Error-based Interference Detection in WiFi Networks

Nicola Inzerillo[1], Daniele Croce[1,2], Domenico Garlisi[1,2], Fabrizio Giuliano[1,2], Ilenia Tinnirello[1]
[1]DEIM, Università di Palermo, viale delle scienze ed. 9 - 90128 Palermo, Italy
[2]CNIT Consortium, Viale G.P. Usberti, 181/A - 43124 Parma, Italy
name.surname@unipa.it

***Abstract*—In this paper we show that inter-technology interference can be recognized by commodity WiFi devices by monitoring the statistics of receiver errors. Indeed, while for WiFi standard frames the error probability varies during the frame reception in different frame fields (PHY, MAC headers, payloads) protected with heterogeneous coding, errors may appear randomly at any point during the time the demodulator is trying to receive an exogenous interfering signal. We thus detect and identify cross-technology interference on off-the-shelf WiFi cards by monitoring the sequence of receiver errors (bad PLCP, bad PCS, invalid headers, etc.) and develop an Artificial Neural Network (ANN) to recognize the source of interference. The result is quite impressive, reaching an average accuracy of almost 99% in recognizing ZigBee, Microwave and LTE (in unlicensed spectrum) interference.**

***Index Terms*—Wireless LAN, Interference, Artificial Neural Networks.**

I. Introduction

Nowadays, we are witnessing an impressive success of IEEE 802.11 technology, better known as WiFi, for supporting the growing demand of wireless broadband connectivity. Public WiFi networks are deployed worldwide, with more than 50% of the total mobile traffic carried by WiFi. The availability of WiFi networks is often considered as a commodity service driving immense economic value, and the unlicensed spectrum is becoming one of society's most valuable resources. Although WiFi is a dominant communication technology in this spectrum, many other low range technologies coexist in unlicensed ISM bands for supporting several vertical applications, such as building automation, smart metering systems, health care monitoring, surveillance systems, game remote controllers and so on. Moreover, cellular technologies are trying to extend their operation to ISM bands for increasing their capacity. Two different solutions have been envisioned by 3GPP in ISM bands, referred to as Licensed Assisted Access (LAA) [1] and LTE-Unlicensed (LTE-U) [2], which work respectively, with and without the listen-before-talk mechanism.

Although in WiFi carrier sense and adaptive modulation mechanisms have been included, it has been shown that serious performance impairments can arise in presence of exogenous interfering signals due to different technologies. For example, in [3] it is shown that the capacity of a good WiFi link can be reduced to zero in presence of analog phones, video cameras, or sensors based on IEEE 802.15.4 technology [4], [5], while other devices such as a Xbox controller and a microwave oven can half the throughput. The effect of sensors' interference on the WiFi link is impressive if we consider that the 802.11 and 802.15.4 technologies (hereafter referred as ZigBee and WiFi) are pretty heterogeneous in terms of bandwidth (2 versus 20 MHz) and transmission power (e.g. 0 dBm for ZigBee and 20 dBm for WiFi). The possible reasons are that some WiFi implementations are unable to detect non-WiFi signals or introduce latencies [6] or because of the different timings to perform CSMA/CA [7], [8]. About the interference with cellular technologies, several research studies are trying to characterize the impact on LTE transmissions on WiFi performance. Preliminary empirical and simulation results [9] show that WiFi performance can be critically affected even when LTE links operate at the minimum bandwidth of 1.4 MHz. In [10] it is demonstrated that even when utilizing the listen-before-talk principle, LAA-LTE heavily impacts WiFi performance, and that WiFi with MIMO performs worse than WiFi without MIMO when LTE interference is strong. Additionally, increasing distance between LTE and WiFi links does not necessarily decrease the impact of interference in indoor environments.

