Towards Accurate, Data-Driven and Lightweight Digital Twins for Wireless Networks

Samir Si-Mohammed, *Member, IEEE* Fabrice Theoleyre, *Senior Member, IEEE*

***Abstract*—Digital twins have recently emerged as a transformative approach to wireless networks, aimed at enhancing network efficiency. By replicating the physical network, they offer significant potential for implementing next-generation wireless networks. Digital twins make it possible to test decisions and evaluate their impact without disrupting network operations. However, digital twins are often based on simulations, and achieving high accuracy typically entails high model complexity and a detailed understanding of the environment. This underscores the need for data-driven modeling with continuous calibration from the radio environment. This paper proposes a self-calibration approach where each radio link is represented by its digital twin. Given the bursty nature of radio link quality, we introduce a self-adaptive method that selects the best prediction model on the fly for future short- and mid-term predictions. We validate our approach on a real network deployed in a testbed with very diverse links, on which we demonstrate the robustness of our self-adaptive model in predicting Packet Delivery Ratio (PDR) in both the short and medium term, giving the network sufficient time to anticipate and implement necessary reconfigurations.**

***Index Terms*—Digital Twins; Wireless Networks; Machine Learning; Regression; Data-oriented Modeling.**

I. Introduction

After revolutionizing Industry 4.0, Digital Twins (DTs) have recently garnered significant attention in the field of networking [1]. A Digital Twin is a real-time digital replica of a physical object or system, continuously reflecting its state. Conversely, any changes made to the Digital Twin are mirrored in the physical world. In wireless networks, radio links are inherently lossy, asymmetrical, and bursty [2], estimating link characteristics in a wireless network Digital Twin critical for effective network management.

Digital Twins can significantly aid in guiding network reconfiguration [3]. Since closed-form equations often remain impractical, approximated models are used to explore alternative network configurations. For instance, RouteNet-Erlang [4] leverages Graph Neural Network (GNN) models to estimate per-flow performance metrics. A Digital Twin (DT) enables the execution of *what-if* scenarios, predicting Key Performance Indicators (KPIs) as network parameters change. This capability allows the DT to support continuous network optimization, ensuring near-optimal performance at all times.

Digital Twins (DTs) play a vital role in predictive maintenance by estimating the evolution of Key Performance Indicators (KPIs) to forecast potential system underperformance. The prediction of Quality of Service (QoS) has been a key focus in recent research [5], aiming to leverage time series data from network measurements for forecasting future values. For example, this approach has been applied to Wi-Fi networks to anticipate performance fluctuations [6].

Although GNNs offer strong generalization capabilities, they provide limited observability for Digital Twins in wireless networks [7]. Consequently, simulation remains a widely used approach in wireless networking [8]. Discrete-event simulation, for example, allows for the detailed reproduction of per-packet actions such as transmission, collisions, and other network events. However, simulations often exhibit low accuracy and rely heavily on the underlying physical layer (PHY) models [9]. For a Digital Twin to be effective in wireless networks, it must remain *synchronized* with the real-world network. This requires continuously integrating real-world measurement data into radio propagation models to enhance accuracy and reliability.

Data-driven approaches, particularly Machine Learning (ML), offer promising solutions for self-tuning radio propagation models [10], enabling adaptation based on observed network behavior. However, accurately obtaining PHY-layer metrics, such as Signal-to-Interference-plus-Noise Ratio (SINR), remains a significant challenge. Therefore, a technology-agnostic method that operates independently of specific PHY-layer metrics would often be preferable.

Colosseum is a cutting-edge research platform for 5G and beyond [11] representing a pioneering step in this field. It leverages a large-scale emulation setup, where software-defined radio nodes simulate both client devices and base stations. To replicate a real deployment, the system can measure the channel response between each pair of nodes, and an FPGA is adjusted to digitally replicate the physical world. Colosseum is particularly useful for benchmarking or prototyping solutions in a controlled environment. However, we are looking for a more cost-effective solution that can be deployed for self-operated wireless networks.

Ideally, we aim to develop an approach that enables the construction of an "agnostic" DT for any wireless network, *i.e.,* regardless of the underlying communication protocols or traffic types. This DT should accurately reflect real-time events occurring within the physical network, such as the reception of a packet, and be capable of predicting future events with a reasonable degree of certainty. Furthermore, the DT should maintain a balance between complexity and scalability, ensuring it remains practical and efficient for various network sizes and configurations.

In this paper, we propose to redesign the PHY model of

This research was funded by the region Grand Est through the project ENGLAB.

Samir Si-Mohammed and Fabrice Theoleyre are with ICube Lab, CNRS / University of Strasbourg, Strasbourg, France (e-mail: simohammed@unistra.fr and fabrice.theoleyre@cnrs.fr).

the network simulator to create an accurate DT. Rather than relying on synthetic radio propagation models, we propose integrating real-world measurements to automatically calibrate the DT. The key contributions of this work are as follows:

1) We analyze the characteristics of radio links in real networks, highlighting the importance of per-link modeling;
2) we compare various forecasting techniques to evaluate their performance in predicting short- and medium-term behavior (a critical property of the DT);
3) we propose a self-adaptive method that dynamically selects the optimal model for each specific link.

II. Related Work

A. Digital Twins for Wireless Networking

Networked Digital Twins (NDTs) have recently emerged as a cutting-edge paradigm, driven by the rapid adoption of digital twin technology across various economic sectors. This concept has garnered significant attention from both academic and industrial communities, particularly among research groups investigating 6G networks [12], [13].

Many NDT models leverage simulation [14], as network performance evaluation has traditionally relied heavily on simulation techniques [8]. Combining simulation with emulation [15] enables the execution of identical code in both simulated and real-life environments, thus facilitating rapid prototyping and validation. For instance, integrating simulation and experimentation can help to calibrate energy consumption models in simulators using real-world measurements [14].

Despite these advancements, simulation, and emulation approaches often exhibit accuracy limitations, as their performance heavily depends on the fidelity of physical (PHY) layer models [9]. These limitations highlight the importance of incorporating data from real-world deployments to construct accurate and reliable NDTs for wireless networks.

Finally, Colosseum presents a notable example of a digital twin relying on a large-scale emulation platform [11]. It employs software radio nodes to represent User Equipment (UE) and Base Stations (BS), interconnected via a sophisticated Field-Programmable Gate Array (FPGA) fabric. This fabric emulates radio propagation using models fine-tuned to real-world conditions, requiring precise measurements of the channel transfer function—data that can be challenging to obtain in practical deployments.

