AGC Index Prediction Using ML on KW45 Board

Morgane Joly
Université Caen Normandie, ENSICAEN
CNRS, Normandie Univ, GREYC UMR 6072
F-14000 Caen, France
morgane.joly@unicaen.fr

Éric Renault
LIGM, Univ. Gustave Eiffel,
CNRS, ESIEE Paris
Noisy-Le-Grand, France
eric.renault@esiee.fr

Fabian Rivière
NXP Semiconductor
Caen, France
fabian.riviere@nxp.com

***Abstract*—In the ISM band, even with coexistence techniques to reduce the impact of interference, over-consumption is inevitable due to concurrent emission in adjacent channel. As a result, packet are remitted to ensure data delivery. Flexible countermeasures are needed to ensure good signal quality, especially for low power protocols, such as Bluetooth Low Energy (BLE), more sensitive to those degradation. Our study is based on a test bench that reproduced real use case. Using receiver internal metric (RSSI, SNR, LQI, Access address match, AGC gain index), it shows that using the prediction of internal adaptor of the receiver (Automatic gain control (AGC) index) divides by four the PER and predicts poor reception in 50% of the target cases. This paper presents a comparison of four machine learning models and an in depth analysis of the performance of a bagged tree reaching an accuracy of 70.8% on training set.**

***Index Terms*—Cognitive Radio, Machine Learning, BLE, AGC index, Coexistence**

I. Introduction

Bluetooth Low Energy (BLE) operates in one of the most used frequency bands in the Internet of Things (IoT), the Industrial, Scientific and Medical (ISM) band and is affected by the presence of other wireless technologies such as wireless local area networks (WLAN).

In the case of the BLE radio system, only three advertising channels and nine data channels out of thirty-nine are free from direct interference from concurrent Wi-Fi operations.However, due to intermodulation products causing interference in the adjacent channel, the signal quality of the channels, that are *a priori* free of any interference, is impacted (Fig. 9 presents the measures done on a BLE signal interfered by a Wi-Fi at 12MHz offset, exposing perfectly the sensitivity limit).

Therefore, this overlap leads to high degradation of the signal, it ahas impact on latency, in [11], the authors highlight the latency increase during devices discovery procedures correlated with the number of devices of the same protocol and Wi-Fi. Also, the instability of the signal leads to inaccurate data collection, due to packet loss, but also, to sensor temporal de-synchronisation [9]. It impacts the data collection which can be critical for real-time application or for sensor fusion-based application and even in-situ research.

Countermeasures are in place but due to the augmentation of the spectrum more flexible countermeasures are need. Cognitive radio address this problem according to two main approaches :

- Through analysis, the targets use software-defined spectrum techniques to improve spectrum detection performance and cooperative detection. The main challenge with these techniques is to make accurate decisions about spectrum availability, detection time, detection frequency and uncertainty in the ambient noise. [5] [4]
- Trough receiver reconfiguration. The goal of the developed algorithm is to dynamically adjust the metric parameters to improve the network performance. The main challenges of this second method are the complexity of the algorithm, the limited convergence time and the ratio between the energy consumed by the algorithm and the energy gain allowed by the algorithm. [14] [2]

Our previous work [7] provided an initial assessment of the signal quality metrics use to overcome Wi-Fi interferences in a controlled environment. Specifically, we demonstrated that the Bagged Tree is able to predict the Automatic Gain Control (AGC) index to improve significantly the native Packet Error Rate (PER) by 10 percentage points in a simulated interfered environment.

The aim of this study is to evaluate the effect of predicting the AGC index on the packet reception without directly accessing this information using ML. To this end, we use an NXP KW45 board implemented with a ML model, a 10-packets history Bagged Tree, on a semi-hosted test bench (see Fig.1).

The paper is structured as follows. Sec. II introduces needed knowledge about BLE receivers, Sec. III, describes the data collection and the feature extraction process. Sec. IV discusses the obtained results. Finally, we conclude in Sec. V.

II. Radio Frequency Background

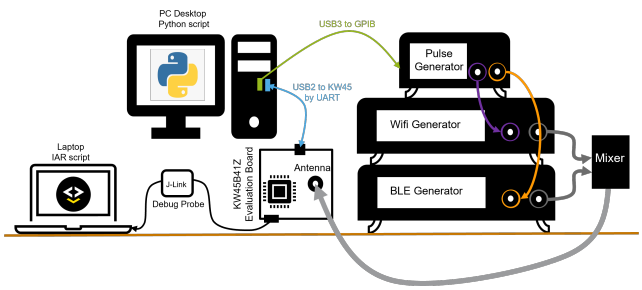


Fig. 1: Test-bench organisation

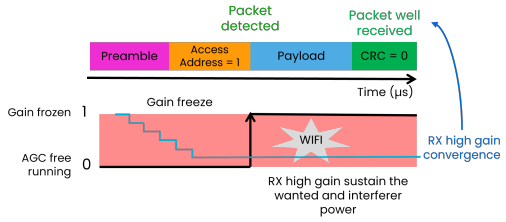
Even if 2.4 GHz protocols could be perfectly synchronised to avoid each other and spread communication evenly across the spectrum with frequency hopping, receiver non-linearity limit effective robustness against jammers. They can be mitigated by silicium design at the cost of a range/power trade-off ; the receiver reduces LNA gain and filters out the interferer to maximise the desired packet level, while removing as much of the interferer level as possible at the demodulator input. However, this makes the demodulator become less efficient to receive distant signal because gain reduction protects the receiver (avoiding saturation) at the expense of sensitivity degradation (corresponding to range reduction). The receiver monitors the internal nodes by means of peak detectors and received signal strength indicators like wideband RSSI, narrowband RSSI. Then, the AGC algorithm adjusts the gain in the receiver chain.