In this scenario, it is important to identify the presence of such coexisting technologies to allow possible countermeasures (e.g. finding a different channel) or activate some coordination mechanisms (e.g. setup a cross-technology TDMA scheme [11]). For this classification, the typical approach in the literature is to analyze the RSSI samples in the frequency and time domain [3], [12], or to perform a cyclostationary signal analysis and blind signal detection [13] and other spectrum sensing techniques [14]. Although these approaches are very effective, they usually require to monitor the interfering signals for some seconds or use specialized hardware. Instead, in this work we propose to simply monitor the reception errors of commodity WiFi cards, and then apply an artificial neural network in order to identify and characterize cross-technology interference. Following the initial approach of [15], we analyze the *error domain*, i.e. the error burst caused by the interfering signal. However, in this paper we exploit a more powerful classification tool, the Artificial Neural Networks (ANNs), and we extend the analysis to the emerging *LTE in unlicensed spectrum*. Experimental results show that our solution provides excellent results, with an average of almost 99% accuracy.

This work has been partially supported by EU funded H2020 research projects symbIoTe, Grant No. 688156, and WiSHFUL, Grant No. 645274.

Although in this work we focus on three interference sources, namely ZigBee, LTE and microwave ovens, our solution does not depend on the type of technology, but only requires a training phase based on the events generated in presence of a controlled source of interference. We then employ an ANN to identify the interference technology from the occurrence of the generated error events. The idea is that the proposed approach could be easily extended to any other type of interference.

After a brief literature review of background information (section II), we analyze the characteristics of the error bursts caused by inter-technology interference in section III. The ANN implementation and model selection are presented in section IV, where we also present our experimental results. Finally, section V concludes the paper and proposes possible future extensions.

II. Background

A. Wireless Technologies

In this section we briefly recall some key aspects of the MAC/PHY layers in WiFi, ZigBee and LTE that affect the power of cross-technology interference and the typical timings of transmissions and channel idle intervals.

Interference power. WiFi and LTE transmissions are typically performed at a maximum power of 15 or $20dBm$, while ZigBee transmissions can span in the range $[-25, 0]dBm$. LTE transmission power is modulated because of power control mechanisms, which are usually not implemented in WiFi and ZigBee. Additionally, each WiFi channel is 20 MHz wide and is spaced of 5 MHz from the adjacent ones. ZigBee channels have only 2 MHz of bandwidth with 3 MHz of inter-channel gap bands (i.e. the center frequencies maintain the spacing of 5 MHz from the adjacent channels). It follows that four ZigBee channels are entirely included in a WiFi channel. LTE center frequencies in ISM bands coincide with WiFi ones, with a typical bandwidth of 5 MHz (but bandwidths as small as 1.4 MHz are possible).

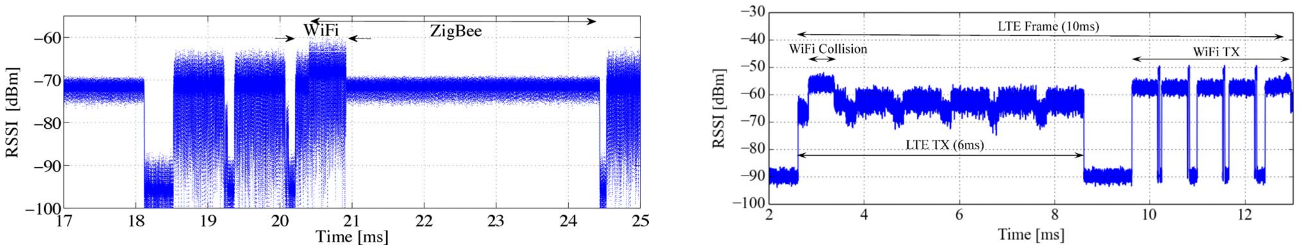
Transmission times. Since the three technologies have been defined for different applications, the frame size, the data rates and the channel access units considered by the standards are quite different. For WiFi and ZigBee, channel access is performed on a per-packet basis, i.e. transmission times correspond to the time required for completing the transmission of a packet (or an aggregation/fragmentation of packets). ZigBee packets are small, with a maximum payload of only 127 bytes. Bytes are organized into 4-bit symbols that are mapped into 16 pseudo-random sequences of 32-chip transmitted at 2 Mchip/s (i.e. 250 Kbps), which correspond to a frame transmission interval of about 4 ms for the maximum frame size. WiFi frames are much longer, with a maximum frame size of 2358 bytes and multiple OFDM modulations and coding schemes available (from 6 Mbps up to 54 Mbps, which lead respectively to a maximum transmission time of about 3.2 ms and 0.37 ms). For LTE, the channel access is performed on the basis of resource block allocations, which are organized into subintervals lasting a fixed time of 1 ms within a frame of 10 ms. Packet transmissions are achieved by scheduling a given set of resource blocks in one or multiple consecutive frames. Although the total number of resource blocks used for each packet depends on the employed data rate and multiple rates are available (up to 25.2 Mbps for 5 MHz of bandwidth with 300 sub-carriers, 64-QAM modulation, and a symbol time of 71.4 μs), the channel occupancy time in each channel access is fixed according to the LTE frame structure.

