Interference Classification in a Factory Environment Based on Semi-supervised Deep Learning

Su Yi, Hao Wang, Wenqian Xue, and Lefei Wang
Fujitsu Research and Development Center
Beijing, China
Email: {yisu, wangh, xuewenqian, wanglefei}@cn.fujitsu.com

***Abstract*—The steadily growing use of license-free frequency bands requires reliable coexistence management and therefore proper wireless interference identification. This paper provides a realtime interference source classification method based on semi-supervised deep learning. It uses Received Signal Strength Indicator (RSSI) samples collected by an 802.15.4-based wireless sensor for formulating training data as well as online test data in a factory environment. To address the issue of laborious process on labeling the training data, a Fast Fourier Transform (FFT)-based algorithm is used to help labeling the sample data. We have trained a deep neural network with two hidden convolutional layers using raw RSSI samples as inputs. The whole realtime management system with the classifier is implemented on IEEE 802.15.4 System on Chip (SoC) and Linux-based system.**

Keywords–Interference classification; Semi-supervised deep learning; RSSI sampling.

I. INTRODUCTION

The Internet of Things (IoT) is one of the most important, exciting, and transformational technology developments today. IoT is global in impact, multi-disciplinary in nature, and spans virtually all industry segments. Recent trends to introduce IoT devices, such as sensors and cameras into factories have been accelerated by a strong demand for improving productivity, reducing labors and cost. For this reason, the digitalization of the factories, as well as the connection of information on production process and supply chain management within a factory and across factories are becoming important.

There are several system applications, e.g., preventive maintenance, management of materials and products, monitoring of movements and machine monitors, which are integrated in the network. More efforts will be required for wireless communication because of its limited and shared radio resources and the sensitive nature of the environment in which it will operate in. One of the most common and frequent wireless issues is related with interference, which is common to all wireless short range networks operating in unlicensed frequency band, such as IEEE 802.11, 802.15.4, 802.15.1, etc. Interference issues are critical since multiple systems may interfere with each other, and the number of nodes in the unlicensed spectrum is increasing rapidly.

When deploying a sensor network in a factory, it has been found that there exists different levels of interference, during different times, and at different positions. This paper aims to identify different interference sources in a factory environment. By diagnosing different interference sources, either at the network provisioning phase or during operation, IoT service providers can do better network planning, take countermeasures to solve the problem or avoid potential problems. Therefore, realtime state monitoring and automated trouble detection are required for efficient operation and management services. For instance, with the help of the resulting knowledge, the system can adapt its communication by choosing a better channel or other mitigation strategies.

The key idea is to use a low power, narrow band IEEE 802.15.4 wireless module to sample radio frequency (RF) energy and identify the interference source by learned interference patterns. The IEEE 802.15.4 standard defines that an Energy Detection (ED) value must be measured for the Clear Channel Assessment (CCA) and channel selection [1]. It is an estimate of the received signal power within the bandwidth of an IEEE 802.15.4 channel. No attempt is made to identify or decode signals on the channel. The ED time shall be equal to 8 symbol periods. This ED value is also widely known as the Received Signal Strength Indicator (RSSI). By setting the sampler on different ZigBee channels, it can detect Wi-Fi, ZigBee, Bluetooth, Bluetooth Low Energy (BLE), microwave oven, and other magnetrons, which use the spectrum overlapped with the sampler's channel.

Many efforts have been made to classify interference in wireless networks, such as in Wi-Fi systems or sensor networks. Airshark [2] leverages powerful Wi-Fi hardware to get the spectrum information to detect and classify non-Wi-Fi interference. In [3]–[5], the authors propose methods to classify interference by the observation that different interferences will result in different corruption patterns on received packets. The authors in [6]–[9] study and extract features purely from the time-domain RSSI sequence, and design a classification approach to identify the existence of different sources of interference. In recent years, deep learning has been introduced to analyze the spectral data for signal identification. In [10]–[12], spectral samples over frequency and time span are collected for training using convolutional neural networks.

