IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 May 2023, accepted 31 May 2023, date of publication 5 June 2023, date of current version 14 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3283208

RESEARCH ARTICLE

ML-RPL: Machine Learning-Based Routing Protocol for Wireless Smart Grid Networks

CARLOS LESTER DUENAS SANTOS[1], (Graduate Student Member, IEEE), AHMAD MOHAMAD MEZHER[1], JUAN PABLO ASTUDILLO LEÓN[2], JULIAN CARDENAS BARRERA[1], EDUARDO CASTILLO GUERRA[1], (Senior Member, IEEE), AND JULIAN MENG[1], (Senior Member, IEEE)

[1]Electrical and Computer Engineering Department, University of New Brunswick, Fredericton, NB E3B 5A3, Canada
[2]School of Mathematical and Computational Sciences, Yachay Tech University, Urcuqui 100119, Ecuador

Corresponding author: Carlos Lester Duenas Santos (cduenas@unb.ca)

This work was supported in part by the Atlantic Canada Opportunities Agency under Project 208437, and in part by the NSERC Collaborating Research and Development under Project CRDPJ537347-18.

ABSTRACT This research explores the potential of Machine Learning (ML) to enhance wireless communication networks, specifically in the context of Wireless Smart Grid Networks (WSGNs). We integrated ML into the well-established Routing Protocol for Low-Power and Lossy Networks (RPL), resulting in an advanced version called ML-RPL. This novel protocol utilizes CatBoost, a Gradient Boosted Decision Trees (GBDT) algorithm, to optimize routing decisions. The ML model, trained on a dataset of routing metrics, predicts the probability of successfully reaching a destination node. Each node in the network uses the model to choose the route with the highest probability of effectively delivering packets. Our performance evaluation, carried out in a realistic scenario and under various traffic loads, reveals that ML-RPL significantly improves the packet delivery ratio and minimizes end-to-end delay, making it a promising solution for more efficient and responsive WSGNs.

INDEX TERMS Machine learning, wireless smart grid networks, neighbourhood area networks (NAN), routing protocol for low-power and lossy networks (RPL).

I. INTRODUCTION

Communication technologies play an important role in the current transformation of the electricity infrastructure. The operation of the new power grids, known as smart grids, heavily relies on communication systems. Incorporating devices with communication capacities into the grid allows the utilities to achieve a high level of control over the grid, making its operation more reliable, efficient, and secure.

From the communication point of view, the smart grid networks can be divided into three segments or subnetworks as it is shown in Fig. 1: Home Area Network (HAN), Neighborhood Area Network (NAN), also referenced as Field Area Network (FAN), and Wide Area Network (WAN). Home Area Networks are deployed inside homes and act as the communication infrastructure to interconnect sensors and smart appliances. Smart meters collect data generated by HANs and transmit it through the NAN to the utility data centers. NANs connect smart meters and other utility equipment such as line sensors, reclosers, bank capacitors, smart switches, battery storage, and EV charge stations with data aggregation points (DAPs), also named collectors. Finally, WANs represent the last segment of the smart grid communication network, serving the critical role of linking the data aggregation points with the utility control centers. The primary function of WANs is to establish a seamless connection between these two components of the smart grid network.

The three subnetworks have their own requirements and are important for the correct functioning of the smart grid network. However, the NAN has attracted the attention of academia and industry since it supports the transmission of a considerable volume of data and distributes necessary control signals between many end devices and utility control centers [1].

The associate editor coordinating the review of this manuscript and approving it for publication was Stefano Scanzio.

NANs usually cover large areas and can interconnect hundreds of thousands of devices, which challenges the reliability of the network. Communication systems deployed in NANs include wired and wireless technologies or a combination of them. However, due to the advantages of wireless technologies, they are becoming more popular for NANs. Two groups of wireless technologies are relevant in this domain, RF mesh technologies, which include 802.15.4 [2] and 802.11s [3], and the second group considered Low Power Wide Area Networking (LPWAN), which includes Sigfox [4], LoRa [5], and NB-IoT [6].

This work has focused on one of the biggest challenges of mesh technologies: the way the packets are routed. Mesh networks heavily rely on the performance of the utilized routing protocol. One of the well-known routing protocols for Wireless Smart Grid Networks (WSGNs) is the Routing Protocol for Low-Power and Lossy Networks (RPL) [7]. However, numerous studies have pointed out that RPL suffers from issues that limit its efficiency and domain of applicability [8].

We present in this article a new routing strategy over RPL based on Machine Learning (ML) predictions, which we have named ML-RPL. In recent years, ML techniques have shown promising results to solve different problems in communication networks, and routing is one of them. In this case, we have selected CatBoost, a gradient-boosting algorithm, to improve the routing decisions in RPL. The predictions made with the algorithm are used to calculate the path cost from the source to the destination. Nodes choose the routes with the highest probability of delivering the packets to the destination.

While our proposed routing strategy uses Catboost for prediction, it is important to note that this approach can be extended to other ML algorithms as well. We chose Catboost for its strong performance in previous studies [9], [10] and its ease of integration into our simulation platform. However, it is not the focus of this work to perform an in-depth comparison of different ML algorithms in our routing strategy.

In order to assess the effectiveness of our proposed approach, we conducted simulations using actual smart meter locations in the city of Montreal, with traffic patterns representative of those found in smart grid applications. Our new ML-based routing mechanism provided notable improvements to both, packet delivery ratio (PDR) and end-to-end delay across a range of traffic loads.

A. CONTRIBUTION AND ORGANIZATION

Based on our findings in this paper, the following contributions can be outlined:

- We propose a novel routing strategy based on ML to improve the routing decisions of RPL. To the best of our knowledge, previous research has not explored the use of ML algorithms to improve RPL's routing decisions.
- We present a method based on simulations to generate a dataset for collecting routing information in wireless smart grid networks.

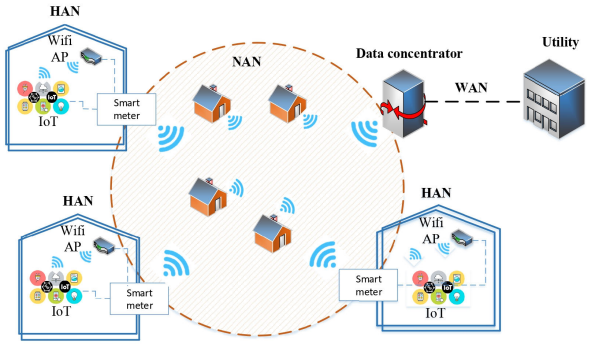


FIGURE 1. Smart grid network [1].

- We conduct simulations using smart meter locations from a real deployment in the city of Montreal to evaluate the performance of our proposed routing strategy. Our simulation scenario was designed to be realistic and reflective of actual network conditions in wireless smart grid networks. To our knowledge, only one previous study [11] has used real-world topology data for evaluation purposes in a simulation environment.

The rest of the paper is organized as follows. Section II discusses some related works. A background on RPL parent selection is provided in Section III. The proposed solution is presented and described in detail in Section IV. The performance of the new routing strategy is evaluated in Section V, and the conclusions and future works are addressed in Section VI.

II. RELATED WORK

This section shows how ML-based solutions have gained interest in solving traditional network issues in wireless networks. We mainly cover solutions for routing algorithms, but other investigations related to congestion control and traffic classification are also highlighted. We also show recent improvements in RPL and the approaches they have followed.

The work in [12] proposes DTMR, a decision tree-based multi-metric routing protocol for vehicular ad hoc networks (VANETs) that aims to make more intelligent forwarding decisions. The authors obtained their targeted dataset from multiple simulations runs over different urban VANET scenarios and evaluated the importance of different routing metrics by applying regularization. Subsequently, only the most relevant routing metrics were incorporated into DTMR for forwarding decisions. However, the authors did not exploit the inherent capability of decision tree-based models to determine feature importance directly, which could potentially have resulted in a different ranking of routing metrics. Additionally, while the evaluation focuses on achieving an optimal protocol design, the proposed solution is not benchmarked against other routing protocols for VANETs, which represents the primary limitation of this work. The same authors present GraTree in [9], another ML-based multimetric routing protocol. For this paper, they employed CatBoost to train the

ML model using the same dataset they obtained in [12]. GraTree is compared to an existing solution not based on ML and to DTMR in terms of packet delivery ratio (PDR), end-to-end delay, and overhead. GraTree performs better than the other routing protocols except for the delay measure. Although the routing solution presented in the paper targets VANETs, the comprehensive description of their ML-based approach, including data collection, processing, learning phase, and integration into the simulator, makes the methodology adaptable to other types of networks.