B. Experimental Characterization of the Link Quality

Experimental studies are essential for bridging the gap between theoretical models and real-world deployments. Numerous works have investigated the performance of IoT networks across various technologies and testbed environments.

For example, Santi et al. [16] examine RPL over IEEE 802.15.4 networks, focusing on latency. Using the FIT IoT-LAB [17] testbed, they evaluate link delay and throughput under varying message sizes and transmission frequencies. Similarly, Brun et al. [18] explore the heterogeneity of FIT IoT-LAB deployments, analyzing phenomena such as external interference, multipath fading, and dynamic connectivity. Their study highlights the diversity of link behaviors observed at the Grenoble site of the platform.

Fraile et al. [19] present an in-depth experimental study quantifying the impact of MAC parameters on network performance (for both IEEE 802.15.4 and LoRa). It is important to be able to predict the performance of a network with a previously untested configuration, for example, by changing the parameters of IEEE 802.15.4 [20]

C. Radio Link Quality Forecasting techniques

Link quality prediction is critical for enabling a DT to anticipate future network conditions, but it presents significant challenges due to potential non-stationarity and the complexity of real-world environments. Numerous radio propagation models have been developed to address specific scenarios such as urban, vehicular, and tunnel environments [21]. However, these models require calibration with data from in-situ deployments to achieve realistic performance. The role of the DT is to synchronize its models with the physical world effectively.

The literature proposes various link quality metrics [22], highlighting the known challenges of radio links, including their lossy [23], asymmetric [24], and bursty [2] characteristics. Cerar et al. [25] classify links based on their RSSI values and quality (low/medium/high). However, a DT requires a more advanced approach that incorporates realistic estimations of packet loss distributions.

Cerpa et al. [26] compare Naive Bayes, Logistic Regression, and Artificial Neural Networks for predicting packet losses based on the physical layer indicators of the last sent packets. However, their method is limited to very short-term predictions, focusing on the next packet reception. In contrast, Benadji et al. [27] use neural network models, including LSTM, GRU, and CNN architectures, to forecast the evolution of global PDR across the network. Despite this innovation, their approach relies on data generated by the Cooja simulator, which may fail to capture the full complexity and variability of real-world environments. Similarly, Almeida et al. [28] replicate network conditions by using network traces in simulators and regression models to effectively capture the dynamic nature of real-world environments.

III. Radio Link Characterization and Limits of Simulation Models

The objective of this section is to present a concrete example of a wireless network and to thoroughly characterize the various wireless links within it. In particular, we highlight a very large heterogeneity of behaviors: simple simulation models will provide a low accuracy in these conditions.

A. Experimental Setup

We conduct our experiments on the large-scale FIT IoT-Lab testbed [17], deploying a network of 9 nodes at the Grenoble site, just like in [29] (see Figure 1). Each node features an M3-board microcontroller running firmware based on the Contiki Operating System [30]. The IoT devices communicate over an IEEE 802.15.4 network using the 6LoWPAN protocol, with

Fig. 1: Network deployment topology.

Parameter	Value
Number of nodes	11
Traffic data rate	1 packet per second
Traffic direction	Broadcast
Packet size	30 bytes
Operating system	Contiki-NG
Communication protocol	802.15.4
MAC Protocol	CSMA/CA

TABLE I: Considered parameters for our experimental setup.

each node broadcasting a 30-byte packet to all other nodes every second. The IEEE 802.15.4 protocol, commonly used in IoT applications, implements the CSMA/CA mechanism at the MAC layer. The parameters of our setup are detailed in Table I. It is important to note that our approach is designed to be generic, without relying on specific assumptions about the MAC layer, communication protocol, or deployment characteristics.

The network operated for 24 hours, during which time we collected time series data and statistics via a serial output connected to the FIT IoT-Lab monitoring infrastructure, focusing on transmission and reception times. Our primary metric of interest is the Packet Delivery Ratio (PDR), defined as the ratio of successfully delivered packets to the total number of packets sent. It is important to note that our analysis treats links as asymmetrical, meaning the link from node A to node B is considered distinct from the link from node B to node A.

Each link between nodes is represented as a binary sequence, where a received packet is indicated by a 1 and a lost packet by a 0. Then, we aggregated these sequences to compute the number of successfully received packets over specific time intervals, set to $T = 50$ seconds for this study. This process generates time series data for each link, showing the count of correctly received packets in each 50-second interval.

B. Links Heterogeneity

To capture the diversity of links and gain insights into their distribution across the network, we categorized them into four classes according to the mean PDR: (i) Bad links, which exhibit a $PDR \leq 20\%$, (ii) average links, with $20\% < PDR \leq 60\%$, (iii) good links with $60\% < PDR \leq 75\%$ and (iv) excellent links ($75\% < PDR\%$). Figure 2 presents the mean time series for each of the three clusters, along with the minimum and maximum values observed within each cluster. This clearly highlights the heterogeneity within the data and emphasizes the distinct characteristics of each cluster.

C. Are Simulations Accurate?

To highlight the limitations of using simulators to replicate the behavior and performance of real-world networks, we recreated the exact same network setup in the Cooja simulator [31]. Cooja, a discrete-event simulator, is well-regarded for its flexibility as it allows real firmware to be run, effectively mimicking the behavior of actual wireless devices. This makes it a natural candidate for implementing a Digital Twin (DT).

In our simulation, we replicated the real testbed setup by using the same number of devices, arranged identically, running identical firmware and traffic applications. Cooja models wireless transmissions based on two user-defined parameters: P_{Tx} and P_{Rx}. These parameters are used to calculate the probability of a packet being successfully transmitted and received, respectively, provided the distance between the sender and receiver does not exceed a predefined threshold.

When a node A transmits a packet, P_{Tx} determines whether the transmission attempt succeeds. If successful, P_{Rx} then determines whether the packet is successfully received by node B. While this simplified model captures basic wireless behaviors, it falls short in replicating the nuanced and dynamic characteristics of real-world wireless networks, as explored in this study.

With 9 nodes and asymmetric links, we analyze 72 distinct links, tracking the evolution of correctly received packets for each. Links with no communication (always zero) are discarded.

To compare the measurements obtained from the real experimental platform with those generated by the simulator, and to accurately model each link individually, we evaluate the experiments using a personalized calibration using the average PDR: this approach calibrates each link individually using the specific average PDR measured for that link on the real platform. For each link $(a \rightarrow b)$, the transmission probability is set as $P_{Tx}^{(a-b)} = \text{Average-PDR}^{(a-b)}$. This simulates a scenario where each link is calibrated based on its unique real-world performance.