Amplifier non-linearities produce intermodulation products components that are proportional to the noise level during low-noise amplifier (LNA) gain and overlap the desired signal.These intermodulation products are proportional in dB (2dB/dB for IM2 and 3dB/dB for IM3) to the spurious level during amplification by the low noise amplifier (LNA), resulting in the introduction of interferences on the desired channel. IM3 will contaminate the spectrum around the wanted signal, while IM2 will overlap the spectrum of the wanted signal.

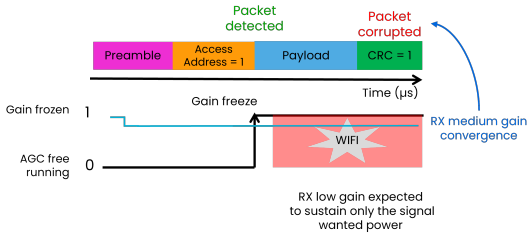
To protect the signal reception, countermeasure are in place, Time-Division Multiple Access (TDMA) coupled with frequency hopping better balance the band occupation, black and whitelisting of channel, but also measures at the physical level (PHY) make filters more efficient, such as the (Automatic gain control) AGC algorithm. It is used to avoid saturation during payload reception by adjusting the receiver gain according to the received signal strength intensity (RSSI). The index is set at the beginning of packet reception, during the preamble, and frozen during the payload to avoid corruption of bit decoding (see Fig. 2a). The latter is efficient for interferer arriving before the freeze, but in the event where the interferer arrives after the gain freeze, the gain is no longer up to date regarding the signal power in the targeted channel(see Fig. 2b), it increase the chance of packet corruption. The right gain is also crucial for it energy efficiency, the more the AGC is reduced, the less the receiver consume energy.

Four other internal metrics are used to indirectly access to the evolution of the interferer and the wanted signal combined :

- The Access Address Match event that detects the start of reception.
- The RSSI, which indicates the total power on the channel being monitored. It also adjusts the chain gain to ensure safe operation of the receiver.
- The Signal to Noise Ratio (SNR) gives an indication of the purity of the signal.
- The Link Quality Indication (LQI) quantifies the receiver's ability to decode the signal in the presence of ambient noise.



(a) Wi-Fi interferer arriving before the freeze.



(b) Wi-Fi interferer arriving after the freeze.

Fig. 2: AGC algorithm convergence and frozen period according the packet frame.

- The Cyclic Redundancy Check (CRC) word, which verifies the validity of the data in the payload.

III. AGC INDEX PREDICTION METHODOLOGY

A. Data collection

The Wi-Fi signal is intermittent, resulting in a pattern of BLE packets that are either jammed or clear. The selected jammer is an IEEE 802.11g Wi-Fi signal with a data rate of 54 Mbps (orthogonal frequency division multiplexing) and a modulation of 64 (quadrature amplitude modulation), with a data length of 1024 bytes. It is kept at a 12 MHz offset from the wanted signal. This overlap is the shortest distance that can be maintained between the BLE and Wi-Fi signals before full signal overlap occurs.

The index produced by the convergence of the AGC algorithm is used to label the data. Twelve numeric prediction classes are used within the range of 0 to 11, corresponding to the possible values of the AGC gain index. In addition, a non-numeric value of 'Not Received' is employed to indicate that the future packet cannot be well received. Therefore, there are 13 prediction classes.

B. Training and testing dataset

The training and test signals are generated with cross validation 70/30 from a source signal of 400 000 packets with a BLE wanted power and Wi-Fi interferer power in the range [-73,-1] dBm. The channel occupancy is set at 14% to match Wi-Fi duty cycle corresponding to a web browsing activity. It presents the greatest variety of interference periods compared to the other behaviours studied (online call, messaging and video streaming) [10]. We obtain a packet distribution exposed in 3, with a similar repartition of the class. It can be noted that some classes (10, 4 and 1) have an undersized

TABLE I: Model hyperparameter optimisation comparison.