Authors in [13] address the network congestion control problem in smart grids. They present a new congestion control mechanism based on feed-forward neural networks for smart grids. The purpose of the work is to guarantee the different Quality of Service (QoS) requirements for different smart grid applications. The proposed mechanism requires source nodes to decide whether to transmit new data packets generated by the applications based on the current state of the network. This state is characterized by the value of the channel utilization factor and by the packet buffer occupation. The evaluation of the proposed mechanism shows significant improvements in terms of throughput, transit time, and quality of service differentiation. However, the grid topology considered in this research work does not represent a real smart grid deployment, which would provide a more realistic test for congestion management. Future works must take into account smart meter positions from real deployments.

An Enhanced Tree Routing Based on Reinforcement Learning for Wireless Sensor Networks (WSN) is presented in [14]. The aim of the protocol is to identify the most suitable parent node within a tree topology by utilizing empirical network data gathered via Q-learning. The authors establish a state space, action set, and reward function based on multiple cognitive metrics, and subsequently determine the optimal parent node through an iterative process. Simulation outcomes indicate that the proposed method outperforms existing techniques, such as the linear weighted sum-based parent selection algorithm, in aspects like end-to-end delay, packet delivery ratio, and energy consumption. The analysis does not address the potential routing overhead incurred by the algorithm due to the cycle detection mechanism implemented, which requires each node to send a join request message containing its list of child nodes to the candidate parent node. Furthermore, the paper does not examine the scalability of the proposed method in larger network scenarios, which may present additional challenges and limitations.

In an attempt to improve the RPL protocol, authors in [15] use Adam Deep Neural Network (ADNN). The work addresses the problem of routing overhead, packet losses, and load imbalanced in RPL. They state that the parent selection policy must be changed based on the type of packet required to transmit in order to achieve a better quality of service in the network. ADNN is used to classify the packets considering packet header information such as packet length, time to live, payload length, and payload content. Although the proposed routing algorithm outperforms other solutions in the literature, the experiments conducted for a small network of 50 nodes raise concerns about the algorithm's complexity. The algorithm comprises multiple components, such as a grid-based network construction with various levels of unequal grids, grid head selection, a transmission scheduling mechanism, two different objective functions, and modifications to the trickle timer algorithm, in addition to employing the Adam Deep Neural Network mentioned above.

Another work focused on improving RPL using ML techniques is presented in [16]. In this case, a new parent selection strategy is proposed to choose the best parent when two or more candidates have the same ranking in the RPL destination-oriented routing tree. The core of this technique is to use Random Forest (RF) [17], for feature importance analysis. The parent selection strategy is designed based on the importance of each feature, which are routing metrics, such as expected transmission count, mac losses, channel utilization, and throughput. The routing metric importance is then used to assign weights to a forwarding score function, which aids in determining the optimal candidate parent among all potential options. The primary limitation of this strategy is the static assignment of weights, which prevents the parent selection strategy from being more adaptable to different load conditions. Despite this limitation, the simulation results presented in the paper demonstrate significant improvement in terms of PDR compared to standard RPL implementations across various network sizes.

Aside from the last two investigations mentioned above, to the best of our knowledge, more recent approaches to improve RPL did not apply ML techniques to enhance the routing decisions. For example, in a recent enhancement of RPL named Weighted Random Forward RPL (WRF-RPL) [18], the authors tackle the load balancing problem by combining the energy remaining in the nodes and the number of possible parents that a node has. These two routing metrics are the base of a weighted random selection algorithm used to choose the best next-hop candidate. Even though WRF-RPL improves the network's lifetime and the PDR, the solution ignores other important routing metrics related to link quality. In addition, considering only the number of parents may not accurately reflect the actual load on a node. Some nodes with fewer parents might experience higher traffic or processing demands, while others with more parents might be underutilized. This can lead to an imbalanced load distribution and potentially reduced network performance.

Authors in [19] stand for QWL-RPL. They analyze the RPL protocol under a heterogeneous traffic pattern and propose a new protocol based on the queue and workload-based condition. The queue condition is obtained by counting the number of packets in the queue, and the workload is measured at each node by counting the number of transmitted packets at the MAC layer during fixed periods of time. The first metric would be an indicator of congestion, and the second one of traffic load. Thus, the node chooses as its preferred parent the

one less congested and with less traffic load. The combination of the metrics is simply the sum of them. The most significant improvement observed with QWL-RPL is a routing overhead reduction. After examining the results and the protocol's performance throughout all the experiments, the authors suggest the potential benefits of incorporating ML into the routing protocol. This would allow the protocol to possess self-learning and self-adaptive features, facilitating a shift from a rule-based routing approach to a learning-based one.

The possibility of exploiting the multi-topology routing feature of the RPL standard has also received some attention. The authors in [20] investigate the use of multiple RPL instances in Wireless Sensor Networks (WSNs). However, they utilize the two standard RPL implementations, and no modifications are proposed for them. The implementation based on the hop count metric is used for the instance of periodic and non-critical traffic, and the implementation based on the expected transmission count (ETX) is used for the instance of high-critical data traffic. Another work in the same direction is presented in [21]. This work focuses on the use of multiple RPL instances to ensure the Quality of Service (QoS) provision for different traffic classes. Also, this investigation addresses the single routing metric problem in RPL, and a new parent selection framework based on a multi-attribute decision-making approach is proposed. The results show that the multi-topology routing approach improves the QoS provision in the network. However, the total length of each simulation may not be sufficient to capture the long-term behavior and performance of the proposed routing approach under various network conditions.

In summary, while some works covered in the literature review have shown promising results, there remain challenges such as routing metrics selection, the use of realistic network topologies, scalability, dynamic adaptability, and balancing the trade-offs between performance improvements and added complexity. Building on these insights, our proposed machine learning-based routing protocol addresses these challenges to further enhance the routing performance for WSGNs.

III. RPL PARENT SELECTION BACKGROUND

To provide basic knowledge for understanding our proposed machine learning-based routing protocol for smart grid networks, it is essential to first explore the parent selection process in RPL. As our design builds upon and enhances the RPL protocol, this section aims to provide the necessary background on RPL's parent selection mechanism.

RPL is a distance vector routing protocol that is adapted to a variety of Low-Power and Lossy Networks (LLN). The protocol constructs a multihop routing tree rooted at a single 6LoWPAN Border Route (6LBR) by forming a destination-oriented directed acyclic graph (DODAG) [7]. When a new node joins an RPL network, it selects a parent node (default route) based on the DODAG information that it receives from its neighbors through DAG information object (DIO) messages.

The rules of how each node determines the preferred parent are guided by the Objective Function (OF). The Internet Engineering Task Force (IETF) has standardized two OFs for this purpose: the Objective Function Zero (OF0) defined in RFC6552 [22], and the Minimum Rank with Hysteresis Objective Function (MRHOF) defined in RFC6719 [23]. OF0 utilizes the hop count as a routing metric to identify the optimal parent among candidate neighbors, making it ideal for selecting the nearest node to the DODAG root as the preferred parent.

In the case of MRHOF, the OF has been designed to find the paths with the smallest path cost while preventing excessive network churn. It does so by using two mechanisms. First, it finds the minimum path cost, and second, it switches to the node that offers the shortest path cost if the path cost through this node is less than the current path cost by at least a given threshold. This second mechanism is called "hysteresis" [23]. By default, the OF uses the expected transmission count as a routing metric to calculate the path cost. The ETX is a measure of the quality of a link in terms of reliability. Low values indicate a link is more reliable and vice versa. The path cost from a node to the DODAG root is the sum of the ETX of each link in the path. If multiple candidate parents share the same path cost, other tie-breaking criteria might be used, which is implementation dependent. One of the effective alternatives for tie-breaking is presented in [16].