Most previous works face the problem of labeling the interference source effectively. They often use human experience to label the training data or test data. Labeled data is always very hard to get since human annotation is boring and time-consuming. In our previous work [13], we study the interference effect on the link condition and the network performance and design a machine learning method to classify the wireless channel errors into different categories. We also conduct extensive experiments in office to study the RSSI patterns and use deep learning to identify different patterns for major wireless scenarios [14]. In this paper, we develop a semi-

 ISBN: 978-1-61208-727-6

supervised deep learning mechanism to identify interference types with focus on a factory scenario. Compared with our previous work, one novel point is an auto-labeling algorithm which can be done using training data collected from natural environment instead of a controlled environment.

The rest of the paper is structured as follows. Section II gives an overview of deployment of the interference identification. In Section III, we describe the semi-supervised deep learning with an auto-labeling method. Section IV provides the test results. Finally, we conclude the work in Section V.

II. Deployment Scenario

The interference identification framework we propose in this work can be deployed in industrial IoT where some example IoT applications include miniaturized sensors integrated into critical equipment that monitor performance parameters to proactively diagnose maintenance issues, enable trend analysis of equipment performance, and optimize overall system operations.

Inspired by recent advances and the remarkable success of deep learning in a broad range of problems such as image or speech recognition and machine translation, we use similar approaches in interference classification with a high rate RSSI sampler to get RSSI traces as input. As a measurement of the RF power level at the input of the transceiver, RSSI is an important parameter to reflect a wireless channel condition. When there is interference, the RF energy increases so it can be used to detect the occurrence of interference.

In our implementation, a 95μs sampling interval (10.5kHz sampling rate) is achievable. We use a TI CC2530 802.15.4 module and program it to read the built-in RSSI register continuously to get the RSSI sampling data. The RSSI sampler captures the energy in the channel due to the interferers' emissions. It continuously reads the RSSI register of the sensor nodes' radio chip.

The interference analysis is done in a very short period to reflect the fast-changing channel condition. We use the training data collected from different channels to train a unified data model, then this data model becomes channel-independent. That is to say, when we run the online classification algorithm, we use the same model for any sampling channel. The end user, such as the network administrator or a network management application, can generate a detailed report for every diagnosis window or a report with statistical results over a longer period.

Figure 1 illustrates our deployment scenario where these samplers are placed in the intended locations in a factory environment. The RSSI sampler is connected to a small single-board computer Raspberry Pi (RPi) 3 with an USB to serial interface. The sampling results are easily accessible by the interference analysis engine on the RPi. One or more samplers can be placed in a certain area to get the channel information of that area. The Network Management System (NMS), usually implemented with a graphical user interface (GUI), or a Web service, can remotely access the analysis engine on the RPi through Internet. This management system can configure the setting of the interference diagnosis and read the results of the interference source classification.

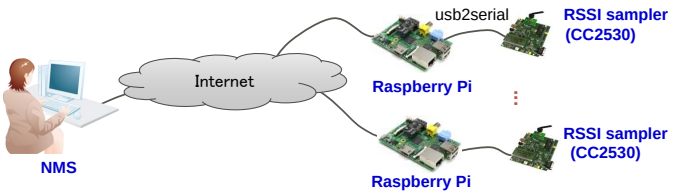


Figure 1. Deployment scenario of the interference classification system.

Another implementation choice is to implement RSSI sampling function on a sensor node. In this case, there is no dedicated node for sampling. The sampling function may be activated during provisioning phase, analyzing phase, or as all-time monitoring.

In the investigated factory, we find out that the interference sources mainly come from electromagnetic radiation caused by the operating machines, and interference caused by the sparse wireless communication signals from the sensor network deployment, such as beacons and sensor data. It has been shown in many studies that industrial and factory equipment produces electromagnetic interference (EMI) that causes a great deal of damage on wireless performance [15].

Figure 2 shows four typical RSSI traces for different interference scenarios in our factory, as well as a *Normal* scenario (Figure 2a) meaning that channel is relatively idle and clear of interference. Sometimes there will be a mix of different patterns of these four.

The RSSI pattern in Figure 2b is a waveform with a cycle of 20ms, which means the energy pattern is periodic with a frequency cycle around 50Hz. Figure 2c has a similar pattern, only that the period is about 10ms, leading to a frequency cycle of 100Hz.

