summarized in Table I. We specifically target everyday devices with limited networking functionality as these are most likely to have least support for interference avoidance built-in. For example, BLE has built-in adaptive frequency hopping support which has been designed to mitigate interference [15], [16]. Thus, the sources of interference in our setup correspond to a bad case scenario where the interference would have the most significant effect on WiFi performance.

Data description. As data we consider spectrum sweeps collected over the 2.4 GHz frequency band. The measurements were taken using the Ekahau Sidekick spectrum analyzer [17]. As sampling resolution we use 39 kHz, which covers the relevant portion of the 2.4 GHz band. This results in 2500 spectrum bins per sweep, each containing a power value in dBm, ranging between -100 dBm and -20 dBm. The samples are taken at 25Hz frequency, and in the comparison we use

TABLE I
DEVICES USED FOR STUDY

Identifier	Description	Transmitter type
babymon	Baby monitor	Frequency hopping
motorola	Baby monitor	Frequency hopping
huhd	Headset	Frequency hopping
boya	Lapel microphone	Frequency hopping
hamy	Analog video camera	Continuous, fixed frequency
skatco	Intercom	Continuous, fixed frequency
rc	RC transmitter	Periodic, fixed frequency
none	No device	Background noise

data frames of 1 second, i.e., the measurements are matrices of size 25 x 2500. Fig. 2 illustrates the measurements for two sources of interference. The sweeps were labeled according to the device that was currently transmitting, and we collected in total 1500 spectrum sweeps per class. The background noise model was constructed similarly, but having all sources of interference turned off.

III. Measures for Interference Detection

Commercial viability of interference detection techniques requires robust performance across different everyday environments with potentially limited knowledge of the deployment domain. At the same time, simply focusing on detection accuracy is not sufficient as network administrators need to be able to identify and isolate sources of interference. In the following we introduce measures for comprehensively assessing interference detection techniques, including their generality and usability for network administrators. Our metrics are inspired by measures designed for testing machine learning algorithms [18], which we have adapted to suit interference detection techniques.

A. Correctness

Correctness is the most common metric for machine learning testing, and the primary measure that has been used for assessing interference detection performance. Correctness measures how well an algorithm's predictions match reality, which in the context of interference detection entails comparing the predicted class labels (i.e. detected devices) with known (*ground truth*) data labels. Formally, correctness is defined as the ratio

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{1}(h(x_i) = y_i), \tag{1}$$

where $\mathbb{1}$ is the indicator function returning 1 for true statements and 0 for false, m is the number of data samples, x_i are the (unknown) data labels, $h(x_i)$ the algorithms predictions of those labels and y_i the known actual labels.

B. Overfitting

Overfitting measures tendency to emphasize fit to training data, usually to the detriment of unseen cases. This may happen because the model has learned to fit against the noise in the data by becoming too complex [18], which results in poor generalization performance. In the context of interference detection, overfitting means that the system performs well in testing, but poorly when it is being adopted in other setups.

We assess overfitting using *Perturbed Model Validation* (PMV) [19], which operates by injecting noise into samples used for training the model used by the interference detection approach. The intuition is that overfitted models can only see a small decrease in accuracy because they are able to fit to the noise in the data. A larger change suggests the model is fitting to the actual properties of the domain at hand instead of fitting the model to the noise. We use the so-called *PMV Accuracy Decrease Rate*, which is represented by the absolute value of the gradient of the best fitting – in the least squares sense – line formed by points (r_i, S_{ri}), where r_i is the ratio of noise in the training set and S_{ri} is the accuracy of the re-trained model on the perturbed dataset. An ideal model is one with a large gradient; i.e., one where perturbing the training sets quickly causes a decrease in training accuracy. Our implementation of PMV follows an adversarial approach where a fraction of the data labels are permuted into other labels [20].

C. Robustness

Robustness refers to the algorithm's capacity to handle noise. Whereas overfitting measures how much the model fits on noise, robustness examines how it fares in testing when noise is injected. In the simplest instance, testing robustness entails injecting noise to test samples and remeasuring correctness. We measure robustness by adopting a measure originally proposed for image recognition [21]. The intuition in the measure is to measure two aspects: *pointwise robustness*, which refers to the smallest level of noise where the solution fails; and *adversarial frequency*, which refers to the rate at which the approach fails when input is perturbed. For estimating pointwise robustness, we add Gaussian noise of increasing standard deviation until labels change, and for estimating adversarial frequency we then proceed to measure how many labels are changed at this threshold. We use zero-mean Gaussian with varying levels of standard deviation for noise injection as this approximates additive white Gaussian noise (AWGN), which is a widely used noise model for information channels [22]

D. Efficiency

We consider two complementary measures of efficiency: *information efficiency* which measures how the amount of training data affects interference detection performance; and *timing efficiency* which measures the difference in re-training a model and running predictions as a number of classes (i.e., types of interference sources) that are being modeled. Information efficiency is estimated by selecting a subset of the training data and measuring how accuracy changes as the number of training samples is reduced. As loss-based subset selection is used by deep learning for optimizing model parameters, we instead use random subset selection to have a fair baseline, in line with current best practice [23].

E. Interpretability

Interpretability is concerned with the transparency of the model and its capability to offer post-hoc explanations. Interpretability is essential for interference detection as it offers network administrators insights into the possible type of device that is causing disturbances, including the frequency band that is most affected. Evaluating interpretability of individual models, however, is not meaningful in isolation and hence we adopt a scenario-based approach where interpretability is assessed by looking at classification results in a confounding case where two different devices are transmitting simultaneously. We also compare predictions of interference detection model with a theoretical model of signal propagation.

IV. Deep Learning-Based Interference Detection

Deep learning models have recently emerged as powerful solutions for interference detection and classification [2]–[4]. In this section we examine how well deep learning solutions support everyday environments by critically evaluating them using the measurements described in Sec. II and the measures described in Sec. III.