IV. MACHINE LEARNING-BASED ROUTING DESIGN

In an RPL network, each node recognizes its neighbor nodes by DIO messages received from them. Thus, every time a node receives a DIO message, it must update the candidate parent set and select the preferred parent considering the routing metrics defined for that purpose. In our strategy, we follow the same approach, but instead of combining the metrics directly as many previous investigations in the literature have done, we use the metrics to calculate the probability of reaching the node that sent the DIO. To calculate the probability, the nodes use an ML model that has as inputs features based on certain routing metrics and outputs a probability on whether the DIO-sender node can be successfully reached.

Let's designate the positive event of reaching the DIO-sender as class label $y = 1$, and the $p(y = 1|x)$ is the probability that a particular sample belongs to class 1 given its features x. Then, the path cost of node k to reach the DIO-sender (e.g. node m) could be expressed by the following equation:

$$path\_cost(k, m) = 1 - p(y = 1|x) \tag{1}$$

As we are interested not only in the path cost to reach node m but also in the path cost to the destination (i.e., DODAG root), the total path cost of node k to reach the DODAG root through node m would be the sum of the path cost to reach node m, plus the node m's path cost to reach the root. That is expressed in Eq.(2):

$$C(k, m) = path\_cost(k, m) + C(m, \hat{P_m}) \tag{2}$$

where $\hat{P}_m$ is the preferred parent of node m. It is worth noting that node m must advertise its path cost so that node k can calculate the path cost through node m.

Nodes keep a record of the total path costs through all possible candidate parents. The node that offers the lowest path cost to the destination is then selected as the preferred parent. In general, If $\mathcal{S}$ is the set of n candidate parents of node k, the best alternative parent or preferred parent $\hat{P}_k$ is given by:

$$\hat{P}_k = \underset{n \in S}{\operatorname{argmin}}\{C(k, n)\} \tag{3}$$

As in the MRHOF version of RPL, a node switches to a new parent only if the new minimum calculated total path cost is smaller than the current total path cost through the preferred parent by the *parent_switch_threshold*. The value of *parent_switch_threshold* serves as a hysteresis factor that aims to prevent unnecessary and inefficient parent node changes.

Algorithm 1 describes the entire strategy for parent selection. The algorithm starts having a set of candidate parents, followed by computing the total path cost to the destination via each candidate. Then, the algorithm determines the candidate with the lowest path cost and switches to a new parent if the threshold decision is satisfied.

Algorithm 1 Pseudo-Code of the Parent Selection Strategy

Require: $\mathcal{S}$, a set of n candidate parents of node k. P_k, current_preferred_parent. ζ, *parent_switch_threshold*.
  for each $n \in \mathcal{S}$ **do**
    Calculate total cost to the root through n, $C(k, n)$.
  end for
  $\hat{P}_k = \underset{n \in S}{\operatorname{argmin}}\{C(k, n)\}$
  if $C(k, \hat{P}_k) \leq (C(k, P_k) - \zeta)$ **then**
    $P_k \leftarrow \hat{P}_k$
  end if
return P_k

The parent selection strategy described in Algorithm 1 is the core of our ML-based routing proposal, ML-RPL. Since the path cost calculation in the strategy depends on the predictions made by an ML model, the necessary steps to develop the ML model are presented in Fig. 2. The following subsections describe each step of the process.

A. DATA COLLECTION AND PROCESSING

The first step in Fig. 2 focuses on data collection and the subsequent processing required to obtain a representative dataset for model training and evaluation. In practice, constructing a good dataset can be challenging as it is heavily reliant on the type of data being observed. To address this challenge, it was imperative to collect data using a real smart grid network, as there were no pre-existing datasets available for our proposal. Accordingly, representative simulation scenarios were created, considering a public database that

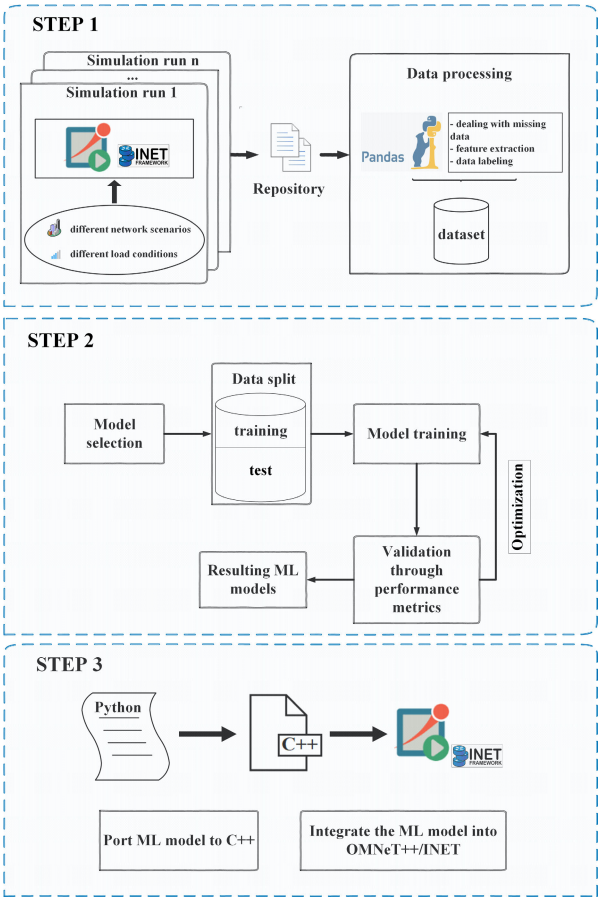


FIGURE 2. Roadmap to build the ML model for ML-RPL.

provides the positions of 335,297 smart meters distributed over an area of approximately 431 square kilometers in the city of Montreal. The large scale of this deployment made it infeasible to simulate the entire scenario using any network simulation tool. Thus, the simulations for data collection were run separately over different areas, as illustrated in Fig. 3. The simulations were carried out using the network simulator OMNeT++ [24].

In each simulation, the smart meters sent packets through random routes to the collector. Our focus was to capture routing metrics at the moment a node was either transmitting or forwarding a packet. This allowed us to collect routing metrics related to the link between sender and receiver and others that give information related to the status of the nodes. In summary, the gathered routing metrics are:

- **Hop count**. Distance to the destination based on the number of hops.
- **Expected Transmission Count (ETX)**. Calculated with respect to the meter that is supposed to receive the packet. The ETX is calculated on a link according to the following expression:

$$ETX = \frac{1}{Df \cdot Dr} \tag{4}$$

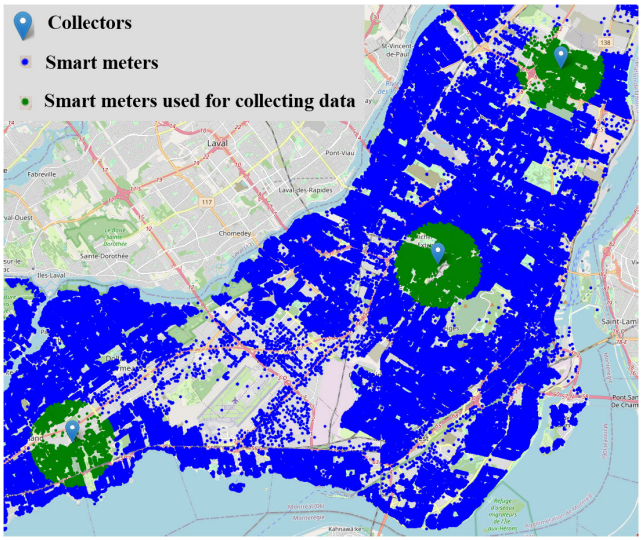


FIGURE 3. Smart meters deployment in the city of montreal.

TABLE 1. Simulation settings for data collection.

Channel	Path loss	$\alpha = 3.6$
	Shadowing	Lognormal, $\sigma = 7.4$
PHY Layer	Standard	802.15.4g
	Frequency band	2.4 GHz
	Transmission rate	115 Kbps
	Transmission power	14 dBm
MAC Layer	Standard	802.15.4g
	Operation mode	Mesh
	ACK	Enable
	Max re-transmission	3
	Backoff procedure	Exponential

Df is the measured probability that a packet is received by the neighbor and *Dr* is the measured probability that the acknowledgment packet is successfully received. The ETX is also smoothed by using an exponential weighted moving average (EWMA) filter, making it robust to sudden link condition changes:

$$ETX_{new} = \alpha \cdot ETX_{current} + (1 - \alpha) \cdot ETX_{prior} \quad (5)$$

the value of α is implementation dependent, and it has been set to 0.8 as in other implementations [25].