Figure 3 compares the actual PDR measurements from the real platform with those obtained from the simulator under the personalized calibration approach. We present a representative collection of wireless links, selected in different clusters. Key observations from the figure include:

- Real-world measurements show highly diverse behaviors, including sudden drops in PDR and consistently low PDR values throughout the deployment.
- Even calibrating each link with its specific average PDR is insufficient, as PDR evolves dynamically over time, influenced by temporal interferences and other environmental factors.

These findings underscore the limitations of simulators like Cooja in accurately replicating the complex and dynamic behaviors of real-world wireless networks. They emphasize the need for adaptive and more sophisticated modeling approaches to better capture the nuances of such environments.

IV. Can we provide link quality predictions?

A DT mirrors the wireless network. Thus, it must replay the network topology and characteristics to compute metrics.

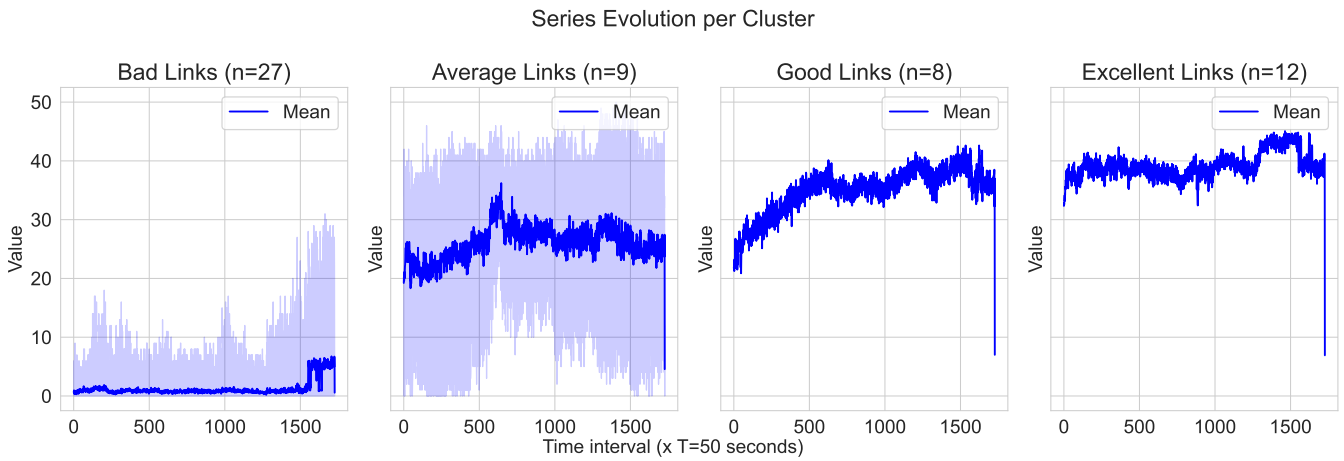

Fig. 2: Cluster time series: each cluster is characterized by the mean of its constituent time series, with the range from the minimum to the maximum values providing a visual boundary around the mean.

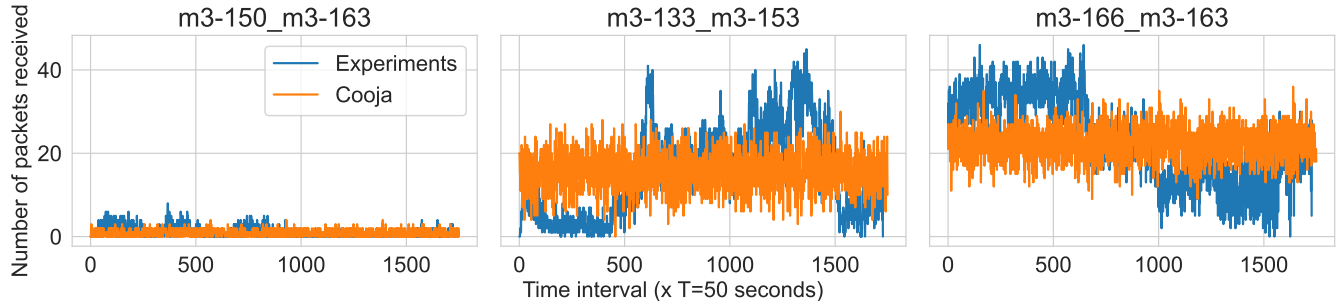

Fig. 3: Wireless links time series for three links from different clusters, where i_j represents the link from node i to node j, observed in real experiments and in the personalized simulator calibration.

In particular, the DT may help to verify that the system operates in *normal* conditions. Modeling the wireless channel represents the most challenging issue in wireless networks since radio links are known to be bursty.

As observed earlier, wireless links within a network exhibit diverse behaviors. While some links maintain consistently high performance with minimal variation, others demonstrate bursty and unpredictable behavior. Consequently, using a single model to replicate all wireless links often yields suboptimal results. Additionally, applying computationally intensive models to links with straightforward, predictable behavior wastes resources.

To tackle these challenges, we propose modeling the DT as a collection of DTs, where each radio link is paired with its own dedicated digital twin (see Figure 4).

Each model is self-calibrated based on real-time measurements from its corresponding link. This modular decomposition offers several key advantages:

- Dynamic adaptability: models for individual links can be updated or replaced on the fly based on real-time measurements, ensuring an accurate representation of their behavior.
- Improved troubleshooting: by isolating underperforming links within their respective digital twins, network issues can be identified and addressed with greater precision.

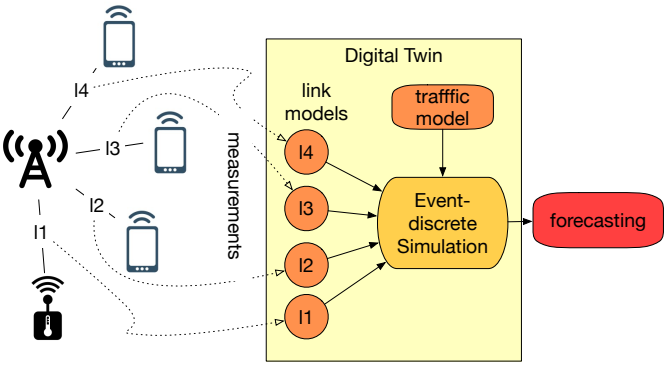

Fig. 4: Architecture of a Digital Twin for wireless networking

- Enhanced scalability: adding a new link and its corresponding digital twin has no impact on the existing network models, enabling seamless scaling.