A. Network Structure

Kulin et al. [3] showed that a spectrum sweep (FFT) based representation offers best performance, particularly for models using convolutional layers as the interference signatures can be easily identified from them, which in turn simplifies that feature learning performed by the model. Motivated by this result, we consider a CNN-based interference detection and classification approach. The structure of our model in shown in Fig. 3 and has been adapted from our previous work [4]. Our previous work focused on using CNN's for alleviating re-training on new devices, and in this work we consider a simplified model that incorporates only the interference detection and classification parts of the model without considering device adaptation. In this model, classification is performed in a dense softmax layer which has one node per device (interference source) and which uses the magnitude of the values to determine the final classification result. Our model was implemented using Python TensorFlow, keras and NumPy. The model uses non-overlapping windows of 25 sweeps as input, a stochastic gradient descent (SGD) learning rate of 0.001, a batch size of 1, a stride and max pool size of 2x2 and training over 100 epochs. These parameters were based on previous results for the model [4] which showed high performance. As the focus of our work is on evaluating suitability of deep learning for everyday setups, we did not perform additional extensive parameter tuning to try to maximize model performance.

In both training and testing, all samples were Z-score normalized using background measurements (i.e., where no device was turned on). Formally, each sample $s - S$ was normalized by

$$s_Z = \frac{s - \mu_{none}}{\sigma_{none}}. \quad (2)$$

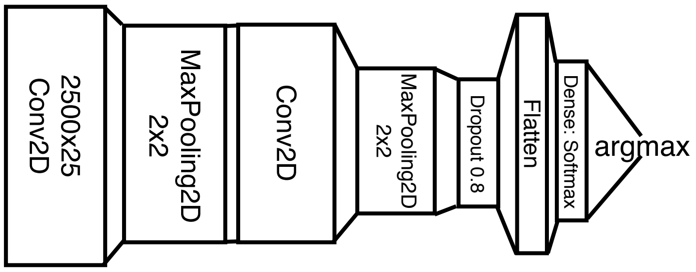

Fig. 3. Interference detection CNN architecture

For our data, the corresponding values are $\mu_{none} \approx -101.5$ dBm and $\sigma_{none} \approx 5.88$ dB. For testing, we fix the random seed before training and testing the model to ensure variance in behavior can be attributed to our experimental setup, and to ensure reproducible results.

B. Evaluation

We first motivate the use of a deep-learning approach by validating its accuracy. Focusing only on a single quantifiable metric like accuracy, however, can lead further development astray if other aspects of the solution are not properly considered. We thus further show that this approach can face severe issues in terms of robustness and efficiency even though the initial estimated accuracy is satisfactory. We postpone discussion about overfitting until the next section where we compare deep learning against a signal modeling approach that we propose.

Accuracy Accuracy was measured by training the algorithm on measurements from the RF shielded testbed and performing predictions based on measurements from the office environment. The results of this evaluation are shown in the form of a confusion matrix in Table II. Total accuracy was calculated as the proportion of the entries located in the diagonal of the matrix, which corresponds to the formulation in III-A. The overall accuracy of the our deep-learning approach was 80%, which is in line with previous evaluations [2]–[4]. The sole exception is the headset (huhd) which is often confused with the intercom (skatco) despite having different connectivity parameters. This suggests that the performance of deep learning can degrade when the environments differ, particularly if the frequency signatures of the devices are unambiguous.

Robustness We next show how the algorithm performs when the noise in the environment increases – a reasonable assumption of any real-world application – we inject synthetic noise into dataset as described in Sec. III-C. We inject test data with noise that has increasing standard deviation in the range $[0, 20]$ dB, in steps of 0.5 dB. This testing mechanic is similar to [3], where the testing accuracy's sensitivity to noise was measured w.r.t. to the signal-to-noise ratio (SNR) in a range from -20 to 20 dB. However, whereas Kulin et al. [3] perturbed measurements collected from a single environment, our data has been collected from two distinct environments and thus offers better insights into cross-environment performance of deep learning. At each iteration, new predictions were

TABLE II

CORRECTNESS: CONFUSION MATRIX FOR PREDICTIONS WITH CNN. ACCURACY: 0.80

truth/predicted	babymon	boya	hamy	huhd	motorola	skatco	rc	none
babymon	19	0	1	3	4	0	3	0
boya	1	29	0	0	0	0	0	0
hamy	1	0	29	0	0	0	0	0
huhd	0	0	0	29	0	1	0	0
motorola	0	0	0	0	30	0	0	0
skatco	0	2	0	21	0	7	0	0
rc	0	9	0	0	0	0	21	0
none	0	0	0	0	0	0	0	30

made based on the perturbed test set and all predicted classes were recorded. To calculate a normalized score, each set of predictions was compared to the initial set (containing no injected noise) and the number of disagreeing labels was summed and scaled by the known number of test samples (240). The results are displayed in Fig. 4.

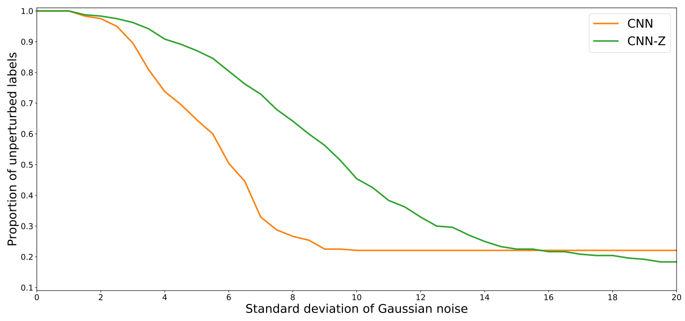

Fig. 4. **Robustness:** Proportion of changed labels with respect to increased SD σ of zero-mean Gaussian noise. First point of failure: $\sigma = 1.5dB$