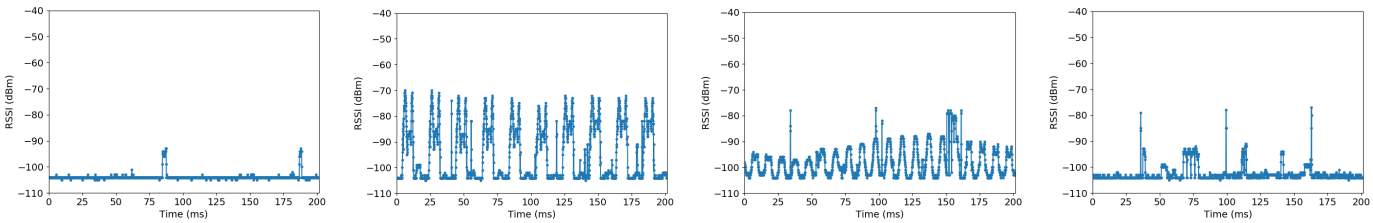
Figure 2d is the RSSI pattern for pulse-shaped wireless communication signals. In factories where there is very sparse traffic, the pulse-like signal pulses are mostly from beacons (Wi-Fi, Bluetooth, BLE, etc.), or sensor data transmitting measurement results. These energy peaks also exhibit a periodic pattern (with a 60+ms period in the figure). We define this pattern as the result of wireless communication signals.

III. Interference Classification with Deep Learning

Traditional machine learning has been used in error diagnosis or interference source identification in the wireless systems in literatures while the results are not so promising. Deep learning may be utilized to automatically extract more low-level and high-level features and has been used in complex applications [16]. It normally requires large amount of training data, which can be achievable by the high rate RSSI sampling.

The RSSI samples in a detection window (20ms) form an N-element input vector. Since a sampling rate 95μs/sample is used, $N = 20\text{ms}/95\mu\text{s} \approx 210$. This N-element input vector is passed to a deep learning classification model to generate an $M \times 1$ output vector. M represents the number of classes defined. In this paper, $M = 4$ representing *Normal* and 3 types of interference sources.

 ISBN: 978-1-61208-727-6



(a) Normal (b) Magnetron with 20ms cycle (c) Magnetron with 10ms cycle (d) Wireless communication signal

Figure 2. Typical RSSI traces of difference cases.

A. Auto-labeling

Training data is collected on a 24-hour basis in a factory. Several days of data are used to formulate the training data set. Before training, we need to prepare all the input vector data composed of RSSI samples together with their labels. This is specifically challenging since it is unfeasible to find out what is actually happening over the air on a 24-hour basis. This paper uses Fast Fourier Transform (FFT) to automatically label the training data to avoid human annotation.

Due to the fact that all interference patterns in our factory are found to be periodic, the proposed automatic labeling utilizes FFT to find the spectrum characteristics of the RSSI samples. Magnetron patterns and wireless communication signal patterns are quite different in that wireless communication signals cover a much shorter time period and are very easy to be overwhelmed by magnetrons or other noises. It happens that magnetrons mostly happen in daytime, since they come from machine noises. We collect sample logs for several days and nights and separate them into daytime training data and night training data. Daytime training data are used to find magnetron patterns and night training data are for wireless communication signals.

Firstly, we use FFT to find the ideal patterns for magnetrons and for normal state. We use daytime training data and label selected data automatically by FFT method. In detail, the sample log is divided into sample sequences of length N_F corresponding to a period of time T_F. T_F can be considered as a hyper-window size (multiple of detection window size) used for FFT algorithm. For each hyper-window, N_F-point FFT is applied to find the fundamental frequency. N_F is large enough to guarantee the frequency resolution but not too large to have a mixed RSSI pattern in a hyper-window. In our experiments, $N_F = 5250$, so $T_F = N_F \times 95\mu\text{s} \approx 500\text{ms}$. Sampling frequency $f_s = 1/95\mu\text{s} = 10.526\text{kHz}$. The frequency resolution $d_f = f_s/N_F = 2\text{Hz}$.