This approach ensures that the DT remains both efficient and responsive, adapting dynamically to the unique characteristics of each wireless link while optimizing resource usage.

A. *Packet Delivery Ratio as Time Series*

The objective of the DT is to predict, or forecast, the future values of a given performance metric for each

link—specifically the PDR in our case. This task can be framed as a time series forecasting problem, which can be tackled using *e.g.*, regression or ARIMA models.

We analyze time series of PDR measurements, denoted as $[x_1, x_2, \ldots, x_n]$. During each time interval T, we calculate for each link the ratio of packets correctly received by the receiver. In our case, x_i represents the number of correctly received packets during the i^{th} interval ($T = 50s$ by default, as illustrated in Table II). We assume that the characteristics of a link are stationary during a time interval T, and that we collect enough measurements to reflect the actual PDR of the link. A low T value means a higher reactivity, while a high T value reflects a more accurate estimator.

To apply regression models, we consider data using a sliding window technique. The digital twin mirrors the actual performance of the corresponding link: at any instant, it uses the last w values to make its predictions. By considering sequences in the models, we capture temporal patterns such as trends and seasonality, enhancing predictive accuracy for future PDR values. By transforming the time series problem into a supervised learning task, we define the regression model as follows:

$$x_i = f(x_{i-w}, \ldots, x_{i-1})$$

where f represents the predictive model.

B. Model training

We can apply regression models for forecasting at any instant to make predictions. These models are useful for their simplicity and efficiency in predicting outcomes based on input features. In the context of time series forecasting, these models are designed to predict a target variable x_{w+1} using one or more independent variables from previous time points $(x_1, x_2, \ldots, x_w)$. In our study, we consider the following models:

- **Support Vector Regression**: SVR aims to find a hyperplane in a high-dimensional feature space that has the maximum margin from the training data points [32].
- **Gradient Boosting**: it builds a set of prediction models in a sequential manner. It combines these models to create a more through the reduction from each model of the errors made by the previous models [33].
- **Decision Trees**: a Decision Tree model splits the data into subsets based on the value of input features, creating a tree-like structure where each node represents a feature, each branch represents a decision rule, and each leaf node represents an outcome [34].
- **AdaBoost**: AdaBoost (Adaptive Boosting) works by combining multiple weak learners, typically decision trees to form a strong classifier [35].
- **Extra Trees**: it builds multiple decision trees using random subsets of the training data and features. It further randomizes the splitting process by considering random thresholds for each feature instead of searching for the best split point [36].

During the training phase, we aim to identify the best regression model and the optimal window size for each link. To achieve this, we allocate 33% of the data for training, representing the network deployment phase, to construct the DT models. Then, we select the models giving the best predictions on that same training data, along with its best window size w. The choice of window size significantly impacts model performance: a small w may fail to capture long-term dependencies, leading to underfitting, while a large w can introduce unnecessary complexity and redundancy, increasing the risk of overfitting.

C. Continuous Predictions

We describe here the prediction process during the testing phase using a given regression model. The objective is to predict the evolution of the PDR for a given number of intervals. More precisely, we predict the PDR values for the next p time intervals from the last observations, as follows:

At the interval i, we first predict the next value with the last w values. Then, we proceed iteratively: we use the predicted value to make the next prediction. We keep a fixed sequence with w elements and consequently remove the first value of the previous sequence. Thus, after predicting $\hat{x}(i+1)$, the next prediction is made using the inputs $(x_{i-w+1}, \ldots, x_i, \hat{x}(i+1))$, treating $\hat{x}(i+1)$ as an observed value. This process is repeated p times to generate a sequence of p predictions.

Let us call the set of predictions according to the different prediction steps (up to p) at the i-th interval $\hat{X}(i)$. The matrix $\hat{X}(i)$ represents the predicted values at different prediction steps. At each instant i, the model predicts up to p future values, forming a diagonal structure. The prediction matrix is structured as follows:

$$\hat{X}(i) = \begin{bmatrix} \hat{x}_{1,1} & \hat{x}_{1,2} & \cdots & \hat{x}_{1,p} & \cdots & \hat{x}_{1,i} \\ & \hat{x}_{2,2} & \cdots & \hat{x}_{2,p} & \cdots & \hat{x}_{2,i} \\ & & \ddots & \vdots & & \vdots \\ & & & \hat{x}_{p,p} & \cdots & \hat{x}_{p,i} \end{bmatrix}$$

where the elements in the matrix correspond to: (i) **rows:** the prediction step (how far ahead the model predicts) and (ii) **columns:** the time interval i. At each time interval i, the model takes the last w values from the real sequence (or previously predicted values) and generates a sequence of predictions recursively.

Considering that the last observed value is x_i, this process can be expressed as:

$$\hat{x}_{m,n} = f([v_{m-w,n-w}, \ldots, x_{m-1,n-1}]); \tag{1}$$

where $m \leq p, n \in \mathbb{N}^+$, and:

$$v_{k,j} = \begin{cases} x_k, & \text{if } k \leq i \\ \hat{x}_{k,j}, & \text{otherwise} \end{cases} \tag{2}$$

Figure 5 illustrates this process, showing how the $\hat{X(i)}$ matrix is constructed, with an example where$w = 3$ and $p = 4$. In practice, it is preferable to have $p \leq w$ to prevent the model from predicting more values than the historical data it relies on, thereby reducing the risk of divergence. However, this example is presented solely for illustrative purposes.

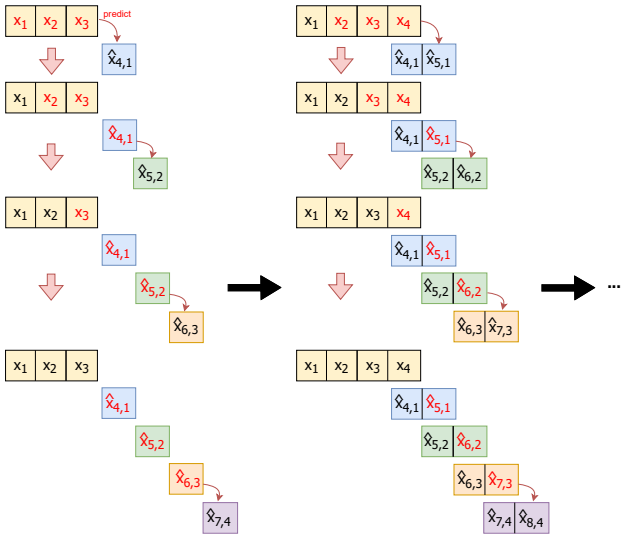

Fig. 5: Prediction matrix construction. We consider the example where $w = 3$ and $p = 4$. At each time step, the model utilizes the most recent observations along with previously predicted values (highlighted in red) to generate the next prediction. This process forms a sliding window that progressively shifts from real observations to the diagonal of the prediction matrix. Once a new observation becomes available, the process repeats, constructing a new diagonal, and continues iteratively.