- **MAC losses**. The average percentage of frame losses in the MAC Access Control (MAC) layer. These losses are computed as the sum of losses resulting from the maximum number of frames retransmitted, the maximum number of extra backoff times reached, and queue drops.
- **Density**. Number of neighbors within the transmission range of the meter that is supposed to receive the packet.
- **Channel utilization**. Percentage of the time the channel is busy at the receiver side.
- **Throughput**. Frames per second transmitted by the meter that is supposed to receive the packet.
- **Queue utilization (Qu)**. Queue utilization of the meter that is supposed to receive the packet. It is computed by each node as:

$$Qu(k) = \frac{\text{Number of packets in queue of node } k}{\text{Total queue size of node } k} \quad (6)$$

- **Received Signal Strength Indicator (RSSI)**. Average of RSSI values of the packets received from the node that is supposed to receive the packet.

All the routing metrics except for **Density** and the **Hop count** were smoothed using an EWMA filter in a similar way as the ETX. In order to obtain a large number of samples covering a wide range of values of these routing metrics, different simulation conditions were considered. For example, to have different congestion levels, the sending interval of the smart meters varied from 5 minutes up to 1 hour. Also, the number of smart meters in the scenarios was changed from 50 to 300.

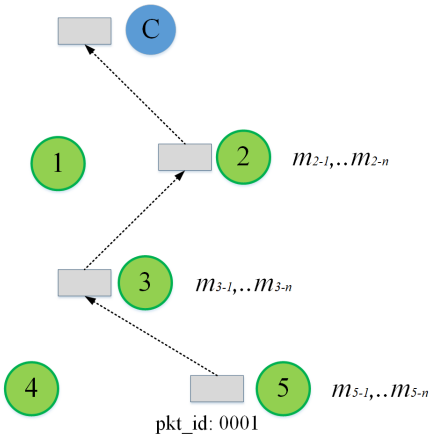
The communication channel was modeled by using the *Log-Normal Path Loss Model* as suggested in [26]. The value for the path-loss exponent α and the shadowing deviation σ are taken from [27], which have been used in previous related works like [28]. Other simulations setting are given in Table 1.

To summarize the data collection, Fig. 4 shows a simplified example of the process. For instance, if node 5 sends a packet to the collector, represented by node *C*, the hop statistics are stored in a repository with the structure shown at the bottom of Fig. 4. The columns of the repository corresponding to packet sending time, source, packet identifier, packet event (packet sent, forwarded, or received at the destination), and the values of the different routing metrics at each hop.

The raw data obtained from the simulations have to be processed and organized in a structured way. To this end, it was used Pandas [29], a library developed in the Python programming language for the purpose of manipulating and analyzing data. Once the data is processed, the dataset used to train the ML model looks like in Fig. 5. Each sample in the dataset represents a unique packet transmitted over the network, recorded hop by hop. The features are the routing metrics recorded at each hop, and the classes represent whether the packet was received at the next hop or not (r = 1, if the packet was received successfully, r = 0 otherwise). A summary of the characteristics of the dataset that will be used for training is shown in Table 2.

B. ML MODEL SELECTION

The second step in our roadmap involves the selection of the ML model that will be used for predicting the best parent for forwarding packets at each hop. We are presented with a binary classification problem where we want the ML algorithm to learn when the packets are received successfully or not at the next hop.



Repository

at	node	pkt_id	action	m_1	m_n
t_1	5	0001	*pkt_sent*	$m_{5\text{-}1}$	$m_{5\text{-}n}$
t_2	3	0001	*pkt_fwd*	$m_{3\text{-}1}$	$m_{3\text{-}n}$
t_3	2	0001	*pkt_fwd*	$m_{2\text{-}1}$	$m_{2\text{-}n}$
t_4	C	0001	*pkt_rcv*	-	-

FIGURE 4. Example of data collection hop by hop and the repository.

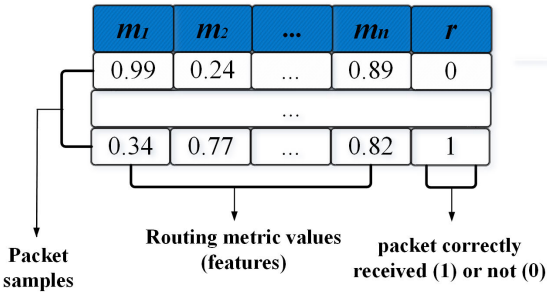


FIGURE 5. Dataset appearance.

TABLE 2. Dataset summary.

Number of samples	67834
Number of features	n = 8 (Hop count, ETX, MAC losses, Density, Channel utilization, Throughput, Queue utilization, RSSI)
Labeled classes	2 (packet received or not, r = [1, 0])

There are many ML algorithms for classification problems. In our case, we have chosen CatBoost, a member of the family of Gradient Boosted Decision Trees (GBDT) algorithms. This algorithm has already been used successfully in networking. In [9], it is compared to other ML algorithms to make forwarding decision tasks, and the results encouraged its use in this work. Catboost can also be exported to C++, which makes it easy to integrate it into the network simulator.

CatBoost is built upon the theory of decision trees and gradient boosting. Gradient boosting ML technique was first described in [30]. The main idea of boosting is to sequentially combine many weak models to create a solid competitive predictive model. When decision trees are used as base learners, a new tree is added at each step of the process. The decision trees are fitted sequentially, so the fitted trees learn from the former trees' errors to minimize the value of the loss function. Adding a new tree to existing ones continues until the selected loss function is no longer minimized or a maximum number of trees is reached. Algorithm 2 describes the boosting process with decision trees as base learners.

Algorithm 2 GBDT Algorithm

Require: $(x_i, y_i)_i^n$ a set of data, where x_i are input values, and y_i, $i \in \{1 \dots n\}$ are the expected output values. $L(y_i, F(x))$, a differentiable Loss Function. Learning rate, α. M, maximum number of decision trees.

Initialize: $F_0(x) = \underset{\gamma}{\text{argmin}} \sum_{i=1}^{n} L(y_i, \gamma)$

for $m = 1$ *to* M **do**

1: Compute $r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)}$ for $i = 1, \dots, n$

2: Fit a regression tree to the r_{im} values and create terminal regions R_{jm}, for $j = 1 \dots J_m$

3: For $j = 1 \dots J_m$, compute $\gamma_{jm} = \underset{\gamma}{\text{argmin}} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)$

4: Update $F_m(x) = F_{m-1}(x) + \alpha \cdot \sum_{j=1}^{J_m} \gamma_m \mathbb{1}(x \in R_{jm})$

end for

Output: $F_M(x)$

Two critical algorithmic introduced in CatBoost are the implementation of ordered boosting, a permutation-driven alternative to the classic algorithm, and an innovative algorithm for processing categorical features. Both techniques were created to fight a prediction shift caused by a special kind of target leakage present in all currently existing implementations of gradient boosting algorithms [31]. Also, CatBoost does not follow similar gradient boosting models in the growing procedure of the decision trees. Instead, it grows oblivious trees, meaning that the same splitting criterion is used across an entire level of the tree. Such trees are balanced, less prone to outfitting, and allow to speed up the executions at the testing time significantly [32].

C. MODEL OPTIMIZATION AND VALIDATION

Prior to initiating the training process, the dataset is partitioned into two subsets, with 80% allocated for training and 20% for testing. The initial phase of the training process involves adjusting the primary hyperparameters of the model. These hyperparameters, such as the learning rate, the number

TABLE 3. Hyperparameter search space.

Parameter	Value
Training dataset	80%
Testing dataset	20%
Cross-validation	5
Iterations	6000
Loss function	"Logloss"
Validation-based	Early stopping
Learning rate	[0.01, 0.05, 0.1]
Tree depth	[6, 8, 10]
L2 regularization	[1, 3, 5]
Tree growing policy	["SymmetricTree", "Depthwise"]
Auto_class_weight	["Balanced", "SqrtBalanced", "None"]

of trees in the ensemble, and the maximum depth of each tree, and others, are predetermined model parameters that are not learned from the data. To do so, we have used a technique named Grid Search, which is implemented in the Python library scikit-learn [33]. The grid search technique involves exploring the optimal parameter values within a specified parameter grid. This means that a range of values is assigned for each hyperparameter, which is then evaluated through the grid search to determine the most suitable combination. Table 3 summarizes the hyperparameters search space.