We group these hyper-windows by their fundamental frequencies, and count the number of hyper-windows for each group over these sample logs. The number of hyper-windows for each fundamental frequency is given in Table I. Hyper-windows with DC (direct current) component (0Hz) as fundamental frequency indicate samples in these hyper-windows represent a normal state, so these hyper-windows belong to *Normal*. It is useful to choose fundamental frequencies (except DC component) with obviously higher number of hyper-windows than others. It implies that the interfering magnetron waveform has a cycle with the corresponding fundamental frequency. As a result, two fundamental frequencies (50Hz and 100Hz) with the most numbers of hyper-windows are selected, which exactly matches our observation on Figure 2b and 2c. Note here 50Hz or 100Hz frequency means that the interference is a waveform with 50Hz or 100Hz cycle (20ms or 10ms period). The carrier frequency of the interference source is the same as that of the sampler. We label hyper-windows with 50Hz fundamental frequency as *Interference #1* and hyper-windows with 100Hz fundamental frequency as *Interference #2*. Other hyper-windows are unlabeled for now.

TABLE I. NUMBER OF HYPER-WINDOWS WITH LENGTH T_F GROUPED BY FUNDAMENTAL FREQUENCY

Fund. freq. (Hz)	0	4	6	8	10	12	16	20
# hyper-windows	**11879**	63	9	3	3	7	2	2
Fund. freq. (Hz)	22	24	26	30	34	36	38	40
# hyper-windows	4	18	4	1	1	1	2	2
Fund. freq. (Hz)	42	44	46	48	50	54	80	88
# hyper-windows	3	17	15	38	**236**	1	1	1
Fund. freq. (Hz)	98	100	136	142	150	188	200	250
# hyper-windows	8	**87**	1	1	40	3	18	2

We break up each labeled hyper-window into multiple (500ms/20ms = 25) detection windows and label each 20ms-detection window using the same label of hyper-window it belongs to. Now we have three types of labeled training data: *Normal*, *Interference #1* and *Interference #2*.

Secondly, we use FFT to find the ideal patterns for wireless communication signals using night training data. Unlike the magnetron, the pulse-like wireless communication signal only covers very short time. Due to the background noise, the energy of the signal pulse cannot be identified by directly using FFT. To identify pulse-like wireless signal, a filter (Figure 3) is used to filter out noises before using FFT.

In our experiments, RSSI_THRESHOLD = -80dBm, NOISE_FLOOR = -108dBm.

The new filtered RSSI trace is used to do FFT and automatic labeling. For each hyper-window with length T'_F, FFT is used to find the fundamental frequency. $N'_F = 52500$, $T'_F = N'_F \times 95\mu\text{s} \approx 5\text{s}$ and thus $d'_f = 0.2\text{Hz}$. T'_F is larger than T_F since wireless communication signals normally have a period longer than that of the magnetrons. The fundamental frequencies are rounded down to the nearest integer and the count of hyper-windows for each rounded-down fundamental frequency over these sample logs is listed in Table II. Similarly, one or more fundamental frequencies which have obviously highest number of hyper-windows are selected. Hyper-window

 ISBN: 978-1-61208-727-6

Input: A sequence of RSSI values: $\{rssi_0, rssi_1, ..., rssi_{n-1}\}$
Output: New sequence of RSSI after filtering: $\{rssi\_filt_0, rssi\_filt_1, ..., rssi\_filt_{n-1}\}$
1: **for** $(i = 0; i < n; i++)$ **do**
2: **if** $rssi_i <$ RSSI_THRESHOLD **then**
3: $rssi\_filt_i =$ NOISE_FLOOR
4: **else**
5: $rssi\_filt_i = rssi_i$
6: **end if**
7: **end for**

Figure 3. Pseudo codes for RSSI filtering

Input: Labeled data (X_l, Y_l), unlabeled data X_u.
Output: CNN model $f : \mathbf{X} \rightarrow \mathbf{Y}$
1: $f \leftarrow$ train using (X_l, Y_l)
2: **for** $x \in X_u$ **do**
3: Predict using $y = f(x)$
4: **if** $y! = Normal$ **then**
5: $(X_l, Y_l) \leftarrow (X_l + x, Y_l + y)$
6: $f \leftarrow$ train using (X_l, Y_l)
7: **end if**
8: **end for**

Figure 4. Semi-supervised learning algorithm

with selected fundamental frequency means that the samples in this hyper-window indicate the existence of a periodic, pulse-like wireless signal. Frequency 15Hz (actually ranging from 15Hz to 16Hz) has the maximum number of hyper-windows, which corresponds to the ˜64ms period signal shown in Figure 2d. These hyper-windows are labeled as *Interference #3*. Then all the RSSI samples are replaced by their original pre-filter values.