In terms of pointwise robustness (first point of failure), CNN fails already at very modest noise (standard deviation of 1.5 dB) changing 4 labels. Thereafter robustness (in terms of adversarial frequency) decreases rapidly, reaching zero at $\sigma =$ 10 dB, after which it only predicts the same label for the rest of the iterations. Part of the reason for the poor robustness of CNN is the fact that changes in domain can affect the range of values, making the normalisation of measurements ineffective and prone to failure. This is highlighted by the CNN-Z line in the figure, which shows results when values were re-normalized after each noise injection. However, overcoming this in practical deployments would require constant sampling and recalculation of the normalization parameters, as well as to establish a mapping across training and testing domains. In practice, this is difficult to perform reliably as outliers or changes in the environment could affect the required parameter values. To put our results into context, fading factors used in propagation models vary between $5 - 14$ dB [24], suggesting that model performance should not decrease significantly with noise levels below 9 dB as this is the expected range of variation for measurements collected from different everyday environments. In theory, performance should generalise as long as training and testing environments are sufficiently similar. However, this is difficult to guarantee in practice. As an example, office environments can experience over 10 dB variations within the same floor, depending on the building materials, furniture and material used for partitioning the space [24]. Re-normalization can improve robustness, but also starts to suffer from $6 - 7$ dB onward.

Efficiency As described in Sec. III, we measure learning efficiency using random subset sampling and measuring performance of the deep learning algorithm with datasets where sample size had been reduced. Since our measurements contained 30 measurements per device, we evaluated the performance of the approach by progressively removing more samples until only one sample was used for training per device. For each sample removal count, we repeated the experiment 30 times removing different randomly chosen subsets of measurements. Thus, in total we performed 870 trials (30 trials per removal level and 29 different removal counts).

The results of the evaluation are shown in Fig. 5. Even with small sample removal counts, the performance of the deep learning model fluctuates considerably, suggesting that the model would require more training samples to be able to train a robust model. Indeed, accuracy can be as low as 30% when just a single sample is removed. Conversely, the model can occasionally reach high performance even when little data is available. For example, the best case performance is close to 80% even when only 7 measurements are used. Thus, data quantity alone does not govern performance as quality of data also clearly plays a significant role. In practice, this means that the deep learning model would either require considerable amounts of training data or an effective subset selection strategy that can (correctly) identify the most informative training measurements. Neither is easy to achieve, as training data collection is highly laborious – and even impossible in many enterprise settings – whereas the effectiveness of subset selection techniques is difficult to guarantee ex-ante as the complex structure of deep learning models makes it difficult to understand effects of individual data samples.

V. SIGNAL MODELING-BASED INTERFERENCE DETECTION

Thus far we have shown that deep learning can indeed reach good interference detection performance, but the underlying models are highly sensitive to noise and amount of training data. In practice these are significant issues, particularly for commercial enterprise detection systems, which must operate with limited knowledge of the deployment domains. To mit-

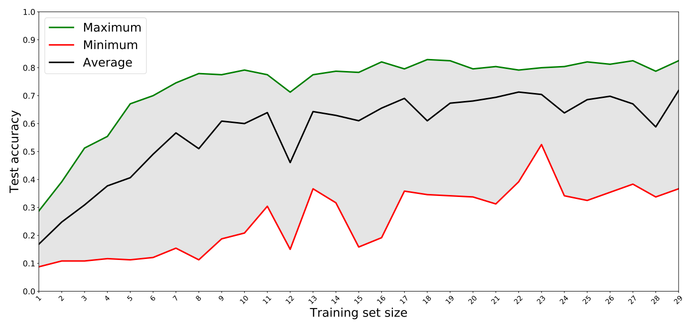

Fig. 5. **Efficiency:** Testing accuracy as number of samples are added to training set. Summary statistics from 30 trials of randomly sampled labels.

igate these issues, we next describe a novel signal modeling based approach for interference detection and classification.

A. Signal Modeling using Non-Negative Multiple Regression

Our approach models interference using non-negative multiple regression, which allows describing all possible interfering devices through their frequency-specific spectrum contribution. In our approach, the spectrum signature of the signal model is estimated by averaging relevant spectrum sweeps over time in the milliwatt domain and then scaling values by the strongest spectrum bin. Specifically, for each possible interferer, we define a vector $v = [s1, s2, ..., sn]$, where n is the number of bins in a spectrum sweep, and each s_i contains a value between 0 and 1. Fig. 6 shows example signatures created out of spectrum samples in Fig. 2.

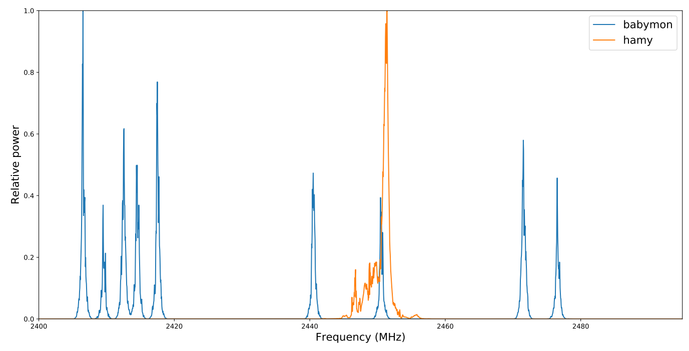

Fig. 6. Device spectrum signatures for the spectrum samples in Fig. 2.

These signatures are collected into a *design matrix* X, which acts the model for interference sources over the specific frequency band. Using standard least squares, given a new spectrum sweep y (specifically, the average spectrum sweep over the measurement window) we can solve the weights for each source of interference simply using

$$w = (X^T X) \backslash (X^T y). \qquad (3)$$

The fitted weights w_i can be interpreted as the estimated power of device i, which can then be used to find the strongest – in our application the most likely – source of interference. In practice, devices cannot transmit with negative range, so we modify the equation into a non-negative least squares (NNLS) form which forces the solution to have only positive weights.