In order to see how the Catboost model performs after the hyperparameter optimization, we show the Receiver Operating Characteristic (ROC) curve in Fig. 6. The ROC curve is one of the most well-known performance metrics for ML classifiers [34]. The area under the curve (AUC) is a measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes. Fig. 6 shows five curves and their mean because K-fold Cross-Validation has been used. K-fold Cross-Validation is a technique where the dataset is randomly split into k folds without replacement, where $(k-1)$ folds are used for the model training and one fold is used for testing. This procedure is repeated k times, and the ROC curve was recorded at each time. The mean value of ROC shown in Fig. 6 is 0.91, which means that our model achieves good class separation performance.

We are also interested in analyzing the relevance of each feature in the model. By identifying the routing metrics that are most relevant, the proposed routing algorithm can improve efficiency. Catboost has implemented different methods to calculate the feature importance after model training, so we can directly get the individual importance values for each of the input features. The result of this analysis is presented in Fig 7. The values are normalized so that the sum of the importance of all features is equal to 100. A higher value of the importance indicates a larger average change to the prediction value, if this feature is changed.

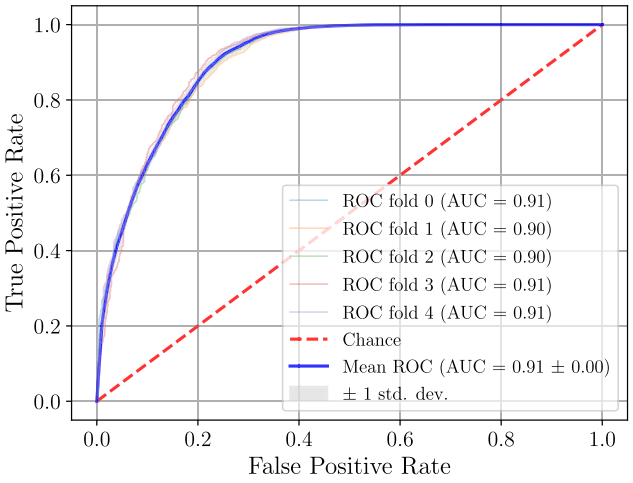


FIGURE 6. Receiver operating characteristic (ROC) curve.

It is seen that the RSSI metric has the most influence on the model output, followed by the ETX and the throughput. Contrarily, the density, and the queue utilization show low significance in the model output. In order to complement the previous analysis, we have applied Recursive Feature Elimination (RFE). RFE is a feature selection method that recursively removes the least important feature until the specified number of features is reached. In Fig 8, we have plotted how the AUC of the model varies versus the number of features selected. The figure shows that utilizing more than six metrics does not result in any improvement in the ROC metric. As a consequence, the ML model does not consider either the density or the queue utilization when making routing decisions.

The developed ML model has to be integrated into the network simulator, step 3 in Fig. 2. Since the model was initially developed using Python, we converted the Python-based model to its equivalent C++ implementation. The C++ model is incorporated into the OMNeT++ simulator by linking the implementation with the rest of the modules and components of the simulator. The next section is dedicated to testing the model as part of the routing protocol in the network simulator.

V. PERFORMANCE EVALUATION

We present the performance evaluation of our proposed ML-RPL in this section. Our approach is compared to one of the standard RPL implementations (MRHOF) and also to RPL+, the proposal presented in [16]. Three different experiments varying the network load were conducted to compare the routing algorithms. Next are the details of the simulation settings, and later the results of each experiment are presented.

A. SIMULATION SETTINGS

To compare the routing algorithms, we utilized the OMNeT++ simulator in a realistic scenario extracted from the city of Montreal. The scenario consisted of a set of

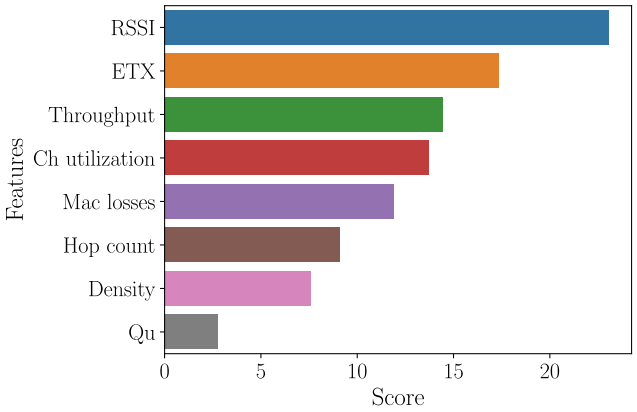


FIGURE 7. Feature importance analysis.

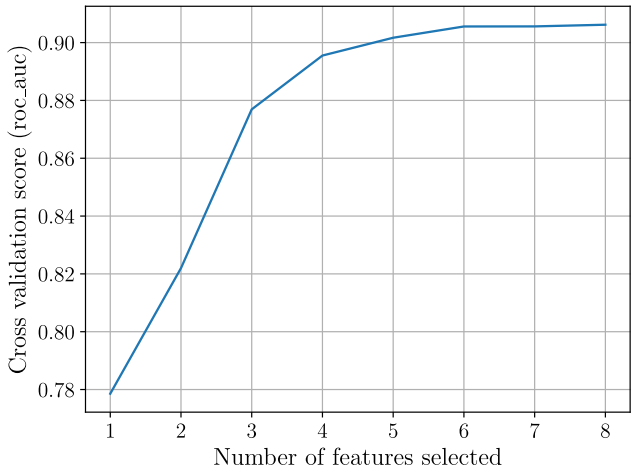


FIGURE 8. ROC_AUC score versus number of features.

TABLE 4. Simulation parameters.

Network simulator	OMNeT++ & INET Framework v4.0
Simulation runs	10
Simulation time	5400s
Nodes	Smart meters: 200 Collector: 1
Packet size & sending interval	Application dependent, according to Table 5.

200 smart meters and one collector, as depicted in Fig. 9. The channel, physical layer, and MAC layer have the same characteristics as was described in the data collection process, Table 1. Each simulation executed corresponds to 1.5 hours of network operation, and the results presented are the average of 10 simulations per experiment with a confidence interval of 95%. The simulation settings can be found in Table 4.

Regarding network traffic, typical smart grid applications such as Meter reading (MR), Alarm events (AE), and Power Quality (PQ) have been considered. MR is the most basic

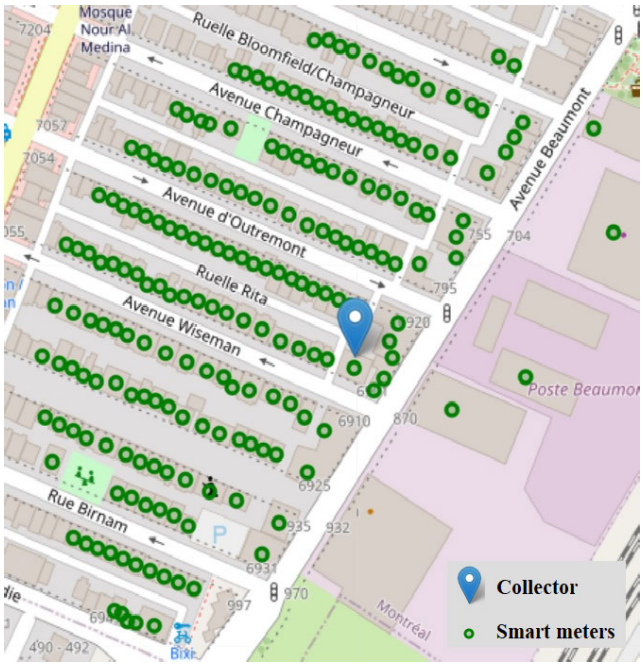


FIGURE 9. Simulation scenario.

smart grid application. It refers to the usage information that smart meters collect and must report periodically to utilities. AE is the second application taken into consideration. Alarms can happen at any time and are sent randomly during the simulation time by a percentage of smart meters. AE can include events such as measurement failure, system restart, system memory full, configuration errors, etc. The other application considered has been PQ. Examples of Power Quality events include but are not limited to leading/lagging power, voltage fluctuations, imbalance in energy flow, harmonics, and voltage sags and swells. Table 5 presents how these applications have been configured in each experiment.