Model	F1-score	Accuracy	Recall	Matthew	Size	Prediction time
Bagged Tree non optimised	71.0 %	**70.8 %**	**70.5 %**	**67.2 %**	800 MB	16.3 µs
Bagged Tree optimised	**71.5 %**	69.2 %	69.1 %	65.5 %	164 kB	5.2 µs
KNN non optimised	70 %	69 %	69.3 %	65.6 %	1.7 Go	4.4 ms
KNN optimised	70.8 %	70.3 %	70.2 %	66.9 %	290 MB	2.8 ms
Decision Tree non optimised	63.6 %	63.6 %	63.6%	59.5 %	14 MB	5.58 µs
Decision Tree optimised	67.5	66.3 %	66.4	62.4	**11 kB**	**2.5 µs**
Linear discriminant non optimised	68.1 %	67.8 %	67.8 %	64.4 %	335 kB	6 µs
Linear discriminant optimised	68.3 %	68.0 %	68.8 %	64.7 %	41 kB	**2.5 µs**

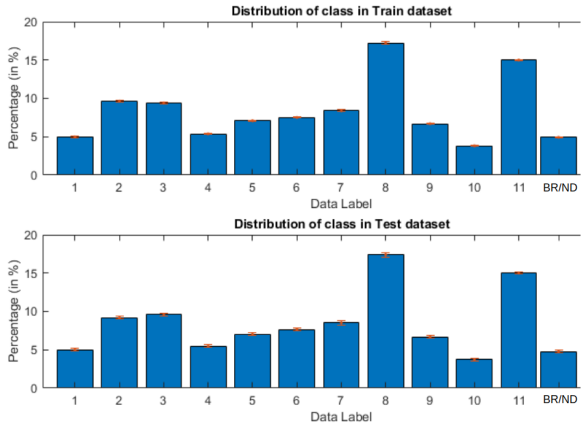


Fig. 3: Class balance between training and test set.

population, whereas other classes (8 and 11), have much more weight.

Since the end application has a memory footprint constraint, the data sampling for continuous data - the RSSI, the SNR and the LQI - is limited to three time events spread over packet reception. Metrics that provide unique information during packet reception - Access Address Found and CRC - are stored at the beginning and end of packet reception respectively.

C. Model selection and hyper-parameters optimisation

We have selected the tested models among the twenty-four algorithms proposed by the ML Graphical User Interface of Matlab and according to our previous works on the analysis of ML algorithms [6].

Despite its substantial memory requirements, the Bagged Tree has been incorporated into the panel due to its overall stability among the training and testing phases. To compare the ratio performance/resources needed a lighter version of trees, the Decision Tree is integrated as well. The memory footprint of K-Nearest Neighbours (KNN) is contingent on the size of the training dataset, and as such, it is prone to substantial energy consumption. Nevertheless, it is selected on account of its lightweight calculation in $\mathcal{O}(nd)$ with n the number of feature and d the number of data) and its accuracy performance, which is comparable to that of the Bagged Tree. Following a comprehensive evaluation of the memory and computational capabilities of KW45, the Linear Discriminant (LD) model is integrated to assess the performance of very light memory model.

The memory footprint analysis is done regarding the limit of the receptor electronic support which is fixed at 100 Ko for the KW45.

The hyperparameter optimisation is obtained by using the Random Search algorithm. The Grid Search algorithm ensure to study in details the evaluation parameter regarding the compromise between model accuracy and memory footprint

Finally, to evaluate the prediction time for a prediction, we use an AMD Rysen 95900HX processor. The experiment was run 10 times on different subsets of 500 samples using the Python flag function *time.time()*.

D. Off line analysis methods

AGC index algorithm works as a discretised scale, meaning that a packet is more likely to be received by indexes that are close together in the native system. If the native AGC algorithm converge for a value x, due to uncertainties in similar reception environment it may converge to a value $x \pm k$ with $k = 1$, but it is unlikely that a packet received in some condition with an index value of x is received with a value of $x \pm k$ with $k \geq 1$. For example packet well received with an AGC index of 3 could be also well received with an AGC of 2 or 3 and on the contrary the system would never converge for an AGC index of 6 and ensure the good reception of the packet. For the evaluation of the prediction model off-line test phase, it means that the $\pm k$ have to be include in the evaluation of the off-line performances.

We define this error as :

$$\Delta_{AGCindex} = |AGC_{predicted} - AGC_{native}| \qquad (1)$$

With $AGC_{predicted}$ being the predicted AGC index by the model and AGC_{native} being the native AGC index.

E. Real-time analysis methods

Throughout the previous experimentations, Bagged Tree has shown high and steady accuracy. Considering the reduction of the memory footprint presented in the previous section, it is a good candidate to embed on a reduced memory chip therefore it has been selected in priority selected to be tested on the testbench.

Since the learning method is off-line, the major concern by limiting the analysis of performance only based on the

learning performance, is that the improvement (or performance decrease) of the packets due to the prediction are not considered in the training data set. To evaluate the real performance of the models, it is necessary to evaluate the Packet Error Rate (PER) and Future Bad Reception Rate (FBRR).

We set up a testbench to allow the model to access in real time to real data from a KW45 Board. It was chosen because we have access to most of the target features without any hardware modifications. It is also a well-researched project with well-documented capabilities. It integrates a high performance Bluetooth Low Energy (version 5.3) with four data rates (125 kbps, 500 kbps, 1 Mbps and 2 Mbps), a 96 MHz Arm Cortex®-M33 core, 1 MB flash memory, 128 Kb SRAM and 88 Kb SRAM optimised for link layer support [1].