Intervals between transmissions. Different channel access schemes are employed in WiFi, ZigBee and LTE for unlicensed bands. WiFi and ZigBee are mostly based on random access although channel sensing is performed with different granularity: ZigBee spends 128 μs for detecting the channel activity and 192 μs to switch from reception to transmission mode. Since WiFi slots are much shorter (9 μs), if a WiFi transmission is originated during this switching time, it cannot be detected by the ZigBee node. Figure 1-a shows a channel occupancy trace acquired by means of a USRP node in a network in which a WiFi node coexist with a ZigBee one. In the figure we clearly observe that each transmitter is characterized by a specific RSSI value and frame transmission time: WiFi frames occupy the channel for less than 1 ms with a RSSI value of -65 dBm, while ZigBee frames last 4 ms with a RSSI value of -72 dBm. The figure also shows that a ZigBee transmission can overlap with WiFi, in case a WiFi frame is transmitted during the time spend by ZigBee for switching from sensing to transmission mode.

LTE transmissions in licensed bands are organized into frames of 10 ms that start at regular time intervals. For operating in unlicensed bands, two different adaptations have been envisioned: employing duty cycles for periodically suspending frame transmissions, while keeping the synchronization of time instants at which frame transmissions can start (LTE-U); employing listen-before-talk before transmitting each frame (LTE-LAA). In this second case, when the medium is sensed as busy, the deferral time is given by a fixed time of 10 ms for maintaining the synchronization of frame starting times (with the so called FBE mechanism) or it is given by a random slotted deferral time compensated by a varying channel occupancy time (with the so called LBE mechanism). In our work, we emulate both the LTE-U and LTE-LAA approach, by assuming that LTE frame transmissions can start only at regular time intervals. Figure 1-b gives an example of the interaction between an LTE-U transmission with 6 active and 4 silent subframes (i.e. 6 ms on and 4 ms off) and a WiFi station which tries to access the same channel: the figure shows that WiFi packets can collide with LTE and that part of the channel time is wasted due to the consequent backoff.

B. Artificial Neural Networks

ANNs are a class of powerful machine learning tools that can be used to solve classification and regression problems. They can be distinguished in two types of architectures, depending on the types of connection between neurons: in the *feedback* architectures, the presence of connections between neurons of the same layer or between neurons of the previ-



(a)

(b)

Fig. 1. Interference between technologies: temporal trace (RSSI samples) of WiFi-ZigBee (a) and WiFi-LTE collisions (b).

ous layer realizes a feedback connection. In the *feedforward* architectures, the connections between the neurons do not allow feedback between layers, and the signal is transmitted only to the neurons belonging to the next layer. In this article we used a neural network model called Multi-Layer Perceptron (MLP), a widely used *feedforward* ANN, formed by one input layer, one or more inner layers called "hidden", and a layer of output neurons. The training of the network is usually done with an algorithm called *back propagation*, which is basically divided into two phases. In the first phase, called forwarding, controlled inputs are applied to the network, causing the activation of the neurons of the input layer. The signal propagates to the next layers, finally reaching the output neurons. The error between the desired output and the obtained result for each neuron is then computed. In the second phase, called backwarding, the error value is propagated backward and the weights of each link accordingly modified with an optimization method, which aims to minimize the output error with respect to all the network weights. Finally, the network "model selection" is achieved by choosing between a set of *hyper-parameters*. The hyper-parameters define the structure of the network, such as the number of hidden layers and the type of activation function, and the configuration of the training algorithm, such as the regularization factor and the learning rate. Comparing the performance scores of all possible combinations of parameters, we finally select the model giving the best score.