As new data points arrive from the network, the model is updated dynamically—when a new value x_{i+1} is received at the $i+1$-th interval, the model is retrained using the most recent w values $[x_{i-w}, \ldots, x_i]$ as input and x_{i+1} as the target. This process continues iteratively throughout the deployment, ensuring that the model adapts to evolving network conditions.

D. Prediction strategies

We consider here the two following variants for the predictions during the deployment:

1) Variant 1: Fixed model: We consider that each link has a DT executing a specific **fixed** model. Our hypothesis is that the model achieving the best performance during training should also deliver satisfactory performance during deployment. Note that this assumes a relatively stable link quality.

This approach is described in Algorithm 1, and works as follows:

1) We store the new observation in a time series (line 3);
2) We use the new observation to refit the model: we can compute the error between this new observation and the corresponding predictions we made in the past (line 5);
3) We can also trigger a prediction: we recursively use our model to make p predictions. All the predictions are stored in $\hat{x}_{i,k}$ (lines 6-11)

Figure 6 illustrates the architecture of this approach, where each link is represented by a single model used for predictions during deployment.

Algorithm 1: Fixed Model

Input: w: Sliding window size
Input: p: Prediction step
Input: `model`
Output: At any instant, the next predictions for the PDR values $\hat{x}(*)$

1 $i \leftarrow 0$
2 **while** *a new observation* x_i *is measured* **do**
  /* Store the new observation */
3 $x \leftarrow x \cup \{x_i\}$
  /* Refit the model comparing prediction and observation */
4 **if** $i \geq w + 1$ **then**
5 `model.`$fit[(x_{i-w}, \ldots, x_{i-1}), x_i]$
  /* Predict the next p values iteratively */
6 **if** $i \geq w$ **then**
7 $x_{input} \leftarrow (x_{i-w}, \ldots, x_i)$
8 **for** $k \in [1, p]$ **do**
9 $\hat{x}_{i,k} \leftarrow$ `model.predict`(x_{input})
  /* Remove the first value of the sequence and insert the predicted value */
10 $x_{input} \leftarrow x_{input} \setminus \{\text{first element}\}$
11 $x_{input} = x_{input} \cup \{\hat{x}_{i,k}\}$
12 $i \leftarrow i + 1$

Output: $\hat{X}(i)$

2) Variant 2: Adaptive Model: The previous approach presents a limited accuracy for bursty links: the best model is selected and fixed for a specific link.

We propose here a self-adaptive approach for model selection, where the best-performing model is chosen dynamically at each $(R = 1)$ iterations. Thus, we can here change the parameters of the model and the model itself. For this purpose, we execute in parallel all the different models. Then, we select continuously, at each time interval, the best model, according to the last predictions.

More specifically, at each interval, we compute the prediction error of all models over the last q predictions. This enables us to retain only the model that has demonstrated the best recent performance. The underlying hypothesis is that the model performing best up to a given interval should continue to yield satisfactory results in the subsequent predictions.

To calculate the error of a model up to the i-th interval, for each step j from 1 to p, the predicted values $[x_{i-q,j}, \ldots, x_{i,j}]$ are compared against the actual values to compute step-wise errors for the last q intervals, where i is the current interval index. The overall prediction error up to the i-th interval is then calculated as the sum of squared mean squared errors (MSE) over the last q intervals for the p different steps:

$$error^{model}\ (\mathrm{i}) = \sum_{j=1}^{p} \left(\frac{1}{q} \sum_{k=i-q}^{i} \left(x_k - \hat{x}_{k,j}^{model} \right)^2 \right)^2 \qquad (3)$$

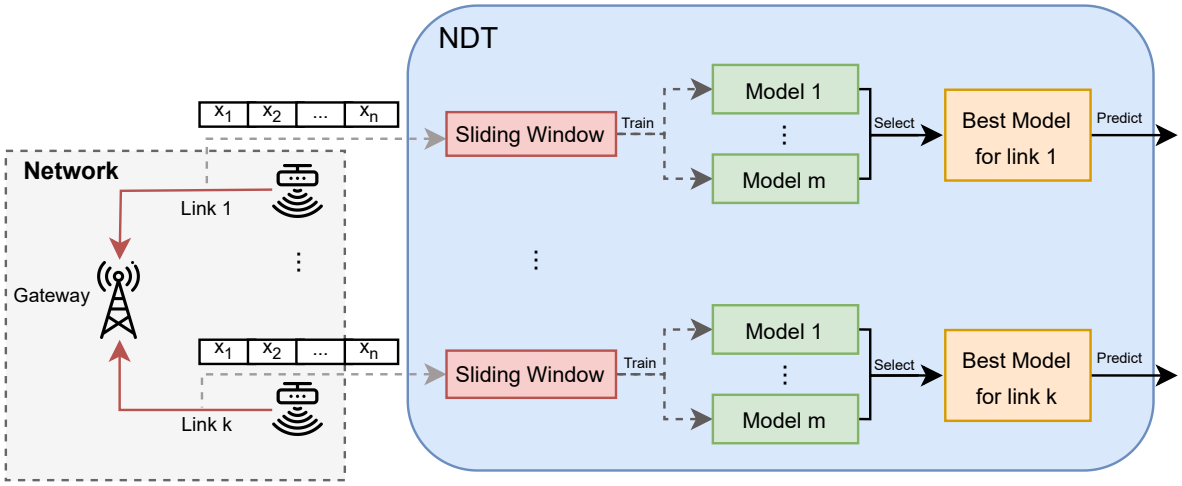

Fig. 6: DT Architecture of our approach. Each link is model separately according to the corresponding measured, and the best model is used for predictions.

The model with the lowest error is selected as the new `BestModel` for the next R intervals:

$$\texttt{BestModel} = \underset{m \in models}{argmin}\ (error^{m}) \tag{4}$$

As a result, predictions over the network's lifecycle may come from different models, ensuring that the system does not rely on a single model that may become inefficient over time.

Algorithm 2 proceeds in detail as follows:

1) The first part of the process is executed, once per model and per link to make the predictions, as previously (lines 7-8);
2) We compute the average error achieved with each model for the last intervals (lines 9-10);
3) We select for the next predictions the best model, *i.e.*, those minimizing the recent error (line 11).