TABLE II. NUMBER OF HYPER-WINDOWS WITH LENGTH T'_F GROUPED BY FUNDAMENTAL FREQUENCY

Fund. freq. (Hz)	0	5	6	7	8	9	10	11
# hyper-windows	2548	874	139	38	534	55	59	34
Fund. freq. (Hz)	12	13	14	15	16	19	20	...
# hyper-windows	18	15	7	**2582**	46	9	13	...

We split the labeled hyper-window with length T'_F into multiple prediction windows. Only the prediction windows which have samples with value greater than RSSI_THRESHOLD are chosen and labeled as *Interference #3*. We put these new labeled data into training data. Other data are left unlabeled and will be used in later semi-supervised learning.

Finally, this auto-labeling process results in the initial training data, each in a length of a detection window, with four different labels. It's necessary to balance the amount of training data for each class before training.

One may ask why not apply the same labeling technique to the test samples without learning process. The main reason for using deep learning is that FFT can only find the typical patterns of these interference source. There are large number of unlabeled data due to FFT resolution errors, non-ideal RSSI patterns, or mixed scenarios in a hyper-window. These data cannot be solved by using FFT alone. Besides, most labeled data are actually *Normal*. Semi-supervised learning is used to utilize larger amounts of unsupervised labels, specifically non-normal labels to improve the accuracy.

B. Semi-supervised Learning

When auto-labeling is done, there are data left unlabeled. To make use of unlabeled data for training as well, each unlabeled hyper-window is partitioned into multiple windows in 20ms. We use a conventional semi-supervised learning algorithm – self-training – to learn the deep model [17]. The basic procedure is shown in Figure 4. We first train a deep model using labeled data, then predict on unlabeled data and get a classification output. After each prediction, we add the wanted input-output pair to the labeled data to form a new set of training data and then train again until all unlabeled data are checked.

Slightly different from Figure 4, in real implementation, unlabeled data are put into iteration in batches for higher efficiency. A final deep model can be achieved in the end. By using semi-supervised learning, the training data amount has been increased from 70,000+ to 100,000+.

C. Convolutional Neural Network

Convolutional Neural Network (CNN) is used in the semi-supervised deep learning since it is applicable to array data where nearby values are correlated, and it greatly reduces number of parameters for deep networks. CNN performs feature learning via non-linear transformations implemented as a series of nested layers. The raw RSSI samples organized into data vectors are pipelined as input for classification. Traditionally CNN is used mostly for 2-dimensional inputs, such as in image recognition. In our case we just use one dimension – a vector – as input. This can be considered as a special case and the other properties for the network remain the same.

The goal of deep learning or more generally, machine learning is to find a mathematical function f, that defines the relation between a set of inputs $\mathbf{X}$, and a set of outputs $\mathbf{Y}$, i.e.

$$f : \mathbf{X} \rightarrow \mathbf{Y} \quad (1)$$

The inputs, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_K]^T$, present a number of distinct data points, samples or observations, where K is the sample size, while $\mathbf{x}_i$ is a vector of N measurements of features for the ith observation called a feature vector. $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{iN}]^T, i = 1, ..., K$. The outputs, $\mathbf{y}$, are all the outcomes, labels, or target values corresponding to the K inputs $\mathbf{x}_i$, denoted by $\mathbf{y} = [y_1, y_2, ..., y_K]^T$. Then the observed data consists of K input-output pairs, called the training data or training set $\mathbf{S}$.

We use three main types of layers to build CNN architectures: convolutional layer, pooling layer, and fully-connected (FC) layer (exactly as seen in regular neural networks). After investigation on relevant literatures and numerous experiments, we have settled with LeNet model, a CNN with two convolutional layers [18]. The configurations for the network architecture are given in Figure 5, with a 210-element vector as input and a 4-class output. The last FC layer $[4 \times 1]$

 ISBN: 978-1-61208-727-6

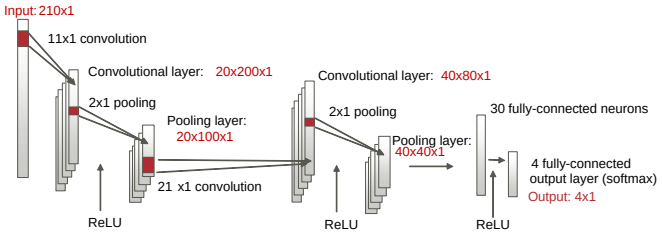


Figure 5. CNN architecture.

will compute the class scores where each of the 4 numbers corresponds to a class score, among the 4 types of classes. The very last layer is a Softmax classifier, which computes the *posterior* probability of each class label over 4 classes as

$$\hat{y_i} = \frac{e_i^z}{\sum_{j=1}^{4} e_j^z}, i = 1, ..., 4 \quad (2)$$

That is, the scores z_i computed at the output layer are translated into probabilities.