III. Error Analysis in WiFi Receivers

A. Monitoring Receiver Errors

In [15], we have shown that WiFi cards receiving non-WiFi modulated signals generate error patterns significantly different, in terms of occurrence probability and time intervals between consecutive errors, from the ones generated by collisions with other WiFi transmissions. In presence of wide-band noise and exogenous interference signals, WiFi receivers demodulate a sequence of completely random bits and try to interpret these bits according to the format of WiFi frames. Being all the bits random, the probability of having a specific error heavily depends on the format of the expected frame.

Most commercial WiFi cards track the occurrence of different *receiver events*, such as the start of a synchronization trial, the detection of wrong PLCP, the end of a frame transmission, etc., by means of specific counters implemented in internal registers. As a reference WiFi receiver, we considered a WiFi card (namely, Broadcom bcm4318) for which the card internal registers are documented and an interface for reading the register values is available [16]. For producing a temporal trace of the receiver events, storing the ordered sequence of event type and occurrence time, we implemented a monitoring process devised to sample at regular intervals the receiver registers. Indeed, the event occurrence cannot be detected by the card host as an interrupt signal, but needs to be indirectly identified by comparing the state of the receiver registers in consecutive sampling times.

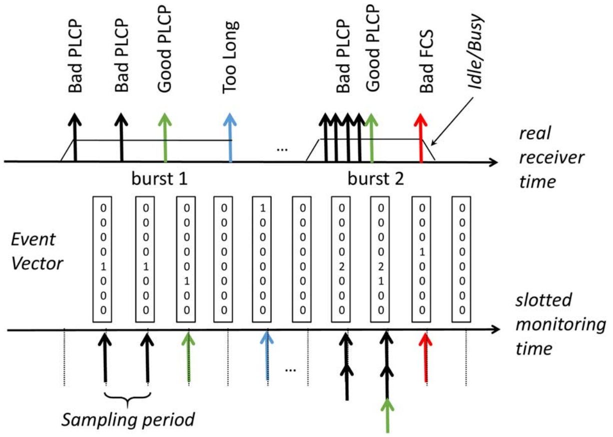
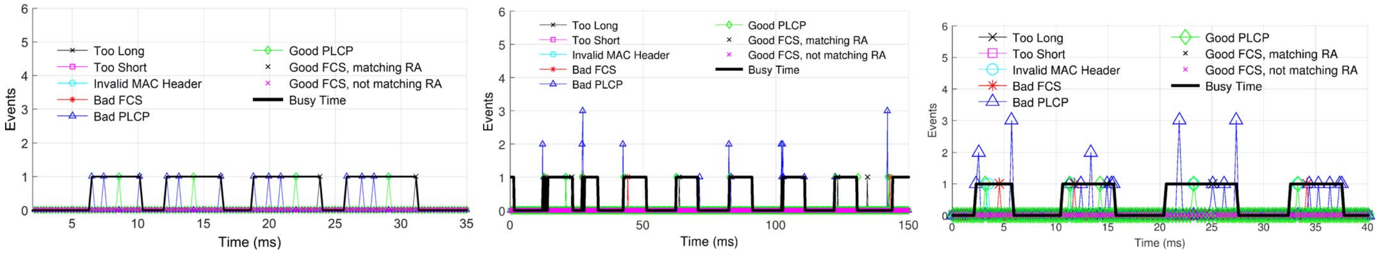


Fig. 2. Mapping between a real trace of receiver events and the time-slotted vectors generated by the monitoring process.

We set a sampling interval equal to $250\mu s$ as a trade-off between detection delay and tracking complexity, while avoiding the overloading of the card to host interface. Because of the periodic sampling, multiple receiver events can occur in the same monitoring interval. Event samples are represented by a vector of eight components, whose value represents the counter of each different event type. We also sampled another card register, called busy time register, which does not track the occurrence of receiver events but rather the cumulative time during which the receiver remains active. The differences among consecutive values of the busy time register can be mapped into a logical idle/busy state of the channel as observed by the receiver.