B. Comparison to Deep Learning

Our signal modeling approach has designed as a method that alleviates and overcomes issues deep learning has while generalising to different types of environments, rather than as an approach that would provide superior interference detection performance. We next demonstrate that our approach indeed improves robustness against noise, model interpretability, and information efficiency in terms of amount of training data is needed. To accomplish this, we repeat our evaluation and compare our approach against the deep learning approach considered in the previous section.

Accuracy The classification accuracy of NNLS is summarized in Table III. At an overall accuracy of 73%, this is similar to the accuracy of CNN, though CNN performs slightly better. On average, this difference is somewhat smaller than this specific scenario because of the stochastic nature of the training process (as discussed in Sec. IV-B). Nevertheless, both approaches clearly suffer from the noisy real-world measurements used for testing.

Comparing the predicted classes to the true labels, the approaches agreed on the true class in 62.5% of cases . This suggests that in about 10% of the cases one of the algorithms succeeded where the other did not. In 5 test cases the algorithms agreed on the predicted class but disagreed with the true label. Part of the reason for disagreement might be down to training and testing labels, which label the entire spectrogram as one device even if the device is not transmitting constantly for the entire duration. In three of those cases, the approaches predicted *rc* when the true label was *babymon*. This understandable confusion stems from the fact that the *rc* device only transmits periodically on a fixed frequency, while *babymon* performs frequency hopping (see Fig. 2). At suitable time frames, their spectrum samples can seem similar.

Overfitting We assessed overfitting by implementing perturbed model validation (PMV, see Sec. III-B). In our implementation, we permute 1 to 15 labels per class, which corresponds to noise ratios r_i in the range of $[0.0333..., 0.5]$ following terminology of Zhang et al. [19]. Both algorithms used the same perturbed training labels to ensure comparison was fair. Additionally, we elected to fix the random seed between each training session to ensure any change in training accuracy could be attributed uniquely to the injected noise and not the underlying stochastic gradient descent.

We calculate PMV decrease rate as the absolute polynomial coefficient of the best fitting line, which was determined through simple least squares. Results from this evaluation are shown in Fig. 7. Both algorithms seem to handle overfitting quite well, clearly decreasing in accuracy as more noise is injected. The rates, as described in the original paper [19], are mostly interesting in a relative sense as what constitutes a good fit depends on application context. Nevertheless, rates above 0.5 typically indicate that the algorithms have no significant issues with overfitting.

TABLE III

Correctness: Confusion matrix for predictions with NNLS. Accuracy: 0.73

truth/predicted	babymon	boya	hamy	huhd	motorola	skatco	rc	none
babymon	12	3	7	0	0	0	4	4
boya	0	21	0	0	0	0	0	9
hamy	0	0	30	0	0	0	0	0
huhd	0	0	0	29	0	1	0	0
motorola	4	1	1	0	13	0	2	9
skatco	0	3	0	1	1	19	0	6
rc	0	0	0	0	0	0	22	8
none	0	0	0	0	0	1	0	29

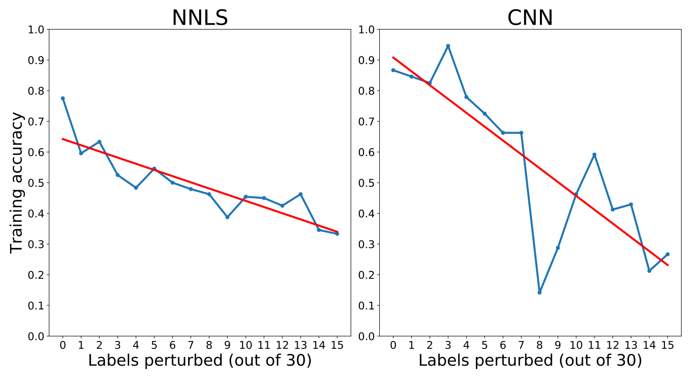

Fig. 7. **Overfitting:** PMV accuracy decrease rate curves (and best fitting line) calculated for NNLS and CNN, with rates ($\hat{k}_s$) of 0.60 and 1.35, respectively.

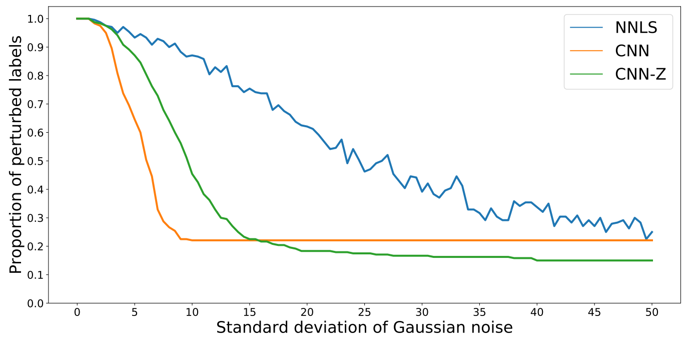

Fig. 8. **Robustness:** Proportion of changed labels with respect to increased standard deviation of zero-mean Gaussian noise. First point of failure: $\sigma = 1.5dB$

CNN clearly has the better accuracy decrease rate in our case (CNN: $\hat{k}_s = 1.35$, NNLS: $\hat{k}_s = 0.60$). It is possible that parameters of the CNN that would cause overfitting have been overcome during the initial construction, where correctness has been the metric for tuning them. Nevertheless, this calculation shows that both of our implementations are reasonable, at least in terms of overfitting.

Robustness The first clear difference between interference detection methods is apparent when measuring robustness. The previous noise injection procedure was repeated, but extended up to 50 dB to cover the full range of the NNLS behavior. Whereas CNN was unable to perform prediction after noise levels reached 10 dB, NNLS still provides non-random predictions at noise levels above 20 dB.