To perform and automate the measurements, we set up a Python-based test bench. The PC controls the power of the two RF generators – K-Sight CXG Vector Signal Generators. The pulse generator – K-Sight 33600A series – synchronises the latter to study the lack of AGC index adjustment in response to the late arrival of an interferer (see Fig. 1).

We evaluate the model by using the same testing scenario as the one used in the off-line analysis. The model is implemented on the validation PC, making the prediction with the data returned by the KW45. A buffer is engineered in the embedded software to store the data from the previous packet's reception according to the size of the sliding window.

To refine the analysis of the results of the QoS improvement, we have developed two additional measures, the global mean PER to capture the improvement due to the prediction of the gain index (see eq. 2) and the Future Bad Received Rate (FBRR) (see eq. 3).

The accuracy alone being not sufficient to estimate QoS improvements, in addition to usual ML evaluation parameters, we use the global PER metric :

$$Global_mean_PER = 100 \times \frac{ND + BR}{ND + BR + GR} \quad (2)$$

BR being the number of packet corrupted, GR being the number of the packet well received, and ND the number of packet whose access address has not even be detected.

Then, to evaluate the ability of the model to predict the proportion of well-predicted future bad packets, we have created the Future Bad Received Rate (FBRR) with the following equation :

$$FBRR = \frac{FBR}{BR + ND} \quad (3)$$

When a packet is predicted to be badly received (i.e. the result of the Bagged Tree is "Not Received"), the AGC index algorithm converges without intervention : the packet is received with the AGC_{native} due to the conception of the receptor. It allows us to effectively determine the prediction correctness represented by the FBR Rate (Eq. 3).

IV. Results

Based on the model training analysis in previous work, four models were trained on the data collected from the testbench.

A. Result of ML training and testing

1) Bagged tree and decision tree: The study of the different evaluation parameters of the Bagged Tree model gives us insights on the behaviour of the model according to the number of trees in the forest. Figure 4 presents the performance of a single Decision tree according all the classes. The results are grouped around the main diagonal which translate the tendency to make connections between classes : classes that are close together are more likely to be confused than classes that are far apart.

Combined in a Bagged Tree, the accuracy regarding the number of trees in the forest displays an asymptote 2-percentage points bellows the asymptote observed for the accuracy of the model without the hyperparameters optimisation. However, the F1-score remains equivalent and the confusion matrix (see Fig. 5c) shows that the prediction are grouped around the main diagonal, as the Decision Tree. It can be noted that, even though the Bagged Tree has better accuracy that a single DT, it significantly underperforms in classes 4, 6 and 10. It can be linked to the imbalance of the dataset, the way the Bagged Tree constructs its forest makes it more sensitive to this kind of imbalance.

The aggregation of weak learners makes the Bagged Tree powerful in the big picture. The way the tree algorithms build their structure has a huge impact on the memory the larger the number of trees and the heavier the model. The control of the model size is made by choosing the optimal combination of the tree depth, the number of splits at each node and the number of leaves. After this optimisation phase for a bagged tree with 15 trees, we have obtained a weak learner that reached 64% of accuracy with a size of 13 kB.

The model size and the prediction time behaviour according to the number of estimators are linear. We can determine that the graph presented in Fig. 5a presents an asymptote at 0.718. From there we determine from x=15 the curve does not evolve significantly (less than 1% of accuracy loss), so the optimal choice to ensure maximum performance and low memory footprint and prediction time (a 50% time decrease reduction (from 28.7 µs to 16.3 µs) and a decrease of 45 % of the memory footprint, from 1.6 Go to 800 MB) is to construct the forest with 15 trees (see Tab. I).

2) K-Nearest Neighbour: The optimisation of KNN takes into account the neighbours considered, the type of distance used and the type of algorithm used to compute the distance. For the model evolutional, the optimisation Python algorithm only consider the accuracy of the models. To include the memory footprint and the prediction time, a dichotomy method is used to find the optimal combination of parameters using Grid search.

Let N be the number of samples and D the dimensionality. After analysis, the hyperparameter that influences the most the size and the model calculation time is the partitioning of the data. It is interesting to note that the accuracy (and other performance metrics) are not impacted. Between the brute force and the other algorithms, the memory footprint doubles its size. Notably, the prediction time is less important

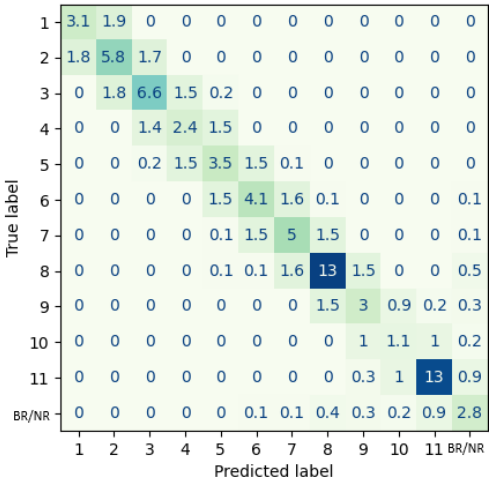


Fig. 4: Confusion matrix of Decision Tree.