Figure 2 shows the operation of our monitoring process: a real trace of receiver errors is mapped into a time series of event vectors, in which we can easily recognize consec-



(a)

(b)

(c)

Fig. 3. Bursts of receiver events corresponding to the reception of ZigBee, Microwave and LTE-U interference respectively.

utive *error bursts* due to the same interfering transmission. Error bursts can be originated for many different reasons: for example, a checksum failure can follow the detection of a good PLCP, or multiple (failed or not) synchronization trials are performed after a bad PLCP event. The total number of receiver events in a burst depends on the duration of the interfering transmission and on the receiver implementation, i.e. on the reset time required by the demodulator for performing consecutive synchronization trials. Each burst can be delimited by observing the time interval elapsed from the previous and next events, and/or by considering the channel transitions from idle to busy and from busy to idle as delimitation times.

B. Temporal Analysis

Testbed. For our experiments, we set up a testbed at the University of Palermo and placed a monitoring WiFi node together with heterogeneous interfering sources. Four different interfering sources have been considered: a ZigBee transmitter, a LTE transmitter, a WiFi transmitter and a microwave oven. All nodes have been set to a few meters distance between each other and the transmitting nodes are programmed to work on different interfering and non-interfering channels. ZigBee nodes used in our testbed are based on Microchip MRF24J40 transceiver. The ZigBee frames are transmitted at 250kbps with a length of 127 bytes. WiFi transmitter has been implemented by using the same Broadcom card used by the WiFi monitoring node, with a frame length of 1500 bytes transmitted at 24 or 36 Mbps. The LTE-U transmitter, instead, was implemented on a SDR platform based on USRP B-210 and the srsLTE framework [17]. We considered a downlink interfering stream with 5 MHz of bandwidth and 300 subcarriers, centered on channel 11. Following the standard, the whole frame allocation time is $10ms$ composed of 10 subframes. The frame structure has been organized introducing silent intervals and a fixed sub-frame pattern, with mask [1,1,1,1,1,1,0,0,0,0] where 1 indicate transmission allowed and 0 transmission denied.

Results. Figure 3 shows three traces of receiver events when receiving ZigBee, Microwave and LTE-U interference. Figure 3-a, for example, shows four ZigBee packets, with error events spaced approximately $1ms$ from each other. Figure 3-b shows the error events caused a Microwave oven. From the figure, it can be clearly recognized the periodical radiation pattern of the oven, with 10 ms of activity and 10 ms idle. During radiation, channel is sensed as busy by the WiFi node, but error events are pretty different from the ones caused by ZigBee transmissions, since they are concentrated at the beginning and at the end of the radiation interval (rather than being continuously repeated). This can be due to the power-on and power-down ramp of the Microwave, being the demodulator unable to work when the radiation power is stable. Finally, figure 3-c shows the receiver events in presence of LTE transmissions. Under this interference source, the WiFi receiver behavior resembles the ZigBee interference with the granularity of consecutive synchronization trials equal to regular intervals of 1 ms. However, occasionally, some events are closer to each other. We also observed, the occurrence of the first synchronization trial is not always synchronized with the activation of the channel busy register: for example, in the figure at time 20 ms the busy channel state switches to 1, while the first event vector with non-null components (namely, three Bad PLCP events) are revealed after 2 ms.

IV. Interference detection

A. Features extraction and normalization

The experimental results presented in the previous section show that, although all non-WiFi interfering signals generate the same type of errors with similar statistics, their temporal analysis can be exploited for discriminating among different interfering sources. From the qualitative description of figure 3, it clearly emerges that several features can be exploited for such a discrimination, such as:

1) *the number of events* generated by the monitoring process during the same interfering burst, which depends on the interfering power, with an higher number of synchronization trials performed in case of LTE-U signals;
2) *the length of the error burst*, delimited by means of the correlation between the error vectors and the channel busy register, which depends on the transmission time of the interfering source;
3) *the temporal gap between consecutive errors* within the same burst, which might be symptom of power ramp effects (e.g. for the Microwave oven).