Though NNLS is better in an absolute sense, it suffers from a larger amount of variance in its robustness. This can likely be attributed to the scaling of the signatures which means a large importance is given to individual spectrum bins for each signature. This is exacerbated by the fact that signatures reside in linear instead of logarithmic space; the relative difference between strong and weak parts of the spectrum signature can differ on the order of magnitudes. Because of this, random noise can occasionally make an irrelevant device seem like a better candidate simply because power was attributed to a location that strongly corresponds with the specific device signature. However, a simple averaging over time can mitigate some of these variations.

Efficiency In stark contrast to the CNN, NNLS achieves an accuracy of almost 70% with only one carefully selected sample per device. It reaches its optimal accuracy with only 5 samples in the best case. In addition to providing a more stable performance over permutations, it also converges to its optimum level of performance faster.

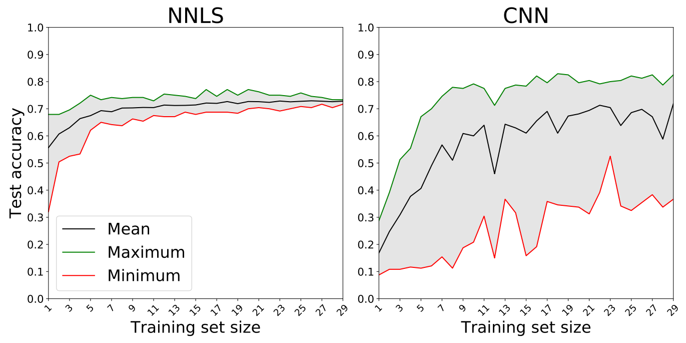

Fig. 9. **Efficiency (Information):** Testing accuracy as number of samples are added to training set. Summary statistics from 30 trials of randomly sampled labels.

In terms of time, CNN required approximately 400-500 ms for the entire testing set on a 2015 MacBook Pro[1], or roughly 2 ms per testing sample. NNLS performed prediction in 70-80 ms for the whole testing procedure meaning it could predict at a rate of 0.3 ms per sample. Though NNLS was clearly faster, it is important to note that the measurement device sampled

[1]CPU: 2.5 GHz Intel Core i7. RAM: 16 GB 1600 MHz DD3. Graphics: AMD Radeon R9 M370X 2 GB.

the spectrum at a speed of 40 ms per sweep, meaning both approaches are well within the limits of the input data rate even if they were to base their prediction on a single sweep of spectrum (let alone the 25 in our experimental setup).

For training, CNN used roughly 5-120 seconds to iterate over 100 epochs, depending on the sample size used. Since the number of training epochs needed depends on the training set, parameters and optimization algorithm, this is reasonable as training time can be amortized in real world applications and is usually a one-time cost that can be performed independently of the application. NNLS performed training in 1-30 ms, which can be attributed to its use of non-iterative matrix operations.

To understand how the algorithms perform as classes are added to the models, an experiment was performed where the original training and testing labels were replaced with labels for an increasing amount of classes. In other words, the first training/testing iteration used only 2 labels (i.e. binary classification) for the same set of training and testing data as used in other experiments. The number of classes was increased until it reached a total of 102, giving a range of 100 added classes. For CNN, re-training involved reconfiguring the final dense layer to contain the tested number of classes. The number of epochs was restricted to 10 because accuracy was not a concern in this setup and we were concerned with the increase in relative time used, specifically.

For each training and testing, the time was recorded and normalized by dividing it by the time used for binary classification (first iteration). This allowed for a fair comparison of the tendency of the algorithm performance as the number of learned and predicted classes increased. In addition, we found that NNLS initialization caused training to be slower for the first few iterations, so calculation was primed by running 5 iterations before the actual testing proceeded.

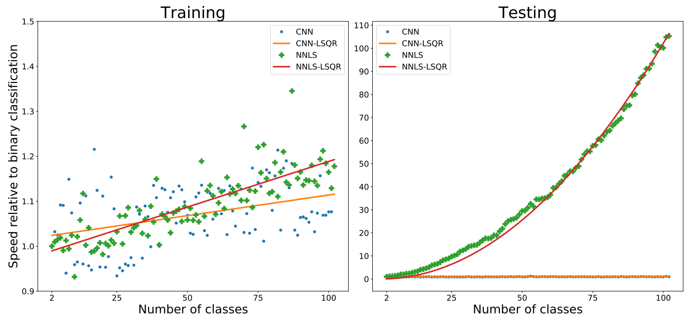

Fig. 10. **Efficiency (Performance):** Relative time spent in training and testing as number of classes increases. Best fitting curve for NNLS testing is $\approx 0.01n^2$

The results of this test are displayed in Fig. 10. Both algorithms have a similar linear increase in training time as the number of classes increases, with NNLS increasing slightly faster. As described earlier, however, NNLS has a faster absolute training time. The major difference between algorithms is apparent in testing performance. Whereas CNN is essentially not impacted by the number of the classes in the prediction phase, NNLS displays a time complexity of $O(n^2)$. Specifically, the best fitting curve for the testing time points is $0.01n^2$. Solving for a desired speed of 1 Hz (a reasonable requirement for an interactive application), we find that on the tested hardware NNLS could perform prediction with 900 classes before crossing this threshold. Though this would cover a broad range of devices – especially since many will display the same transmitting patterns – it is one potential limiting factor of this approach.

Interpretability To test interpretability, the algorithms were applied to a new set of test data (n=30) where devices *hamy* and *huhd* were transmitting concurrently. The intuition here was that the final weight vector of a highly interpretable model should reflect this multi-label scenario even it has never been trained on a combination of devices.

To establish a CNN model with multi-label classification capabilities, another configuration was trained where the last layer used sigmoid activation, loss was measured with binary cross-entropy and optimization was performed with Adam. This allows the CNN to predict more than one label per test sample. Training otherwise proceeded in the same way as before. As a form of comparison, the softmax-activated CNN was also allowed to predict labels for the new test cases. The results are displayed in Fig. 11. Weights for NNLS were scaled by the largest detected power to bring them into the same [0,1] range as the CNN approaches.