V. Performance Evaluation & Validation

We evaluated our approach using the network described in Subsection III-A to validate it. To simulate the behavior of a real digital twin, we split the collected PDR measurements into two datasets: 33% (approximately eight hours of deployment) for model training and the remaining 66% (roughly sixteen hours) for testing. The testing dataset represents the projected future data that a real network would transmit to its digital twin.

For each link, we recorded the number of correctly received packets and generated time series representing the count of successfully received packets within successive $T = 50$-second intervals. To assess model performance, we used the Mean Absolute Error (MAE), which quantifies the deviation between predicted and actual values during testing. The overall MAE for each model was determined by averaging the MAE values across all network links.

To sharpen our analysis, we varied the prediction horizon from 1 to 20 steps, testing each model's ability to forecast packet reception from 50 to 1000 seconds ahead. In this way, we assessed the models' effectiveness in both short- and medium-term forecasting.

A. One step prediction Error

Figure 7 compares the predictions achieved with the adaptive technique with the measurements observed for the considered links in Section III-C, with $p = 1$, which means that the models are used for one prediction only, and are recalibrated at each interval. The results show that adaptive regression captures the intricacies of the experimental data in a very effective way even for different types of links, such as the highly dynamic **m3-133_m3-163** or **m3-166_m3-163** links.

B. Adaptive Model vs Fixed Model

We compare here the two approaches described in section IV-C:

fixed selects the best model at the end of the training phase. It remains fixed for the rest of the time;

adaptive selects the best model at each iteration ($R = 1$), using the most recent q predictions (and their corresponding error).

Notably, both approaches use a separate model for each link, trained solely on observations from that specific link.

We identified in the preliminary section three distinct categories among the network links (cf. Figure 2): (1) excellent wireless links, (2) good performing links, and (3) average performing links. Therefore, we evaluated the behavior of fixed and adaptive approaches separately for the different link categories. We omit to consider the bad quality links due to their constant low PDR, which makes it easy to provide accurate predictions.

Across all classes, the adaptive method consistently outperforms the single-model approach at every prediction step, demonstrating its ability to handle dynamic link variations

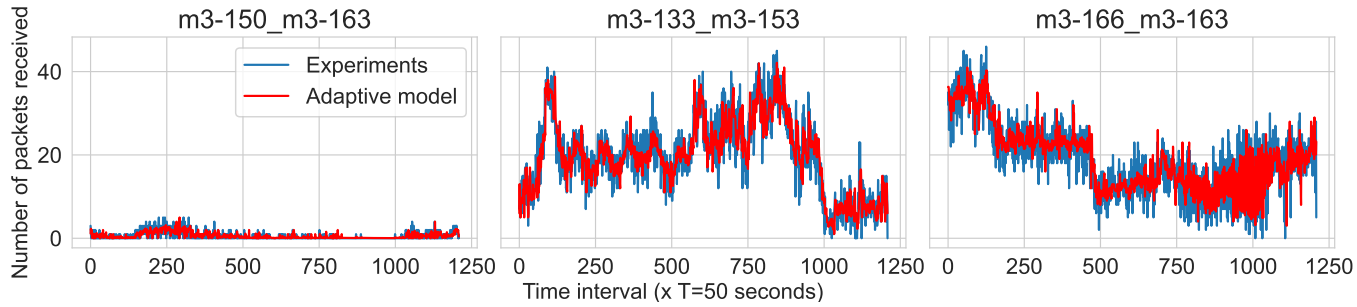

Fig. 7: Wireless links time series for three links from different clusters: Real Measurements vs. Adaptive Regression ($p = 1$)

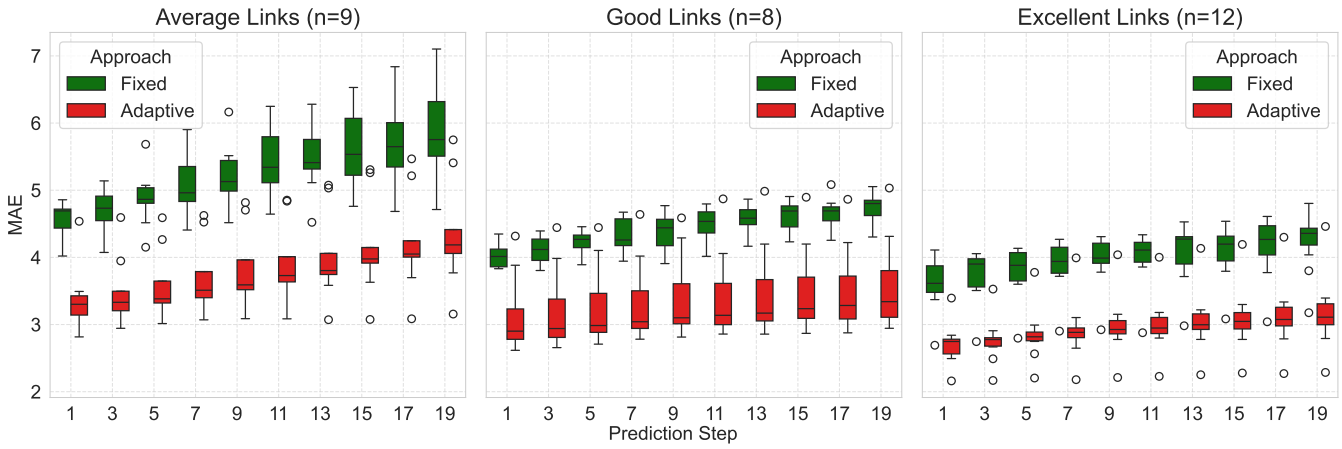

Fig. 8: Prediction error of the adaptive method evolution on the different clusters.

and enhance prediction accuracy. Notably, the prediction error remains remarkably low at step 1, highlighting the method's effectiveness in short-term predictions, with errors increasing gradually as the prediction horizon extends.

For average links, the method performs well in shorter prediction steps, maintaining errors between 3 and 5. This is likely due to minimal variation over short intervals, allowing the model to retain high accuracy. As the prediction step increases, performance declines, but errors remain below 6, demonstrating the method's resilience in sustaining reasonable accuracy over longer horizons.

For good links, both approaches perform even better. The adaptive method remains robust, particularly for short-term predictions, with errors around 3. Even as the prediction step increases, errors stay within a manageable range, not exceeding 4. This suggests the method effectively adapts to dynamic conditions while maintaining reliability.

For excellent links, performance improves further, with errors generally staying below 3, even for higher prediction steps.