As shown in Table 5, experiment 1 has the lowest traffic load. For experiment 2, the three applications are transmitted, but only 25% of the smart meters transmit AE and PQ. Experiment 3 has the highest traffic load, where the MR traffic is doubled, and 50% of the smart meters generate AE and PQ traffic.

B. SIMULATION RESULTS

Fig. 10 shows the PDR measured at the collector for the first experiment. Recall that the PDR expresses the ratio between the number of successfully received packets at the destination and the total number of transmitted packets. In this experiment, the smart meters transmit only MR packets. It can be seen that ML-RPL performs better than MRHOF and RPL+ by 7% and 5%, respectively. Overall, ML-RPL is able to achieve 94% of PDR on average.

In terms of end-to-end delay, the comparison among the three routing protocols is shown in Fig 11. The highest delay corresponds to MRHOF since it is based on the ETX metric which measures the most reliable paths not necessarily the

TABLE 5. NAN applications transmitted over the SG per experiment.

Exp	Applications	Sending period	Payload (bytes)	Percentage of meters
1	Meter reading (MR)	Every 1 hour	400	100 %
2	Meter reading (MR)	Every 1 hour	400	100 %
	Alarm events (AE)	Every 1 hour	278	25 %
	Power Quality (PQ)	Every 1 hour	278	25 %
3	Meter reading (MR)	Every 30 min	400	100 %
	Alarm events (AE)	Every 1 hour	278	50 %
	Power Quality (PQ)	Every 1 hour	278	50 %

shortest ones. The average delay of RPL+ is almost half of MRHOF's value. It is worth noting that RPL+ takes into account the hop count metrics to build the routes from nodes to the root. Later other metrics are applied to decide among candidates with the same hop distance, so the paths with minimum distance to the destination are always chosen. This is why better results with respect to MRHOF were expected. ML-RPL was found to be slightly better than RPL+ with an average difference in delay of only 60 ms. However, it is worth noting that ML-RPL exhibits greater deviation from the mean delay value, with some individual simulation runs experiencing delays exceeding 600 ms. ML-RPL is more of an intermediate solution between MRHOF and RPL+. It considers the hop count metric alongside other metrics to predict the optimal path. As a result, the weight assigned to other metrics may sometimes result in longer paths having a better probability of successful delivery. Nevertheless, ML-RLP guarantees successful delivery to the destination.

In the second experiment, we added AE and PQ traffic. Fig 12 depicts the PDR for each routing variant. We can observe that ML-RPL achieves the best PDR for the 3 traffic categories. However, for MR, the PDR fell 3% with respect the experiment 1 when there was no more traffic simultaneously. The other routing algorithms were also unable to achieve the same PDR for MR traffic as in experiment 1. Nevertheless, for RPL+ the reduction was only 1%. The difference in favor of our machine learning-based routing protocol is more notable for the other traffic cases. For AE, the PDR of ML-RPL is 6% and 9% better than the PDR observed with RPL+ and MRHOF, respectively. Regarding the third traffic type, PQ, the 94% of PDR achieved by ML-RPL is better by 11% with respect MRHOF and 4% with respect to RPL+.

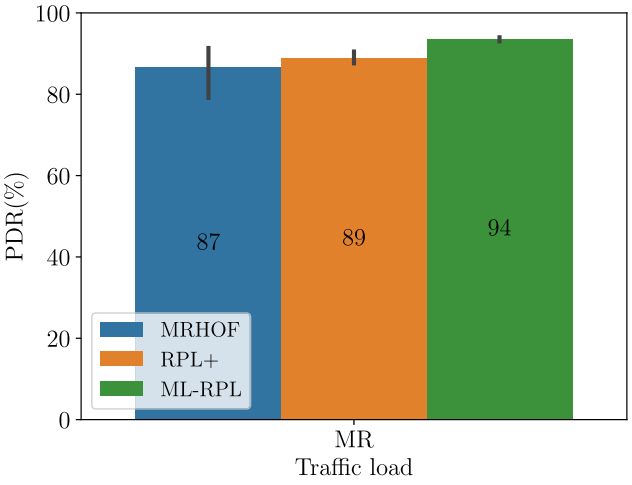


FIGURE 10. Packet delivery ratio in experiment 1.

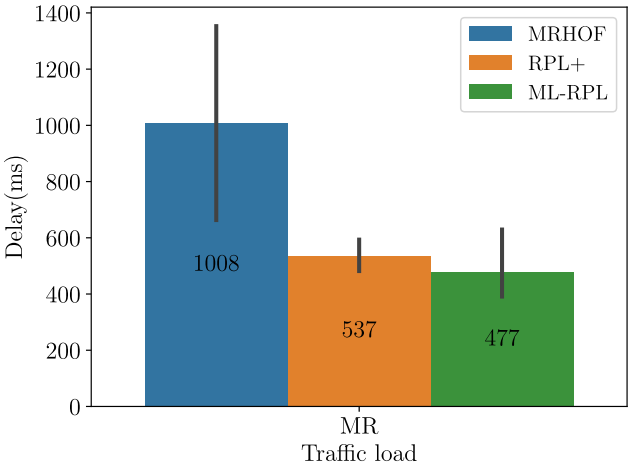


FIGURE 11. End-to-end delay in experiment 1.

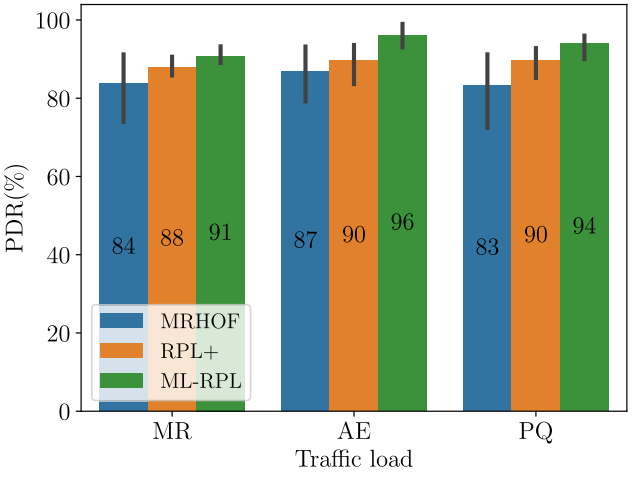


FIGURE 12. Packet delivery ratio in experiment 2.

The packet delay for the second experiment is shown in Fig 13. It can be seen again that MRHOF has the worst average packet delay. It is also seen that ML-RPL achieves

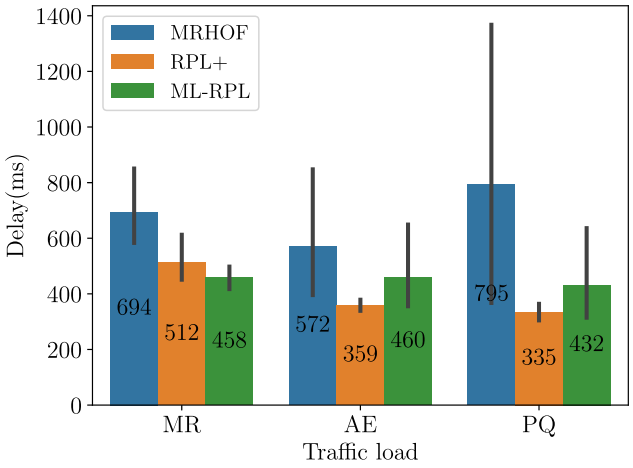


FIGURE 13. End-to-end delay in experiment 2.

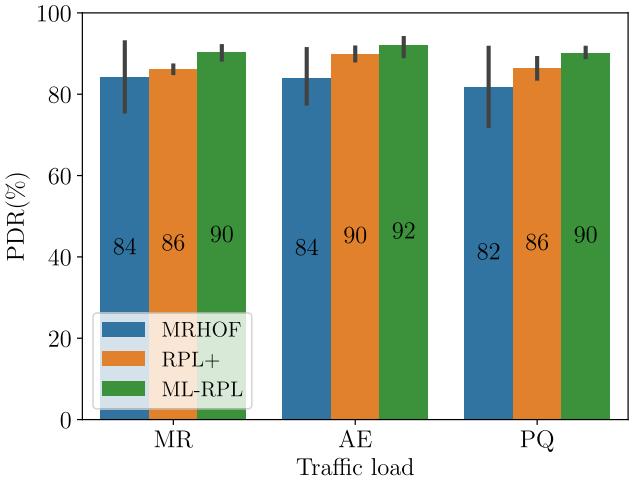


FIGURE 14. Packet delivery ratio in experiment 3.

slightly better delay that RPL+ for MR traffic as in experiment 1, but this is not the case for the rest of the traffic cases.