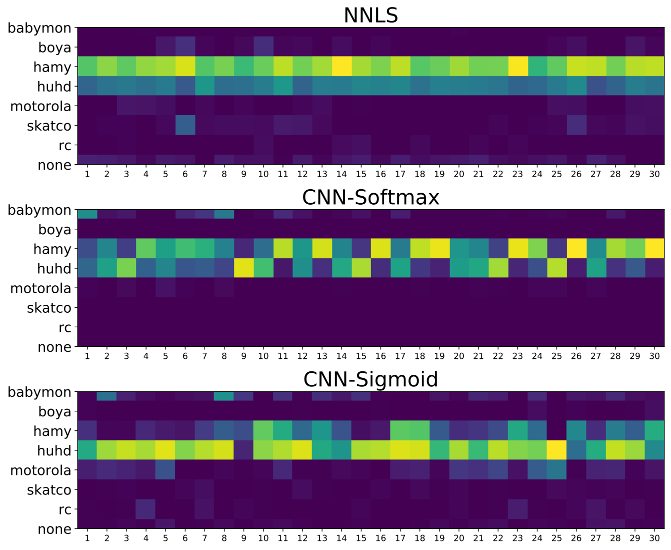

Fig. 11. **Interpretability (Multi-class):** Heatmaps of weight vectors for classification results when two devices (*hamy, huhd*) are transmitting concurrently.

It is apparent that NNLS more smoothly predicts both classes concurrently, albeit *huhd* with less distinction. This is likely due to its frequency hopping nature, transmitting less power at any one instance than the continuous transmitter *hamy*. Interestingly, the sigmoid version of CNN is able to detect *huhd* more consistently than *hamy*.

Because CNN-Softmax has been trained to classify one label at a time it struggles with multi-label scenarios. Its fluctuation between the two true labels is relatively consistent, however, whereas CNN-Sigmoid is not able to detect *hamy* to any discernable degree in the early samples. In our interpretation, this behavior – though not as robust as NNLS – is more easily managed in an end user application. A smoothing of labels over time, for instance, could easily perform to a necessary degree of accuracy.

Finally, a set of measurements were made while moving towards and away from the *hamy* device, in order to determine the correlation between predicted weights and the distance from – and by proxy the transmit power of – the target device. The results are displayed in Fig. 12. Note that the distance along X-axis turns back around the mid point. The measurements were performed in this manner to ensure that the results were robust with respect to the orientation of the measurement device, in terms of antenna directionality or polarization, for instance.

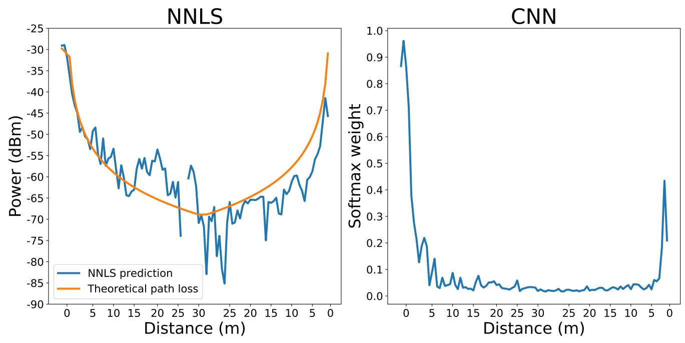

Fig. 12. **Interpretability (Distance):** Relationship between predicted weight and distance from interfering device (*hamy*). Note the travelled path loops back around the mid point.

Both algorithms have a clear tendency to recognise when they are close to the target device, but CNN's confidence in its prediction falls rapidly, reaching 10% after less than 5 meters. In other words, the only real-world interpretation that the softmax weight allows for is a vague measure of proximity. Due to the domain-aware configuration of the NNLS algorithm, however, the predicted weight can be interpreted as the target device's transmit power. This allows for a smoother estimate over distance, as well as a good fit to the theoretical power level. This real-world interpretation of the prediction result means that not only is thresholding a detection instance to a specific power level easier, it also allows for a more principled approach to locating the interferer in the target environment since the detected power level can be assumed to follow the standard distance-based path loss formulation.

Summary of Comparison Table IV summarises the results of the comparison. Deep learning has slightly better performance than the signal modeling approach, which is mostly due to the signal modeling failing to recognize devices in some cases. In terms of overfitting, the approaches are similar. However, the signal modeling approach is more resilient against noise (better robustness) and offers significantly better interpretability. Deep learning has slightly better runtime complexity, but is in turn highly sensitive to the amount of data that is available for training. Thus, the signal modeling approach provides a competitive alternative to deep learning based techniques, particularly for enterprise environments where collecting extensive amounts of training measurements is difficult or even next to impossible.

TABLE IV
CONCLUSION: RESULTS OF COMPARISON BETWEEN DEEP LEARNING (**CNN**) AND LINEAR REGRESSION (**NNLS**).