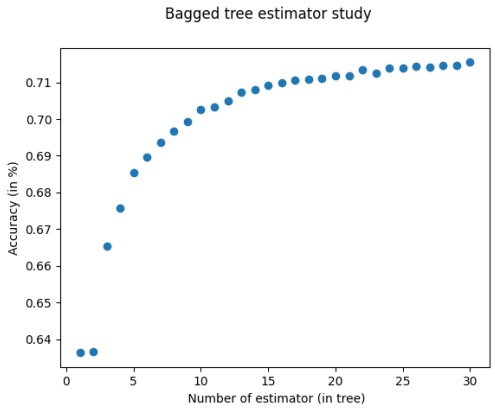
for the Ball tree and the KD tree : since for the KD tree the calculation cost of the nearest neighbour search can be reduced to $O(DNlog(N))$ and for the Ball tree it grows up to approximately $O(Dlog(N))$, and the brute force algorithm grows up to $O(DN)$. Other hyperparameters, like the distance calculations or the number of leaves have a marginal impact.

The tuning of the number of neighbours shows no impact on the memory. This was expected since the model consists in charging in memory the training dataset to compare the new value to make a prediction. The heavier the training set, the heavier the model. The prediction accuracy is stable compared to the model without optimisation but the prediction time is still really high after optimisation.

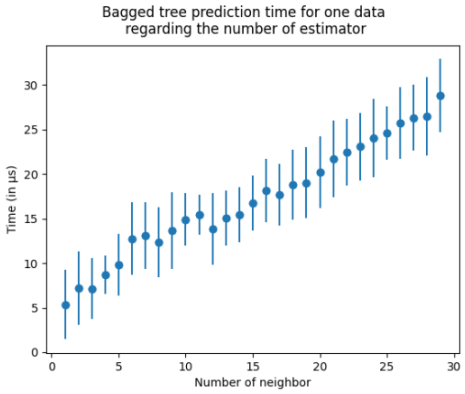
The study of the different evaluation parameters of KNN model gives insights on the behaviour of the model according to the number of neighbours needed to obtain the best performance. The accuracy presents an asymptote at 69 % and the prediction time according to the number of estimators is not impacted by the number of neighbours. If we compare the evolution of performance from 29 to 15 neighbours, the accuracy loss is negligible (from 69.3 % to 69 %), the time prediction is stable (4.4 ms) and the same goes for the memory footprint (1.7 Go) which is, as for the Bagged Tree model, too much for the selected hardware.

With a feature selection process like PCA or genetic algorithm, the size of the model can be more effectively reduced.

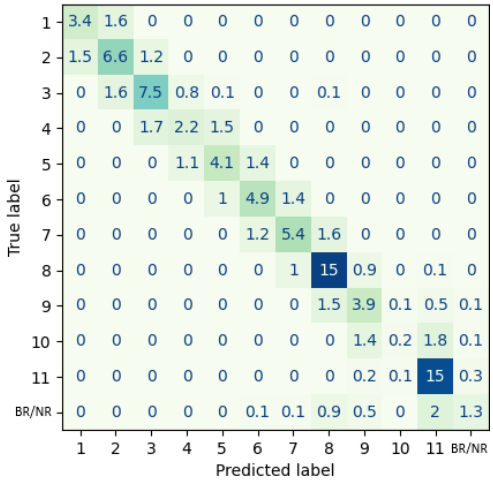
3) Linear Discriminant: LDA without any hyperparameter optimisation does not have an accuracy at the level of the Bagged Tree or KNN for a 10-packet sliding window. However, it is the most efficient in terms of memory footprint (335.7 kB) and execution time (6 µs). LDA performs well for the classes with the largest populations. In fact, for the classes 1, 4, 7 and 9, it even performs better than the Bagged Tree (see Fig. 7). Nonetheless, the predictions are less accurate for classes 2, 3, 5, 8 and 10. For the latter, the low accuracy can be explained by the fact that they have the fewest samples,



(a) Bagged Tree Accuracy according to estimator without hyperparameter of the weak learner.

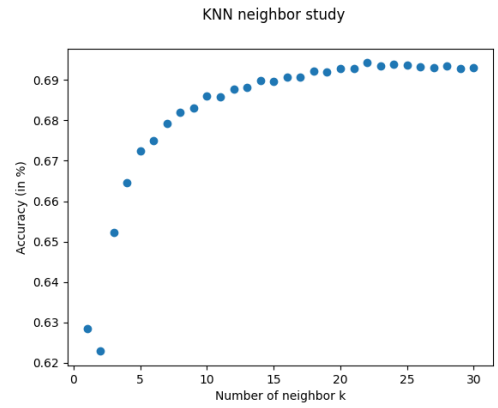


(b) Bagged Tree prediction time according to estimator without hyperparameter of the weak learner.

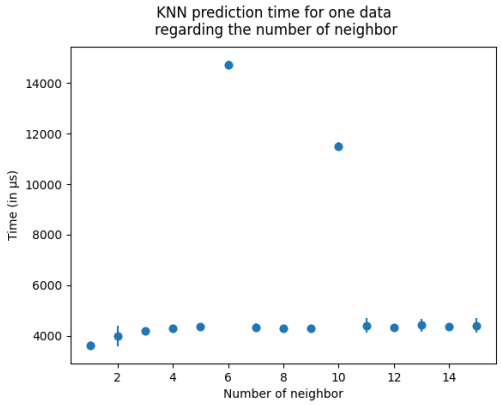


(c) Confusion matrix of Bagged Tree with 15 iterators.