A cost function, C, is calculated on the last fully-connected layer that measures the difference between the estimated probability vector, $\hat{\mathbf{y}}_i$, and the index encoding of the true class label, y_i. The CNN parameters, Θ, are obtained by minimizing the cost function on the training set $\{\mathbf{x}_i, y_i\}_{i \in \mathbf{S}}$ of size K,

$$\min_{\Theta} \sum_{i \in \mathbf{S}} C(\hat{\mathbf{y}}_i, y_i) \quad (3)$$

where $C(\hat{\mathbf{y}}_i, y_i) \equiv -\ln(\hat{\mathbf{y}}_i[y_i])$ is the negative log-likelihood cost function. Note that $\hat{\mathbf{y}}_i[y_i]$ means the y_i-th element of the vector $\hat{\mathbf{y}}_i$.

All the RSSI samples are normalized to range [-1, 1] as CNN inputs. In the training process, stochastic gradient descent (SGD) is used with backpropagation with a mini-batch size of 500 and a learning rate 0.1, as well as L2 regularization to avoid overfitting.

One advantage of this framework is that our data inputs for learning and testing do not need feature extraction as most prior arts do. Feature extraction may have a chance to lose some hidden features in the data. We use raw RSSI samples as input so that all information is conveyed to deep learning.

IV. Results

We have implemented the off-line training on a Linux based server, and an online realtime detection system with a RPi, a ZigBee sampler, and a GUI on a PC.

With regard to the implementation on the off-line training, Python 2.7 in combination with computation library Theano 0.9 is utilized. The CNN is trained and validated on a high computation platform with 24-core CPU Intel(R) Xeon(R) E5-2620 v2 @ 2.10GHz, with 128GB RAM and the Cuda enabled GPU Nvidia Tesla K80.

When the model is trained, the calculation for classifying a test instance into one of the classes is very fast since each test instance needs to be compared against the pre-computed model. The computation time for a single test data is around 8ms on RPi. If test data are input to the classifier in batches the average computation speed will be even faster. Therefore, the RSSI samples can be fed into the trained CNN model to get the diagnosis result in realtime with a RPi.

To evaluate proposed interference source identification scheme, extensive experiments are conducted in a factory for several days. Test data and training data are collected at different time to reduce the dependency of the data. The RPi is used as a sample data collector, as well as a predictor during online test phase. This system basically only consists of 'listening' radio devices, which do not interfere with the current wireless communication system.

Due to security and other reasons, any change of the operating machines or change of the wireless system in the field is not allowed. This leads to the difficulty to obtain the ground truth of the environment. Nevertheless we have randomly picked some RSSI traces from different times and checked with the prediction result using human's knowledge, and the detailed prediction results (%) are compared with observed results in Table III. As introduced in Section III-A, three types of interference sources are: *Interference #1* – magnetron with 50Hz frequency cycle, *Interference #2* – magnetron with 100Hz frequency cycle, *Interference #3* – wireless communication signal. Note that some RSSI patterns are unidentifiable by humans, so there is an additional observed class named *Unknown*.