Metric	**Winner**	**Notes**
Correctness	**CNN**	NNLS is close but has blind spots [4]
Overfitting	**Tie**	CNN has better absolute score
Robustness	**NNLS**	CNN struggles with high noise values
Efficiency (Info.)	**NNLS**	CNN especially sensitive to data size
Efficiency (Time)	**CNN**	NNLS has high prediction complexity
Interpretability	**NNLS**	NNLS conforms to theoretical model

C. *Recommendations*

Based on our results, we can draw the following guidelines for interference detection techniques:

1) Deep-learning approaches require use of efficient training subset selection schemes whenever the available dataset is limited. If training and testing domains have significantly different measurement distributions, sample renormalization or other more elaborate techniques are required for adapting performance across domains.
2) Deep-learning approaches either require testing or training sets to have similar levels of noise, or to have a large amounts of data with different noise levels from multiple environments. Models created or tested using data from individual environments are unlikely to generalize well.
3) Signature-based approaches are highly effective for devices with clearly identifiable and unambiguous fingerprints, being able to work with very small training data size (even one sample for some devices in our experiments). However, they are not effective for devices that have non-decodable frequency content, such as microwave ovens [4]. Thus, signature-based approaches are likely to work most effectively for common types of transmitting devices, and other solutions are needed for broad frequency interference, and adversarial scenarios such as frequency jamming.
4) Regression-based methods can provide estimates of interference power levels, which can be used for estimating the impact on network throughput, as well as a proxy for distance when locating a source of interference.
5) Algorithms should be trained, from the start, for multi-label classification. Techniques based on spectrum unmixing can provide this as part of their solution, but a deep-learning network might need to optimize with a sigmoid loss function to discover concurrent transmitters.
6) Algorithm efficiency should be evaluated with respect to the number of devices detected. Even simple approaches might display polynomial time complexity as the number of devices increases.

VI. Discussion

Naturally our work has room for improvements, below we discuss some points.

Transfer learning Deep-learning approaches have shown a great capability for transfer learning. For instance, in image recognition it is common to train the initial – feature learning – layers of the network separately, after which they can be re-used in future training sessions [25]. This can greatly improve robustness, since the network is not randomly initialized every time new training occurs. A possible way to improve robustness of deep learning then is to use transfer learning to map measurements across training and testing domains, similarly to what has been explored in WiFi sensing domain [14]. Another benefit of transfer learning is that it can be used to reduce model training time, as shown in our previous work [4].

Unintentional radiators Our previous work [4] suggests that signal modeling approaches may have difficulties with devices that have ambiguous spectrum signatures, such as microwave ovens which emit across a broad spectrum range and which are not designed to transmit a decodable signal. In this particular case, however, the signal is known to correlate with AC frequency (60Hz), which could be exploited using autocorrelation [26].

Learning device classification in the wild. The signal model approach proposed in this worked showed great capability to learn from limited data. This would allow new interference sources to be added to the model by end users – e.g., potentially as part of a standard wireless survey. Due to improved robustness offered by our model, it is also likely that this learned signature is easier to apply to other environments.

Deep learning for complex cases Because a deep-learning technique such as a convolutional neural network can learn features over both the frequency and time domain, it has the capability of modelling devices whose output cannot strictly be described through their frequency output. Unexpected sources of interference like USB 3 hardware [27] are likely easier to learn with a technique that can take into account non-linearities in the time domain. From a security perspective, deep-learning could also be used to detect anomalies in packet traffic, or work in concert with existing approaches [28] by providing a physical layer perspective on potentially adversarial devices.

VII. Related work

Interference Detection The main idea in interference detection and classification approaches is to examine radio frequency spectrum sweeps and to classify signatures or fingerprints that can be identified from them. An early example of such systems is AirShark which detects and classifies interference sources using commodity WLAN hardware, a combination of generic and device-specific features and a decision tree for each device [1]. Importantly, for many classes of devices, a separate classifier and a set of features needs to be learned. WiSlow [29] also uses commodity hardware, but performs detection based on communication characteristics, such as packet analysis. Another work that uses device-specific features, such as the 60 Hz periodicity of microwave ovens, is presented in [26]. Though all these approaches more or less work on commodity hardware, they either require separate classifiers and features for each device, or have only been verified to work for known transmission protocols such as Bluetooth and ZigBee.

Deep Learning for Interference Detection The most recent interference techniques are based on deep learning. Our work considers a model based on our previous work [4]. Kulin et al. [3] examined different types of data representations for deep learning based interference detection, but found spectrum sweeps to be the most suitable. In terms of deep learning architecture, most approaches build on convolutional neural networks as it is often the network of choice in many approaches. For instance, a CNN has been used to detect specific radios transmitting WLAN signals from raw IQ samples (as opposed to FFT) in [30], or to detect transmitters in the 802.x family of protocols, i.e., Bluetooth and ZigBee, from spectrum (FFT) samples [2]. Common to these techniques is either the focus on simulated or lab-controlled signals, and/or technologies with known transmission protocols. Though works such as [31] have also been tested in noisy environments, both test and training measurements were collected from the same environment. Our evaluation provides the most comprehensive and realistic assessment thus far, using a systematic approach for varying training and test measurements, and considering a wide range of performance metrics.

Traditional Machine Learning cf. Deep Learning Differences between deep learning and traditional machine learning algorithms have been explored in several domains, mainly through different measures of accuracy (i.e., *correctness* in Sec. III) [32]–[34], and efficiency (in terms of the number of training samples) [35]. Mengmeng et al. [36] use three metrics, but compare robustness only among deep learning approaches. Closest to our work in terms of scope, Liu et al. [37] compare deep learning frameworks in terms of accuracy, efficiency and robustness. Their work focuses on the initial configuration of the learning frameworks, however, and does not provide a comparison between algorithms specifically. Our focus is narrower than most in terms of the absolute number of algorithms compared, but this specialization allows us to perform comparisons along more modes and with a wider examination of interpretability than previous works.

VIII. Summary and Conclusions

We contributed by performing a comprehensive performance evaluation of deep learning based interference detection using measurements from two highly diverse environments, and a systematic approach to noise injection that allowed us to evaluate how differences between training and testing data affect performance. Our results showed that deep learning is sensitive to noise, suffering from poor robustness, and sensitive to the amount of training data that is available. To overcome weaknesses in deep learning, we also presented a simpler yet efficient signal modeling based interference detection technique, which offers better robustness and interpretability than deep learning, with minimal degradation in detection performance. As such, both techniques have their uses, with signal modeling being better suited for enterprise environments where testing and training measurements can be difficult or next to impossible to collect, whereas deep learning is better suited for situations where training data can be easily collected or where the interference patterns are highly complex, such as in adversarial scenarios.