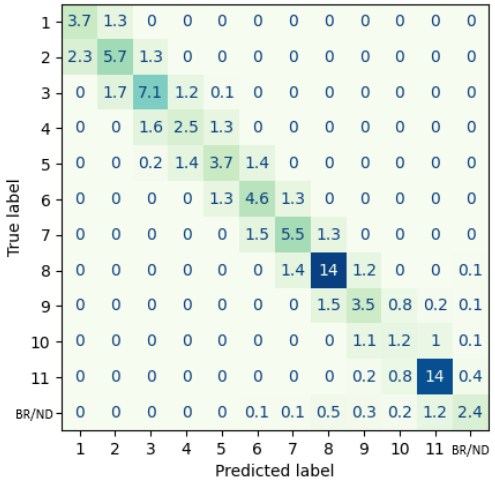
Fig. 5: Bagged Tree results with iterator number variation.



(a) KNN prediction accuracy vs number of neighbours.



(b) KNN prediction time according to the number of neighbor.



(c) Confusion matrix of KNN.

Fig. 6: KNN Results.

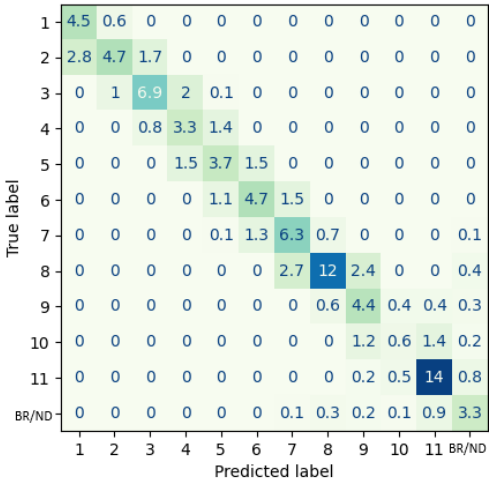


Fig. 7: Confusion matrix of Linear Discriminant.

which is consistent with the LDA's statistical methods.

LDA is based on the dimension reduction of the data while conserving as much as possible the relevant information. The hyperparameter optimisation lies in the choice of adequate shrinkage to improve the estimation of covariance matrices and the solver choice. The result of the Grid Search shows that the best combination is with the Oracle approximating Shrinkage (OAS) and the least squares solution solver which aims to minimise the sum of the squares of the overall error in the model. However, after comparison with the result of the two other solvers, its choice does not impact the performance metrics. Nonetheless, it severely influence the model memory footprint which makes the SVD solver the best choice.

B. Training Results discussion

Concerning the performance result on test dataset, the Bagged Tree presents an accuracy of 70.5%, 7 percentage point better than a single decision tree (see Tab. I). The F1-score shows a good balance between precision and sensitivity. However, its memory consumption is 29 times higher and its prediction time is three times higher than the one of the decision tree and it is sensible to the class imbalance. The linear discriminant offers a good balance between all the parameters analysed, especially for size and prediction time, although its accuracy is slightly lower than the Bagged Tree. The KNN seems interesting in terms of accuracy, but its prediction time is the highest. The possibility of reducing it by optimising the features need to be investigated. The Fig.8 summarize the repartition of errors according the $\Delta_{AGCindex}$. For all models, 93% of the packet well received are with a $\Delta_{AGCindex} \leq 1$ and from a distance $\Delta_{AGCindex} \geq 4$, there is a reduced amount of error. Thus, even if the accuracy could be improved, the ML models tends to predict coherent value regarding the targeted label which leads to strengthen the hypothesis that on real time the models could perform better as strict accuracy would suggest.

The hyperparameter optimisation reduces the size and the prediction time of all the models. Most improved their

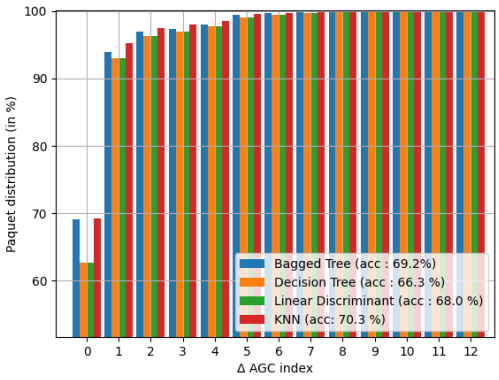


Fig. 8: Highlight of deviations from the native AGC index using cumulative sum (exclusion of the BR/ND class representing between 2% and 4 % of the dataset).