Fig. 14 shows the PDR results for the third experiment. In this experiment, we increased the network load by decreasing the sending interval of the MR application by 50% and doubling the percentage of the smart meters sending alarm reports and power quality events. The main goal is to see how the new proposal routing protocol behaves when the network load increases. The PDR shows that ML-RPL decreases slightly for each traffic category with respect to experiment 2. The reduction in PDR is 1% for MR, 4% for AE, and 4% for PQ. The PDR of RPL+ and MRHOF decreases as well for 2 out of 3 traffic types with respect to experiment 2. It was expected as the traffic load increases, the PDR would decrease. However, ML-RPL can keep the PDR above 90% for each traffic type, which is a significant advantage of our strategy based on ML techniques.

The end-to-end delay for the last experiment is depicted in Fig. 15. It shows a similar pattern as in experiment 2, but all the routing variants have higher values of delay for the three applications. Since the traffic load was increased, the transmission attempts face more contention, which results in longer back-off times and more re-transmission, so the delay increases in consequence. It is worth noting that there is a spike in the PQ application when ML-RPL is used. The average value is slightly bigger than even MRHOF, but we know from Fig. 14 that more smart meters are able to reach the destination when using ML-RPL. Some of those smart meters are further away from the collector, which results in an increase in the average end-to-end delay.

In summary, the improvement observed in our ML-based routing proposal compared to the other protocols can be attributed to several key factors. Firstly, we emphasize the data-driven approach employed in our design. In our specific context, this means that the parent selection process is guided by insights extracted from a previously generated dataset. This allows our routing protocol to make more accurate decisions when selecting the most suitable node for forwarding packets.

In addition, the incorporation of an ML model into our routing protocol design enables greater adaptability compared to traditional approaches. Our proposal has demonstrated its ability to adapt to varying network conditions and to learn the complex relationships between diverse features. This adaptability allows our ML-based routing protocol to make more accurate predictions even in the presence of network load variations, leading to improved performance metrics such as PDR for all test scenarios.

C. ROUTING OVERHEAD

The consumption of significant network resources and reduction of efficiency are major concerns in wireless networks due to routing overhead. This section will evaluate the routing overhead of our suggested routing protocol and compare it with MRHOF and RPL+.

Routing overhead is defined as the number of control messages generated by a routing protocol to maintain network connectivity and routing tables. In RPL, this involves DIO and DAO messages transmitted to keep network connectivity. Fig. 16 shows the average number of control messages generated in each experiment by each protocol.

As traffic load increases, the number of control messages tends to increase for all routing protocols. This is because nodes change their preferred parent more frequently when traffic load increases, looking for the best alternative. These changes generate extra control messages to notify neighbors about the change, which can also cause neighbors to change parents based on the new information. When nodes change parents too frequently, it can cause network instability and potential loops. Thus, there is a clear trade-off between changing to the best alternative candidate parent at the price of increasing network overhead and instability, and keeping the current parent even when it is not the best option.

Fig. 16 shows that ML-RPL has slightly less overhead than MRHOF. This is due to our protocol's consideration of additional factors in the state of the link and the state

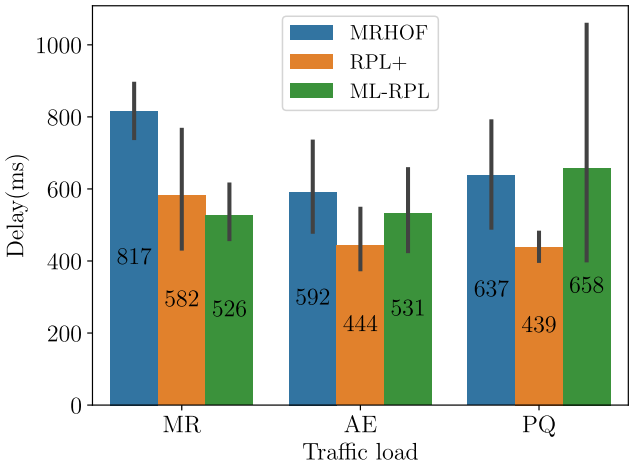


FIGURE 15. End-to-end delay in experiment 3.

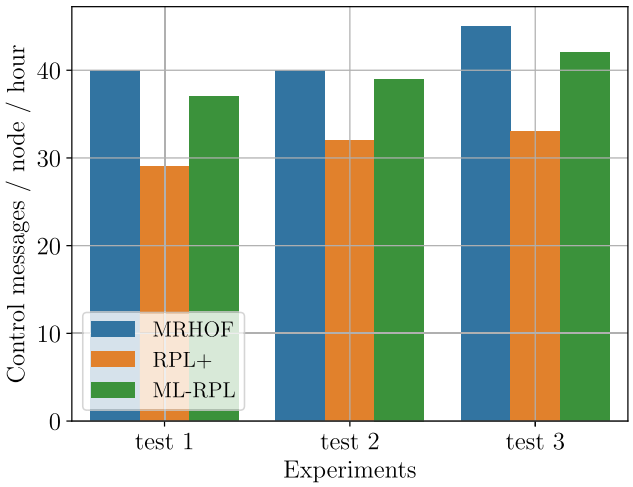


FIGURE 16. Overhead per experiment.

of the possible candidate parents, rather than relying solely on the ETX routing metric, as MRHOF does. Consequently, variations in the ETX value do not always prompt a parent change in our protocol.

Compared to RPL+, ML-RPL incurs more overhead. This difference in overhead can be attributed to certain key differences between the two protocols that affect their operation and performance. RPL+ is an extension of the RPL standard that uses the hop count metric as its base. It also takes into account other metrics such as ETX, throughput, MAC losses, and channel utilization, in a tie-breaking equation to decide among candidates with the same ranking in the routing tree, but the weights assigned to each of them are fixed. In contrast, ML-RPL is based on a Gradient boosted decision tree algorithm, which is a non-linear model that can capture complex interactions between features, resulting in a more dynamic routing protocol with wider ranges of prediction and higher variability in parent selection. Despite the increase in overhead, network performance is not degraded; in fact, it is beneficial based on the PDR. Therefore, our innovative protocol strikes a good balance between the other two routing protocols.

VI. CONCLUSION AND FUTURE WORK

We have presented a novel proposal of a machine learning-based routing protocol for Wireless Smart Grid Networks named ML-RPL. The proposal focuses on enhancing the parent selection strategy of one of the well-known protocols in this type of networks, the Routing Protocol for Low-Power and Lossy Networks. We started our design by gathering a large amount of simulation data from a real deployment of smart meters that helped us to build a representative dataset. Each feature in the dataset corresponds to a value of a routing metric that characterizes the link or the state of the nodes measured hop by hop as the packets travel from source to destination. This dataset served to train the ML algorithm and is later used to make the routing decisions. In addition, a feature importance analysis was conducted to use, by the protocol, only the routing metrics more relevant.

The performance evaluation of the new proposal showed significant improvements with respect to one of the standard RPL implementations in terms of packet delivery ratio and end-to-end delay. The proposal also overcame an RPL modification (RPL+) that uses Random Forest to improve the parent selection strategy. When comparing to RPL+, it can be seen that ML-RPL is consistently better in terms of PDR, and in some cases, the end-to-end delay achieved is also better. However, as more packets arrive at the destination when ML-RPL is used, some of those packets followed longer paths and consequently increase the average end-to-end delay with respect to RPL+.

In general, the results show a significant potential of the ML techniques to be applied in solving or improving networking tasks. The capacity of learning from a dataset that contains previous events and later use that knowledge has been the key for better results. For future work, we are strongly considering to incorporate Quality of Service (QoS) criteria in the decision-making process to guarantee the QoS of some applications with specific requirements. In addition, we plan to extend our routing strategy to other ML algorithms and conduct a detailed comparison among them.