References

[1] S. Rayanchu, A. Patro, and S. Banerjee, "Airshark: Detecting non-wifi rf devices using commodity wifi hardware," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, 2011, pp. 137–154.

[2] N. Bitar, S. Muhammad, and H. H. Refai, "Wireless technology identification using deep convolutional neural networks," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct 2017, pp. 1–6.

[3] M. Kulin, T. Kazaz, I. Moerman, and E. De Poorter, "End-to-end learning from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications," *IEEE Access*, vol. 6, pp. 18 484–18 501, 2018.

[4] K. Longi, T. Pulkkinen, and A. Klami, "Semi-supervised convolutional neural networks for identifying wi-fi interference sources," in *Proceedings of the Ninth Asian Conference on Machine Learning*, vol. 77, 2017, pp. 391–406.

[5] https://www.cisco.com/c/en/us/solutions/enterprise-networks/cleanair-technology/index.html, [Online; accessed January 2020].

[6] https://www.netally.com/products/airmagnet-spectrum-xt/, [Online; accessed January 2020].

[7] https://www.netally.com/products/etherscopenxg, [Online; accessed January 2020].

[8] J. Van Waes, J. Vankeirsbilck, D. Pissoort, and J. Boydens, "Electromagnetic interference in the internet of things: An automotive insight," in *2017 XXVI International Scientific Conference Electronics (ET)*, Sep. 2017, pp. 1–4.

[9] N. Promsuk, A. Taparugssanagorn, and J. Vartiainen, "Interference suppression methods with adaptive threshold in internet of things (iot) systems," in *2017 9th International Conference on Information Technology and Electrical Engineering (ICITEE)*, Oct 2017, pp. 1–6.

[10] S. Paris, J. Elias, and A. Mehaoua, "Cross technology interference mitigation in body-to-body area networks," in *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, June 2013, pp. 1–9.

[11] T. Elshabrawy and J. Robert, "The impact of ism interference on lora ber performance," in *2018 IEEE Global Conference on Internet of Things (GCIoT)*, Dec 2018, pp. 1–5.

[12] K. Wiklundh and P. Stenumgaard, "Emc challenges for the era of massive internet of things," *IEEE Electromagnetic Compatibility Magazine*, vol. 8, no. 2, pp. 65–74, 2019.

[13] H. Zhang, B. Di, K. Bian, and L. Song, "Iot-u: Cellular internet-of-things networks over unlicensed spectrum," *IEEE Transactions on Wireless Communications*, vol. 18, no. 5, pp. 2477–2492, May 2019.

[14] J. Zhang, Z. Tang, M. Li, D. Fang, P. Nurmi, and Z. Wang, "Crosssense: towards cross-site and large-scale wifi sensing," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. ACM, 2018, pp. 305–320.

[15] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan, "Understanding and mitigating the impact of rf interference on 802.11 networks," in *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM 07. New York, NY, USA: Association for Computing Machinery, 2007, p. 385–396.

[16] R. Heydon and N. Hunn, "Bluetooth low energy," *CSR Presentation, Bluetooth SIG https://www. bluetooth. org/DocMan/handlers/DownloadDoc. ashx*, 2012.

[17] https://www.ekahau.com/products/sidekick/tech-specs/, [Online; accessed January 2020].

[18] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," arXiv, 2019.

[19] J. M. Zhang, E. T. Barr, B. Guedj, M. Harman, and J. Shawe-Taylor, "Perturbed model validation: A new framework to validate model relevance," arXiv, 2019.

[20] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," arXiv, 2018.

[21] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, and A. Criminisi, "Measuring neural net robustness with constraints," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 2613–2621.

[22] R. G. Gallager, *Principles of Digital Communication*. Cambridge University Press, 2008.

[23] H. Spieker and A. Gotlieb, "Towards testing of deep learning systems with training set reduction," arXiv, 2019.

[24] T. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall PTR, 2001.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[26] Z. Weng, P. Orlik, and K. J. Kim, "Classification of wireless interference on 2.4ghz spectrum," in *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, April 2014, pp. 786–791.

[27] https://www.intel.com/content/www/us/en/products/docs/io/universal-serial-bus/usb3-frequency-interference-paper.html, [Online; accessed January 2020].

[28] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 2177–2184.

[29] K.-H. Kim, H. Nam, and H. Schulzrinne, "Wislow: A wi-fi network performance troubleshooting tool for end users," 04 2014, pp. 862–870.

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

[31] 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, Feb 2018.

[32] S. Karimi, X. Dai, H. Hassanzadeh, and A. Nguyen, "Automatic diagnosis coding of radiology reports: A comparison of deep learning and conventional classification methods," in *BioNLP 2017*. Vancouver, Canada,: Association for Computational Linguistics, Aug. 2017, pp. 328–332.

[33] N. G. Paterakis, E. Mocanu, M. Gibescu, B. Stappers, and W. van Alst, "Deep learning versus traditional machine learning methods for aggregated energy demand prediction," in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Sep. 2017, pp. 1–6.

[34] J. Lago, F. D. Ridder, and B. D. Schutter, "Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms," *Applied Energy*, vol. 221, pp. 386 – 405, 2018.

[35] B. Dong and X. Wang, "Comparison deep learning method to traditional methods using for network intrusion detection," in *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*, June 2016, pp. 581–585.

[36] M. Cai, M. Pipattanasomporn, and S. Rahman, "Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques," *Applied Energy*, vol. 236, pp. 1078 – 1088, 2019.

[37] L. Liu, Y. Wu, W. Wei, W. Cao, S. Sahin, and Q. Zhang, "Benchmarking deep learning frameworks: Design considerations, metrics and beyond," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, July 2018, pp. 1258–1269.