We propose to classify the interference sources through an MLP neural network with one hidden layer because this already provides good results and is computationally less expensive than networks with multiple hidden layers. As

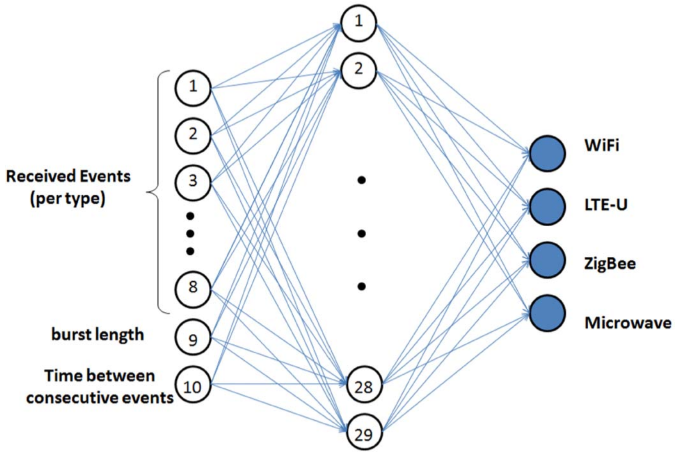


Fig. 4. Structure of the MLP neural network used in our experiments.

Features	value
Too Long	0
Too Short	0
Invalid Mac Header	0
Bad FCS	1
Bad PLCP	4
Good PLCP	1
Good FCS, matching RA	0
Good FCS, not matching RA	0
Burst length	5370
Time between consecutive events	1100

TABLE I
INPUT RECORD RELATIVE TO THE LAST ERROR BURST OF FIG. 3-C.

depicted in figure 4, features in the input layer are organized in 10 neurons (8 representing the counter of 8 types of reception errors [15], one for the error burst length in μs, one representing the maximum distance between two consecutive events in the same burst in μs). For example, Table I shows the input features generated by the last LTE-U error burst shown in 3-c. In the output layer, instead, we will have 4 neurons, each of them mapping the relevant interfering technology (WiFi, ZigBee, Microwave, LTE-U). The MLP network was implemented in Python using the *scikit-learn* machine learning library [18], trained using back propagation with a Cross-Entropy loss function. Since MLP is sensitive to work on normalized data, i.e. on features of Gaussian distribution with zero mean and unit variance, we preprocessed our data by removing the average value and dividing the values by the feature's standard deviation. The dataset was randomly divided into two parts (using the *train_test_split* function of scikit-learn), the training set and the test set: the first one is used for training and validating the neural network, the second one is used for evaluating the classification accuracy. We considered a training set of 4716 samples (equally distributed between LTE-U, WiFi, ZigBee and Microwave oven), while the test set was composed of 2020 samples. Each sample is constituted by a vector of ten features associated to the interference that caused it (e.g. Table I). From the point of view of the library it is interpreted as a float vector, with ten features (see table I). Finally, the hyper-parameters of the network, i.e. the number of neurons in the hidden layer, the activation function and the regularization factor have been studied in the Model Selection phase, as discussed in the following sub-section.

Solver	Accuracy	Training time	Iterations
SGD with constant learning	93.8%	100.54 s	276
SGD with adaptive learning	93.8%	98.90 s	276
L-BFGS	98.5%	7.48 s	305
Adam	96.1%	18.56 s	276

TABLE II
OPTIMIZATION SOLVERS WITH RELATIVE TRAINING TIMES.

Function	Accuracy
Identity	87.6%
Logistic	98.5%
ReLU	98.2%
tanh	98.1%

TABLE III
AVERAGE ACCURACY OBTAINED BY DIFFERENT ACTIVATION FUNCTIONS.

B. Model Selection

The model selection phase consists in comparing the performance obtained by changing different hyper-parameters, and choose accordingly the hyper-parameters that maximize the classification accuracy. To avoid the overfitting problem, we carried out a "k-fold" cross-validation with k = 10: we divided the training set into 10 equal parts and, at each step, one sub-sequence of the data set was used to evaluate the accuracy of the model trained with the remaining nine sub-sequences. We used the GridSearchCV function of *scikit-learn* to carry out an exhaustive "grid" search over the space of hyper-parameters considered in our analysis, and performed a k-fold cross-validation for each obtained model. Specifically, the space of hyper-parameters was configured by considering the following factors:

1) *solvers:* L-BFGS, adam, SGD with constant learning rate, SGD with adaptive learning rate;
2) *number of neurons in the hidden layer:* from 1 to 50;
3) *regularization factor "alpha" (L2 penalty):* 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7};
4) *activation function:* identity, logistic, tanh, ReLU.