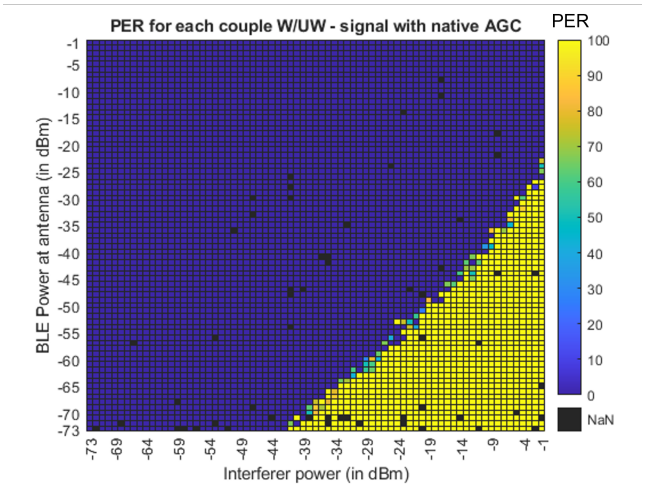


Fig. 9: Native AGC signal PER for each (BLE;Wi-Fi) power couple on the synthetic signal Test set.

evaluation parameter (KNN and Decision Tree). Bounding the size depth reduces the Bagged Tree, which presents the most drastic memory footprint reduction and multiplies the number of weak learners in the forest that improves the accuracy results. However, the selected hardware presents high RAM compared to other systems. In view of the solution adaptability, the Decision Tree model remains worth testing in real-time conditions. The same observation can be made for LDA, which presents equivalent results. KNN is the best performing model of the selection. Its prediction time and size do not suit the constraints of the selected hardware.

C. Bagged tree real time analysis

The real time performance of the Bagged Tree is evaluated on the same packet sequence used for the test set during the learning phase, except that the packets are generated in real time, which means that the predicted index is taken into account in the packet history. The results obtained show that the native global PER is reduced from 10% to 3% thanks to the implementation of the Bagged Tree index prediction. However, to explore the performance of the model in depth, the Normalised Mean PER for each pair (BLE Power;Wi-Fi Power) is preferred.

A simple graphical analysis shows the regression of the proportion of packets that cannot be received at all by the receiver. In Fig. 9, it can be seen that the distribution of the (P_{BLE};P_{Wi-Fi}) couple is binary : The "Yellow" area with the couple impacted by the Wi-Fi interferer corresponds to the area under the Minimum detectable Wanted signal versus Wi-Fi blocker level curve, sensitivity limit curve for short, while (P_{BLE};P_{Wi-Fi}) couples with a zero PER in the "Blue" area corresponds to the area over the sensitivity limit.

Lets the *GR* area be the area over the curve and the *(BR;ND) area* be the area under the curve.

The PER comparison between those areas shows that the average PER is divided by a factor of 3.4 (from 10% to 3%) when using the index prediction in the *(BR;ND)* area which is far more than we expected (see Table II).

TABLE II: PER results.

	PER (BR;ND) area	PER GR area
Native AGC	95%	0.02%
Predicted AGC	25.7%	0.16 %

Even if the Mean Normalised PER is low in the *GR area*, further investigation are needed to fully appreciate the impact on the packet reception.

83% of the (BLE power;interferer power) couples do not see any improvements nor performance decrease which 94% of this subset set are in the *GR area*. 15% of the couples showed improvement and 29% of this subset (corresponding to 5.5% of the global packet population) are always improved.

D. Future Bad Reception Results

The AGC index prediction method led to four situations :

- The AGC index predicted value is between 1 and 11 and the packet is effectively well received.
- The packet is predicted to be badly received, and the packet is effectively not received (because of no detection or error during the payload reception).
- The AGC index predicted value is between 1 and 11 and the packet is not received.
- The packet is predicted to be badly received, and the packet is well received.

As seen in Sec. IV, most of the packets are *well received*. Table III shows that 92% of the packets are well received thank to a good AGC prediction and only 1.5% of the packets are bad received or not detected due to bad prediction. The *Future Bad Reception* detection tends to overreact with 5.6% of the packets being predicted as a FBR, and 0.2% of packets are being effectively detected as a real FBRR.

To better understand the impact of Future Bad Reception, we refine our analysis by differentiating either packet belong-

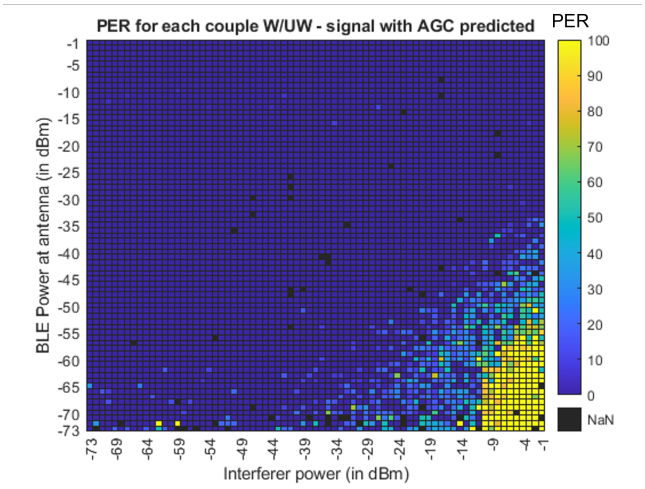


Fig. 10: PER for each (BLE;Wi-Fi) power couple on the synthetic signal Test set with the AGC predicted.