TABLE III. The confusion matrix of identified classes

		Predicted class (%)			
		Normal	*Intf. #1*	*Intf. #2*	*Intf. #3*
Observed class	*Normal*	**100**	0	0	0
	Intf. #1	0	**100**	0	0
	Intf. #2	1.3	16.4	**82.3**	0
	Intf. #3	15.8	10.5	0	**73.7**
	Unknown	25.5	16.4	58.2	0

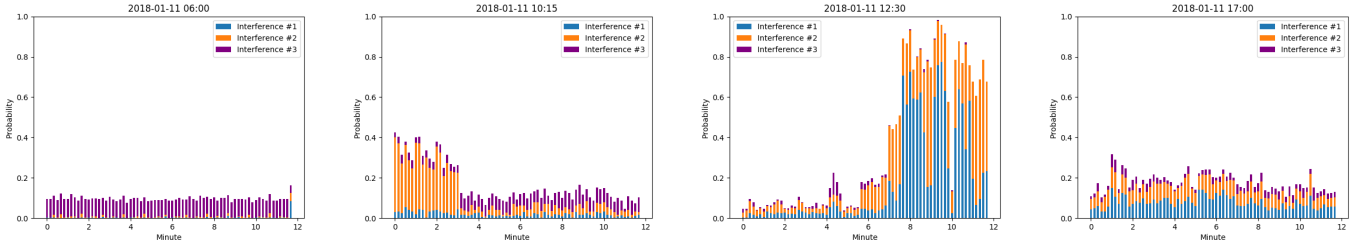
The predictions for selected time periods are plotted in Figure 6 with each plot having a duration of about 12 minutes. The ratio of outputs for each of the 4 classes is calculated every 10 seconds. Only ratios of non-*Normal* results are plotted in colored bars. Predictions for different time periods from two days are compared. In early morning (6:00∼6:12), the predictions of all interference types are low for both days. During 10:00∼10:27, probability of *Interference #2* increases for some time on both days. Around noon (12:30∼12:42), predictions for *Interference #1* become dominant and show a very high probability for both days. In the afternoon (17:00∼17:12), the predictions for interference decrease while the results for Day1 and Day2 are slightly different.

Generally, the magnetron cycle is a combination effect of the machine model, AC (alternating current) power cycle, transformer type, switching type, inverter type, etc. This prediction can help manage, locate, or mitigate interference caused by magnetron leakage or wireless systems in deployment scenarios.

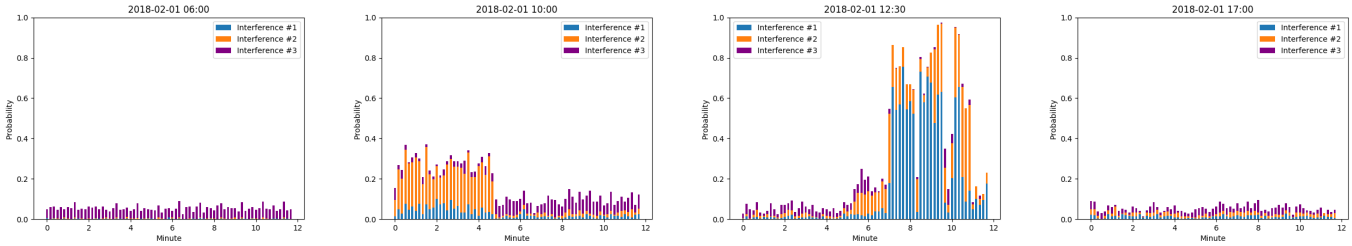
V. Conclusion

In this paper, we collect extensive sample data from a factory to study the RSSI trace patterns which are sampled by IEEE 802.15.4 nodes for different interference sources. A semi-supervised deep learning utilizing RSSI samples is

 ISBN: 978-1-61208-727-6



(a) Day1 06:00∼06:12 (b) Day1 10:15∼10:27 (c) Day1 12:30∼12:42 (d) Day1 17:00∼17:12



(e) Day2 06:00∼06:12 (f) Day2 10:00∼10:12 (g) Day2 12:30∼12:42 (h) Day2 17:00∼17:12

Figure 6. Selected predictions on factory data.

proposed to infer the interference type. Automatic labeling of training data is achieved taking advantage of the periodicity of the interference patterns. Sample collection and the classification algorithm are implemented on RPi 3 to monitor the wireless channel condition in a realtime fashion. Compared with the spectral data from a spectral analyzer, RSSI samples on the working channel are much easier to obtain.

The training procedure has to be performed when used in different environments and must be validated in the field to become a viable option as a classifier. This interference source identification scheme opens up numerous other possibilities. There can be a dedicated device or it can be embedded in a sensor to do external interference avoidance mechanisms based on the input from the sampling. Finally, the classification result of a channel is not only informative, but can be used to adapt the transmit parameters. Thus, an interference-aware communication protocol that adapts its parameters to the class of interference is a potential application for this algorithm. Further validation of these prediction results in a factory deployment and how to utilize these results remain for our future research.