For solvers adam and SGD the initial learning rate was set to 10^{-3} (default value in *scikit-learn*), which controls the step-size in updating the weights. SGD was set with a nestorovs momentum of 0.9, while in adam the exponential decay rate for estimates of first and second moment were set to β_1 = 0.9 and β_2 = 0.999. All solvers have tolerance *tol* = 10^{-4}. The solvers iterate until convergence (determined by *tol*) or up to a maximum number of iterations (never reached in our experiments).

In Table II it is shown the average accuracy, the time required for training the weights of the optimization algorithms and the number of iterations until convergence. The optimization were run on a laptop PC with dual core 1.8 GHz CPUs and 4 GB of RAM. It is clear that L-BFGS method converges faster and with higher accuracy. Figure 5 shows that, for a given configuration of the other hyper-parameters, increasing the number of neurons in the hidden layer improves the accuracy until a limit value of about 98.5%.

Tables III and IV show the performance achieved with different activation functions and regularization factors. In

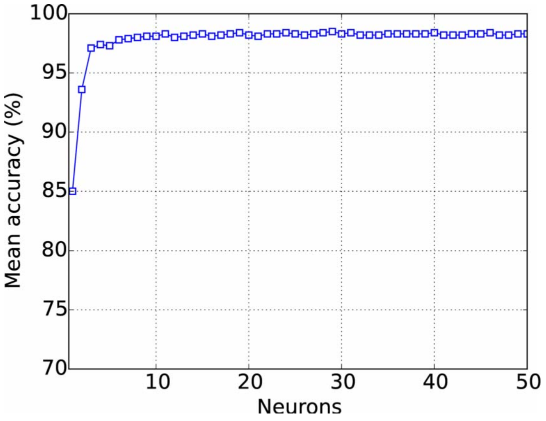


Fig. 5. Average accuracy versus the number of neurons in the hidden layer.

Alpha:	0.1	0.01	0.001	10^{-4}	10^{-5}	10^{-6}	10^{-7}
Accuracy:	98.5%	97.9 %	97.5%	97.7%	97.9%	97.8%	97.9%

TABLE IV
AVERAGE ACCURACY OBTAINED BY VARYING THE REGULARIZATION FACTOR.

particular, the logistic function reaches a higher accuracy compared to other activation functions, while the optimal regularization factor alpha was 10^{-1}. The final hyper-parameters derived by the model selection phase result in the MLP architecture shown in figure 4, where we omit the bias node for the sake of simplicity, with 29 neurons in the hidden layer.

C. Classification performance

After identifying the best hyper-parameters, we trained the network on the entire training set and evaluated the classification accuracy on the test set. To this purpose, we used a test set of 2020 burst samples (505 samples per class) representative of the four categories WiFi, ZigBee, Microwave and LTE-U. Table V shows the confusion matrix of the classifier, which obtains an average accuracy of 98.6%. The few errors are between ZigBee and LTE-U, because of the similarity of the error burst, as shown in 3. Finally, to verify the robustness of the model, we evaluated the classifier on the entire dataset composed of 67653 elements. Table VI shows that the classification performance is maintained even considering such a larger dataset, confirming the excellent results shown on the test set.

V. Conclusions and Future Work

In this paper, we presented a novel classification scheme for detecting ZigBee, LTE-U or microwave oven interference only using commodity WiFi cards. The idea is to exploit the error events caused by cross-technology interference on the WiFi node. Based on such error signals, we developed and optimized an MLP neural network to automatically classify the source of interference. After selecting the most appropriate model and training the network on a training set, we then tested the classifier with a limited number of samples and finally run the classifier on a large dataset. The result is quite impressive, reaching an average accuracy of almost 99%.