TABLE III: Predicted AGC value range versus the packet reception status

Packet status	GR	(BR;ND)
AGC in [1:11] range	92%	1.5%
FBRR	5.6%	0.2%

ing to the *GR area* and *(BR;ND) area*. Table IV shows that in the *(BR;ND) area*, among the 15.5% of the packets predicted as FBR, only 16% are true FBRR and this is only 10% of the packets that could be detected as Future Bad Reception. In the *GR area*, 94.8% of the packets are well received among which 5.16% are wrongly predicted as Future Bad Reception.

TABLE IV: Predicted index range ML according to the area.

Area	*(BR;ND) area*		*GR area*	
Packet Reception Status	GR	ND/BR	GR	ND/BR
AGC [1:11]	61.16%	22.4%	94.8%	0.16%
FBRR	13%	2.5%	5.16%	0%

It has been found that most of the FBRR prediction are located between from -10 dBm to 0 dBm for interference power and from -73 dBm to -55 dBm for BLE power, it has been found that 53% of the (BR;ND) packets have been successfully predicted.

V. Conclusion

BLE is a low-power protocol optimised to save energy, subject to many sources of sensitivity degradation on nearby channels. This leads to throughput degradation, connection failure and end device overconsumption.

This paper has shown that the prediction with an ML of the AGC index for the next reception slot use of an ML model increase the resilience of a BLE receiver subject to interference due to Wi-Fi co-temporal emission. In particular, our experiments with Bagged Tree showed that the PER is divided by four in the most impacted reception cases. The tested model is also able to detect future unfavourable slots and could be used to detect highly disturbed channels to improve the quality of the frequency hopping table assignment.

VI. Further research

Since the sensitivity to interferers can be reduced, the number of packet retransmissions is reduced too. Therefore, it impacts the energy consumption on the receiver and the transceiver side. The balance between the energy required to power up the microprocessor model need to be investigated to fully appreciate the energy gain.

References

[1] KW45B (radio) and KW45Z (MCU): Highly secure, bluetooth low energy 5.3 wireless MCU with CAN FD. Tech. rep., NXP Semiconductor, 2022.

[2] Enwere, P. I., Cervantes-Requena, E., Camuñas-Mesa, L. A., and de la Rosa, J. M. Using ANNs to predict the evolution of spectrum occupancy in cognitive-radio systems. *Integration 93* (2023), 102070.

[3] González-Cañete, F. J., and Casilari, E. A feasibility study of the use of smartwatches in wearable fall detection systems. *Sensors 21*, 6 (2021).

[4] Hanna, S., Dick, C., and Cabric, D. Combining deep learning and linear processing for modulation classification and symbol decoding. In *GLOBECOM 2020 - 2020 IEEE Global Communications Conference* (2020), pp. 1–7.

[5] Irsyad, F., Hendry, J., Slamet Saputro, J., Bambang Sasongko, A., and Harrys Gumay, M. Improved indoor localization mechanism for automated guided robots using bluetooth beacons. *Journal of Electrical, Electronic, Information, and Communication Technology (JEEICT)* (Apr. 2025), 34–38.

[6] Joly, M., Renault, É., and Rivière, F. Comparison of ai-based algorithms for low energy communication. In *Machine Learning for Networking* (Cham, 2023), É. Renault and P. Mühlethaler, Eds., Springer Nature Switzerland, pp. 1–11.

[7] Joly, M., Renault, É., and Rivière, F. Reducing ble energy loss in busy 2.4ghz band. In *Machine Learning for Networking* (Cham, 2026), F. Hacène, B. Selma, and R. Éric, Eds., Springer Nature Switzerland, pp. 85–96.

[8] Lee, Y.-S., Fanjiang, Y.-Y., Hung, C.-H., and Huang, Y.-S. Ble signal processing and machine learning for indoor behavior classification. *Sensors 25*, 14 (2025).

[9] Li, J., Quintin, E., Wang, H., McDonald, B. E., Farrell, T. R., Huang, X., and Clancy, E. A. Application-layer time synchronization and data alignment method for multichannel biosignal sensors using ble protocol. *Sensors 23*, 8 (2023).

[10] Palit, R. Anatomy of wifi access traffic of smartphones and implications for energy saving techniques. In *International Journal of Energy*.

[11] Qureshi, U. M., Umair, Z., and Hancke, G. P. Evaluating the implications of varying bluetooth low energy (ble) transmission power levels on wireless indoor localization accuracy and precision. *Sensors 19*, 15 (2019).

[12] Treurniet, J. J., Sarkar, C., Prasad, R. V., and De Boer, W. Energy consumption and latency in BLE devices under mutual interference: An experimental study. In *2015 3rd International Conference on Future Internet of Things and Cloud* (2015), pp. 333–340.

[13] Tsu Yian, L., Bahru, J., and Adib Sarijari, M. Optimal bluetooth low energy (ble) system deployment strategy prior to system application on the construction site. *Jurnal Teknologi (Sciences amp; Engineering)* (Sep. 2023), 161–168.

[14] Wang, D., and Yang, Z. An advanced scheme with decision tree for the improvement of spectrum sensing efficiency in dynamic network. In *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)* (2016), pp. 1288–1292.