Overall, the method's robustness across all link categories highlights its versatility and potential for broader application in diverse network conditions. While the adaptive approach incurs higher computational costs than a single regression model, the simplicity and efficiency of the underlying models ensure that continual learning remains feasible. As a result, resource consumption remains well within acceptable limits for network management.

VI. Conclusion & Perspectives

In this paper, we introduce a self-adaptive approach as the foundation of a digital twin (DT) for wireless networks. Our method redefines wireless network modeling by constructing a collection of digital twins, each specifically tailored to represent an individual radio link. This approach facilitates the dynamic, real-time selection of the best model for each link, using the most recent observations. Our model presents a key adaptability to the diverse and fluctuating conditions of wireless networks. We rigorously validated our approach on a real-world wireless network and achieved remarkable accuracy in replicating the network's PDR performance. This ability positions our digital twin as a transformative tool for enabling proactive and intelligent network reconfiguration. To promote transparency, reproducibility, and further research, we have made our dataset and implementation publicly available to the community[1].

For future work, we plan to extend our approach to include additional metrics, such as latency and the number of consecutively lost packets, which are crucial indicators of network quality for critical applications [37]. Another priority will be enhancing the generalization capabilities of the DT to predict

[1]**Code sharing:** To ensure reproducibility, we provide the source code and dataset at the following link: https://github.com/SamirSim/Wireless-Link-Quality-Prediction

Algorithm 2: Adaptive Model

Input: w: Sliding window size
Input: p: Prediction step
Input: q: History window interval
Input: `models`
Input: `BestModel`
Output: At any instant i, the next predictions for the PDR values from the `BestModel`: $\hat{x}_{(*)}^{\texttt{BestModel}}$

$i \leftarrow 0$
while *a new observation* x_i *is measured* **do**
  /* Store the new observation */
  $x \leftarrow x \cup \{x_i\}$
  /* If we collected enough observations */
  if $i \geq w$ **then**
    for *model in* `models` **do**
      /* Refit the models after observing a new value */
      `model.fit`$((x_{i-w}, \ldots, x_{i-1}), x_i)$
    for *model in* `models` **do**
      /* Compute the predictions for each model as in Algorithm 1 */
      $\hat{X}^{model}(i) \leftarrow$ Algorithm 1(w, p, `model`)
    /* Compute the prediction error for adaptive model selection */
    for *model in* `models` **do**
      $error^{\texttt{model}}(i)$ following Equation 3
    /* Select the best model with the least prediction error */
    `BestModel` following Equation 4
  $i \leftarrow i + 1$

TABLE II: Signification and defaut value for the different parameters.

Parameter	Signifcation	Value
T	Time interval for the computation of number of received packets	50 seconds
w	Sliding window interval	$w \in \{3, 5, 10, 15, 20\}$
p	Prediction step	$p = 20$ intervals
q	History window interval	$q = 5$ intervals
R	Refit frequency	$R = 1$ interval

the performance of unseen links, considering environmental factors like the distance between the sender and receiver or the specific deployment area. These advancements will contribute to a comprehensive DT that not only replicates the wireless network with high fidelity but also supports reliable *what-if* scenario analyses.

REFERENCES

[1] Mehrad Vaezi, Kiana Noroozi, Terence D. Todd, Dongmei Zhao, George Karakostas, Huaqing Wu, and Xuemin Shen. Digital Twins From a Networking Perspective. *IEEE Internet of Things Journal*, 9(23):23525–23544, December 2022.

[2] Kannan Srinivasan, Maria A. Kazandjieva, Saatvik Agarwal, and Philip Levis. The β-factor: measuring wireless link burstiness. In *Conference on Embedded Network Sensor Systems (Sensys)*, pages 29–42. ACM, 2008.

[3] Paul Almasan, Miquel Ferriol-Galmes, Jordi Paillisse, Jose Suarez-Varela, Diego Perino, Diego Lopez, Antonio Agustin Pastor Perales, Paul Harvey, Laurent Ciavaglia, Leon Wong, Vishnu Ram, Shihan Xiao, Xiang Shi, Xiangle Cheng, Albert Cabellos-Aparicio, and Pere Barlet-Ros. Network Digital Twin: Context, Enabling Technologies, and Opportunities. *IEEE Communications Magazine*, 60(11):22–27, November 2022.

[4] Miquel Ferriol-Galmés, Krzysztof Rusek, José Suárez-Varela, Shihan Xiao, Xiang Shi, Xiangle Cheng, Bo Wu, Pere Barlet-Ros, and Albert Cabellos-Aparicio. Routenet-erlang: A graph neural network for network performance evaluation. In *INFOCOM*, pages 2018–2027. IEEE, 2022.

[5] Elif Ak and Berk Canberk. Forecasting quality of service for next-generation data-driven wifi6 campus networks. *IEEE Transactions on Network and Service Management*, 18(4):4744–4755, 2021.

[6] Alberto Salvatore Colletto, Stefano Scanzio, Gabriele Formis, and Gianluca Cena. On the use of artificial neural networks to predict the quality of wi-fi links. *IEEE Access*, 11:120082–120094, 2023.

[7] Duc-Thinh Ngo, Ons Aouedi, Kandaraj Piamrat, Thomas Hassan, and Philippe Raipin-Parvédy. Empowering digital twin for future networks with graph neural networks: Overview, enabling technologies, challenges, and opportunities. *Future internet*, 15(12):377, 2023.

[8] Kosmas Kritsis, Georgios Z. Papadopoulos, Antoine Gallais, Periklis Chatzimisios, and Fabrice Theoleyre. A tutorial on performance evaluation and validation methodology for low-power and lossy networks. *IEEE Communications Surveys & Tutorials*, 20(3):1799–1825, 2018.

[9] Elyes Ben Hamida, Guillaume Chelius, and Jean Marie Gorce. Impact of the physical layer modeling on the accuracy and scalability of wireless network simulation. *SIMULATION*, 85(9):574–588, 2009.

[10] Ramoni Adeogun. Calibration of stochastic radio propagation models using machine learning. *IEEE Antennas and Wireless Propagation Letters*, 18(12):2538–2542, 2019.

[11] Davide Villa, Miead Tehrani-Moayyed, Clifton Paul Robinson, Leonardo Bonati, Pedram Johari, Michele Polese, and Tommaso Melodia. Colosseum as a Digital Twin: Bridging Real-World Experimentation and Wireless Network Emulation. *IEEE Transactions on Mobile Computing*, pages 1–17, 2024.