	WiFi	ZigBee	Microwave	LTE-U
WiFi	**100.0**	0.0	0.0	0.0
ZigBee	0.0	**97.5**	0.4	2.1
Microwave	0.0	0.0	**100.0**	0.0
LTE-U	0.0	2.2	0.7	**97.0**

TABLE V
CONFUSION MATRIX FOR THE TEST SET.

	WiFi	ZigBee	Microwave	LTE-U
WiFi	**100.0**	0.0	0.0	0.0
ZigBee	0.0	**98.2**	0.4	1.3
Microwave	0.0	0.0	**100.0**	0.0
LTE-U	0.0	3.4	0.6	**96.0**

TABLE VI
CONFUSION MATRIX FOR THE ENTIRE DATASET.

Although in this paper, the focus was to identify the interference caused by ZigBee, WiFi, microwave and LTE-U, the proposed approach could be easily extended to additional interfering technologies operating in the ISM band, e.g. Bluetooth or cordless phones.

References

[1] http://www.3gpp.org/ftp/Information/WORK_PLAN/Description_Releases/Rel-13_description_20150917.zip

[2] http://www.3gpp.org/ftp/Information/WORK_PLAN/Description_Releases/Rel-10_description_20140630.zip

[3] S. Rayanchu, A. Patro, and S. Banerjee. Airshark: detecting non-WiFi RF devices using commodity wifi hardware. In Proc. of IMC 2011.

[4] S. Pollin, I. Tan, B. Hodge, C. Chun, and A. Bahai. Harmful Coexistence Between 802.15.4 and 802.11: A Measurement-based Study. In Proc. of CrownCom, 2008.

[5] R. Gummadi, D. Wetherall, B. Greenstein, S. Seshan. Understanding and Mitigating the Impact of RF Interference on 802.11 Networks. In Proc. of ACM SIGCOMM '07, Pages 385-396.

[6] J. Huang; G. Xing; G. Zhou; R. Zhou. Beyond Co-existence: Exploiting WiFi White Space for ZigBee Performance Assurance. ICNP, 2010.

[7] X. Zhang, K. G. Shin. Enabling Coexistence of Heterogeneous Wireless Systems: Case for ZigBee and WiFi. In Proc. of ACM MobiHoc '11.

[8] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis. Surviving Wi-Fi Interference in Low Power ZigBee Networks. In Proc. of SenSys 10, pages 309-322, 2010.

[9] A. M. Cavalcante et al., Performance Evaluation of LTE and Wi-Fi Coexistence in Unlicensed Bands, 2013 IEEE 77th Vehicular Technology Conference (VTC Spring), Dresden, 2013, pp. 1-6.

[10] Yubing Jian, Chao-Fang Shih, Bhuvana Krishnaswamy, Raghupathy Sivakumar. Coexistence of Wi-Fi and LAA-LTE: Experimental Evaluation, Analysis and Insights. IEEE International Conference Communication Workshop (ICCW), 2015

[11] P. De Valck, I. Moerman, D. Croce, F. Giuliano, I. Tinnirello, D. Garlisi, E. De Poorter, B. Jooris, Exploiting programmable architectures for WiFi/ZigBee inter-technology cooperation, EURASIP Journal on Wireless Communications and Networking, Vol. 1, pp. 1–13, 2014.

[12] K. Lakshminarayanan, S. Sapra, S. Seshan, and P. Steenkiste. RF-Dump: An Architecture for Monitoring the Wireless Ether. In Procs. of CoNEXT 09, Dec. 2009.

[13] O. Zakaria. Blind signal detection and identification over the 2.4 GHz ISM band for cognitive radio. In MS Thesis USF09

[14] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty. Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey. Comput. Netw., 2006.

[15] D. Croce, D. Garlisi, F. Giuliano and I. Tinnirello, Learning from Errors: Detecting ZigBee Interference in WiFi networks, in Proc. 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET) 2014.

[16] http://bcm-v4.sipsolutions.net/802.11/Registers/

[17] https://github.com/srsLTE/srsLTE

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau. "Scikit-learn: Machine Learning in Python". Journal of Machine Learning Research Vol. 12, pp. 2825-2830, 2011.