ACKNOWLEDGMENT

The authors would like to thank Yuki Nishiguchi, Ai Yano, Takeshi Ohtani, and Ryuichi Matsukura from Fujitsu Laboratories Ltd., Kawasaki, Japan for providing experimental data in factory and all the fruitful discussions on the topic.

REFERENCES

[1] IEEE Std 802.15.4-2015, “Part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs),” 2015.

[2] 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, Berlin, Germany, November 2011, pp. 137–154.

[3] F. Hermans, O. Rensfelt, L.-Å. Larzon, and P. Gunningberg, “A lightweight approach to online detection and classification of interference in 802.15.4-based sensor networks,” ACM SIGBED Review – Special Issue on the 3rd International Workshop on Networks of Cooperating Objects (CONET), vol. 9, no. 3, July 2012, pp. 11–20.

[4] F. Hermans, O. Rensfelt, T. Voigt, E. Ngai, L.-Å. Nordén, and P. Gunningberg, “SoNIC: Classifying interference in 802.15.4 sensor networks,” in Proceedings of ACM/IEEE IPSN, Philadelphia, PA, USA, April 2013, pp. 55–66.

[5] K. Wu, H. Tan, H.-L. Ngan, Y. Liu, and L. M. Ni, “Chip error pattern analysis in ieee 802.15.4,” IEEE Transactions on Mobile Computing, vol. 11, no. 4, April 2012, pp. 543–552.

[6] X. Zheng, Z. Cao, and J. Wang, “ZiSense: Towards interference resilient duty cycling in wireless sensor networks,” in Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys), Memphis, TN, USA, November 2014, pp. 119–133.

[7] S. Zacharias, T. Newe, S. O’Keeffe, and E. Lewis, “Identifying sources of interference in rssi traces of a single ieee 802.15.4 channel,” in Proceedings of the Eighth International Conference on Wireless and Mobile Communications (ICWMC), Venice, Italy, June 2012, pp. 408–414.

[8] ——, “A lightweight classification algorithm for external sources of interference in ieee 802.15.4-based wireless sensor networks operating at the 2.4 GHz,” International Journal of Distributed Sensor Networks, vol. 10, no. 9, August 2014, pp. 265–286.

[9] V. Iyer, F. Hermans, and T. Voigt, “Detecting and avoiding multiple sources of interference in the 2.4 GHz spectrum,” EWSN 2015, LNCS, vol. 8965, 2015, pp. 35–51.

[10] M. Schmidt, D. Block, and U. Meier, “Wireless interference identification with convolutional neural networks,” CoRR, 2017. [Online]. Available: http://arxiv.org/abs/1703.00737

[11] 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, ser. PMLR 77, Seoul, Korea, November 2017, pp. 391–406.

[12] M. Kulin, T. Kazaz, I. Moerman, and E. de Poorter, “End-to-end learning from spectrum data: A deep learning approach for wireless

 ISBN: 978-1-61208-727-6

signal identification in spectrum monitoring applications," IEEE Access, March 2018, pp. 18 484–18 501.

[13] S. Yi et al., "Machine learning based channel error diagnostics in wireless sensor networks," in Proceedings of IEEE VTC Spring, Sydney, Australia, June 2017, pp. 1–5.

[14] S. Yi et al., "Interference source identification for ieee 802.15.4 wireless sensor networks using deep learning," in Proceedings of IEEE PIMRC, Bologna, Italy, September 2018, pp. 1–7.

[15] J. Chilo, C. Karlsson, P. Ängskog, and P. Stenumgaard, "EMI disruptive effect on wireless industrial communication systems in a paper plant," in Proceedings of IEEE International Symposium on Electromagnetic Compatibility (EMC), Austin, TX, USA, August 2009, pp. 221–224.

[16] M. A. Nielsen, Neural Networks and Deep Learning. Determination Press, 2015.

[17] X. Zhu, "Semi-supervised learning tutorial," in Tutorial of International Conference on Machine Learning (ICML), Corvallis, OR, USA, June 2007.

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, November 1998, pp. 2278–2324.

 ISBN: 978-1-61208-727-6