[12] Hamed Ahmadi, Avishek Nag, Zaheer Khar, Kamran Sayrafian, and Susanto Rahardja. Networked Twins and Twins of Networks: An Overview on the Relationship Between Digital Twins and 6G. *IEEE Communications Standards Magazine*, 5(4):154–160, December 2021.

[13] Latif U. Khan, Walid Saad, Dusit Niyato, Zhu Han, and Choong Seon Hong. Digital-Twin-Enabled 6G: Vision, Architectural Trends, and Future Directions. *IEEE Communications Magazine*, 60(1):74–80, January 2022.

[14] Samir Si-Mohammed, Anthony Bardou, Thomas Begin, Isabelle Guérin Lassous, and Pascale Vicat-Blanc. Ns+ ndt: Smart integration of network simulation in network digital twin, application to iot networks. *Future Generation Computer Systems*, 157:124–144, 2024.

[15] Xiaofeng Wang, Taiqian Shen, Yi Zhang, and Xinyu Chen. An efficient topology emulation technology for the space-air-ground integrated network. In *INFOCOM Workshops*. IEEE, 2023.

[16] Nina Santi, Rémy Grünblatt, Brandon Foubert, Aroosa Hameed, John Violos, Aris Leivadeas, and Nathalie Mitton. Automated and reproducible application traces generation for iot applications. In *Proceedings of the 17th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, pages 17–24, 2021.

[17] Cedric Adjih, Emmanuel Baccelli, Eric Fleury, Gaetan Harter, Nathalie Mitton, Thomas Noel, Roger Pissard-Gibollet, Frederic Saint-Marcel, Guillaume Schreiner, Julien Vandaele, et al. Fit iot-lab: A large scale open experimental iot testbed. In *IEEE WF-IoT*, pages 459–464. IEEE, 2015.

[18] Keoma Brun-Laguna, Pascale Minet, Thomas Watteyne, and Pedro Henrique Gomes. Moving beyond testbeds? lessons (we) learned about connectivity. *IEEE Pervasive Computing*, 17(4):15–27, 2019.

[19] Lidia Pocero Fraile, Stelios Tsampas, Georgios Mylonas, and Dimitrios Amaxilatis. A comparative study of lora and ieee 802.15. 4-based iot

deployments inside school buildings. *IEEE Access*, 8:160957–160981, 2020.
[20] Samir Si-Mohammed and Fabrice Théoleyre. *Data-Driven Prediction Models for Wireless Network Configuration*. In *AINA 2025 - 2025 International Conference on Advanced Information Networking and Applications (to appear)*, 2025 :.
[21] Fidel Alejandro Rodríguez-Corbo, Leyre Azpilicueta, Mikel Celaya-Echarri, Ana Vazquez Alejos, and Francisco Falcone. Propagation models in vehicular communications. *IEEE Access*, 9:15902–15913, 2021.
[22] Nouha Baccour, Anis Koubâa, Luca Mottola, Marco Antonio Zúñiga, Habib Youssef, Carlo Alberto Boano, and Mário Alves. Radio link quality estimation in wireless sensor networks: A survey. *ACM Trans. Sen. Netw.*, 8(4), sep 2012.
[23] CJ. Bernardos et al. Reliable and available wireless (raw) use cases. IETF RFC 9450, https://datatracker.ietf.org/doc/rfc9450/, 2024.
[24] Marco Zúñiga Zamalloa and Bhaskar Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Trans. Sen. Netw.*, 3(2):1–34, jun 2007.
[25] Gregor Cerar, Halil Yetgin, Mihael Mohorčič, and Carolina Fortuna. On designing a machine learning based wireless link quality classifier. In *International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1–7, 2020.
[26] Tao Liu and Alberto E. Cerpa. Data-driven link quality prediction using link features. *ACM Trans. Sen. Netw.*, 10(2), jan 2014.
[27] Hanane Benadji, Lynda Zitoune, and Véronique Vèque. Predictive modeling of loss ratio for congestion control in iot networks using deep learning. In *GLOBECOM*, pages 2814–2819. IEEE, 2023.
[28] Eduardo Nuno Almeida, Mohammed Rushad, Sumanth Reddy Kota, Akshat Nambiar, Hardik L Harti, Chinmay Gupta, Danish Waseem, Gonçalo Santos, Helder Fontes, Rui Campos, et al. Machine learning based propagation loss module for enabling digital twins of wireless networks in ns-3. In *Workshop on ns-3*, pages 17–24, 2022.
[29] Samir Si-Mohammed and Fabrice Theoleyre. Data-driven prediction models for wireless network configuration. In *International Conference on Advanced Information Networking and Applications (AINA)*, 2025.
[30] Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. Contiki-a lightweight and flexible operating system for tiny networked sensors. In *29th annual IEEE international conference on local computer networks*, pages 455–462. IEEE, 2004.
[31] Fredrik Osterlind, Adam Dunkels, Joakim Eriksson, Niclas Finne, and Thiemo Voigt. Cross-level sensor network simulation with cooja. In *conference on local computer networks (LCN)*, pages 641–648. IEEE, 2006.
[32] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14:199–222, 2004.
[33] Jerome H Friedman. Greedy function approximation: A gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.
[34] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1:81–106, 1986.
[35] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156. Citeseer, 1996.
[36] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63:3–42, 2006.
[37] Carlos J. Bernardos, Georgios Z. Papadopoulos, Pascal Thubert, and Fabrice Theoleyre. Reliable and Available Wireless (RAW) Use Cases. RFC 9450, August 2023.

Samir Si-Mohammed is a Postdoctoral Researcher at the ICube lab of University of Strasbourg (France). He graduated with a PhD in Computer Science from École Normale Supérieure (ENS) de Lyon (France) in 2023 and as a Computer Science Engineer from the Higher National School of Computer Science (ESI) of Algiers (Algeria) in 2020. He did his final year internship at EURECOM in Sophia-Antipolis (France). He was a visiting scholar at the Electrical and Computer Engineering department of the University of Waterloo (Canada) during Summer 2023. His research interests include IoT, 5G, Wireless Networks, Digital Twins, and Machine Learning

Fabrice Theoleyre is a senior researcher at the CNRS. After having spent 2 years in the Grenoble Informatics Laboratory (France), he is part of the ICube lab (Strasbourg, France) since 2009. He received his Ph.D. in computer science from INSA, Lyon (France) in 2006. He held several visiting positions at the University of Waterloo (Canada), Inje University (South Korea), Jiaotong University (China). His research interests mainly concern distributed algorithms and experimental design for the Internet of Things.