REFERENCES

[1] J. P. Astudillo León, C. L. Duenas Santos, A. M. Mezher, J. Cardenas Barrera, J. Meng, and E. Castillo Guerra, "How does the selection of wireless technology impact the performance of the smart grid? A simulation approach," in *Proc. 19th ACM Int. Symp. Perform. Eval. Wireless Ad Hoc, Sensor, Ubiquitous Netw.* New York, NY, USA: Association for Computing Machinery, 2022, pp. 67–74.

[2] *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANS) Amendment 3: Physical Layer (PHY) Specifications for Low-Data-Rate, Wireless, Smart Metering Utility Networks*, Standard 802.15.4g-2012, 2011.

[3] *IEEE Draft Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications-Amendment 10: Mesh Networking*, Standard P802.11bb/D5.0, Dec. 2022.

[4] U. Raza, P. Kulkarni, and M. Sooriyabandara, "Low power wide area networks: An overview," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 855–873, 2nd Quart., 2017.

[5] *What is it ? A Technical Overview of LoRa*, LoRa Alliance, San Ramon, CA, USA, 2015.

[6] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "Overview of cellular LPWAN technologies for IoT deployment: Sigfox, LoRaWAN, and NB-IoT," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2018, pp. 197–202.

[7] R. A. T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, and J. P. Vasseur, *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*, document RFC 6550, 2012.

[8] B. Ghaleb, A. Y. Al-Dubai, E. Ekonomou, A. Alsarhan, Y. Nasser, L. M. Mackenzie, and A. Boukerche, "A survey of limitations and enhancements of the IPv6 routing protocol for low-power and lossy networks: A focus on core operations," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1607–1635, 2nd Quart., 2019.

[9] L. Lemus Cárdenas, J. P. Astudillo León, and A. M. Mezher, "GraTree: A gradient boosting decision tree based multimetric routing protocol for vehicular ad hoc networks," *Ad Hoc Netw.*, vol. 137, Dec. 2022, Art. no. 102995.

[10] J. P. Astudillo León, L. J. de la Cruz Llopis, and F. J. Rico-Novella, "A machine learning based distributed congestion control protocol for multi-hop wireless networks," *Comput. Netw.*, vol. 231, Jul. 2023, Art. no. 109813.

[11] G. Iyer, P. Agrawal, E. Monnerie, and R. S. Cardozo, "Performance analysis of wireless mesh routing protocols for smart utility networks," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Oct. 2011, pp. 114–119.

[12] L. Lemus Cárdenas, A. M. Mezher, J. P. Astudillo León, and M. A. Igartua, "DTMR: A decision tree-based multimetric routing protocol for vehicular ad hoc networks," in *Proc. 18th ACM Symp. Perform. Eval. Wireless Ad Hoc, Sensor, Ubiquitous Netw.*, 2021, pp. 57–64.

[13] J. P. Astudillo León, F. J. Rico-Novella, and L. J. de la Cruz Llopis, "Predictive traffic control and differentiation on smart grid neighborhood area networks," *IEEE Access*, vol. 8, pp. 216805–216821, 2020.

[14] B.-S. Kim, B. Suh, I. J. Seo, H. B. Lee, J. S. Gong, and K.-I. Kim, "An enhanced tree routing based on reinforcement learning in wireless sensor networks," *Sensors*, vol. 23, no. 1, p. 223, Dec. 2022.

[15] A. Kumar and N. Hariharan, "DCRL-RPL: Dual context-based routing and load balancing in RPL for IoT networks," *IET Commun.*, vol. 14, no. 12, pp. 1869–1882, Jul. 2020.

[16] C. L. Duenas Santos, J. P. Astudillo León, A. M. Mezher, J. Cardenas Barrera, J. Meng, and E. Castillo Guerra, "RPL+: An improved parent selection strategy for RPL in wireless smart grid networks," in *Proc. 19th ACM Int. Symp. Perform. Eval. Wireless Ad Hoc, Sensor, Ubiquitous Netw.*, 2022, pp. 75–82.

[17] S. Raschka and V. Mirjalili, *Python Machine Learning: Machine Learning and Deep Learning with Python, Scikit-Learn, and TensorFlow*. Birmingham, U.K.: Packt Publishing Ltd, 2017.

[18] P. D. Acevedo, D. Jabba, P. Sanmartín, S. Valle, and E. D. Nino-Ruiz, "WRF-RPL: Weighted random forward RPL for high traffic and energy demanding scenarios," *IEEE Access*, vol. 9, pp. 60163–60174, 2021.

[19] A. Musaddiq, Y. B. Zikria, and S. W. Kim, "Routing protocol for low-power and lossy networks for heterogeneous traffic network," *EURASIP J. Wireless Commun. Netw.*, vol. 2020, no. 1, pp. 1–23, Dec. 2020.

[20] W. Mardini, S. Aljawarneh, and A. Al-Abdi, "Using multiple RPL instances to enhance the performance of new 6G and Internet of Everything (6G/IoE)-based healthcare monitoring systems," *Mobile Netw. Appl.*, vol. 26, no. 3, pp. 952–968, Jun. 2021.

[21] K. S. Bhandari, I. Ra, and G. Cho, "Multi-topology based QoS-differentiation in RPL for Internet of Things applications," *IEEE Access*, vol. 8, pp. 96686–96705, 2020.

[22] E. P. Thubert, *Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)*, document RFC 6552, 2012, pp. 5–48.

[23] P. L. O. Gnawali, *The Minimum Rank With Hysteresis Objective Function*, document RFC 6719, 2012.

[24] *OMNeT++ Discrete Event Simulator*. Accessed: Jan. 31, 2023. [Online]. Available: https://omnetpp.org/

[25] H.-S. Kim, H. Cho, H. Kim, and S. Bahk, "DT-RPL: Diverse bidirectional traffic delivery through RPL routing protocol in low power and lossy networks," *Comput. Netw.*, vol. 126, pp. 150–161, Oct. 2017.

[26] K. Kritsis, G. Z. Papadopoulos, A. Gallais, P. Chatzimisios, and F. Théoleyre, "A tutorial on performance evaluation and validation methodology for low-power and lossy networks," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 1799–1825, 3rd Quart., 2018.

[27] D. Cypher, *NISTIR 7761 NIST Priority Action Plan 2 Guidelines for Assessing Wireless Standards for Smart Grid Applications NIST Priority Action Plan 2 Guidelines for Assessing Wireless Standards for Smart Grid Applications*, document NISTIR 7761, Feb. 2011, pp. 1–104.

[28] G. Rajalingham, Y. Gao, Q.-D. Ho, and T. Le-Ngoc, "Quality of service differentiation for smart grid neighbor area networks through multiple RPL instances," in *Proc. 10th ACM Symp. QoS Secur. Wireless Mobile Netw.*, Sep. 2014, pp. 17–24.

[29] W. McKinney, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, S. van der Walt and J. Millman, Eds., 2010, pp. 51–56.

[30] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

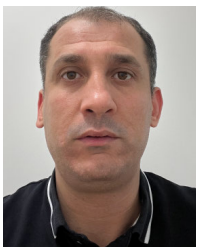
[31] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–11.

[32] A. Veronika Dorogush, V. Ershov, and A. Gulin, "CatBoost: Gradient boosting with categorical features support," 2018, *arXiv:1810.11363*.

[33] G. Hackeling, *Mastering Machine Learning With Scikit-Learn*. Birmingham, U.K.: Packt Publishing Ltd, 2017.

[34] A. Tharwat, "Classification assessment methods," *Appl. Comput. Informat.*, vol. 17, no. 1, pp. 168–192, Jan. 2021.

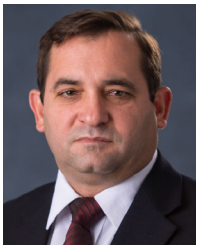
CARLOS LESTER DUENAS SANTOS (Graduate Student Member, IEEE) received the B.Sc. degree in telecommunication and electronics and the M.Sc. degree in telecommunication from the University "Marta Abreu" of Las Villas, Santa Clara, Cuba, in 2008 and 2015, respectively. He is currently pursuing the Ph.D. degree with the Emera and NB Power Research Center for Smart Grid Technologies, Electrical and Computer Engineering Department, University of New Brunswick, Canada. His research interests include smart grid communications, routing protocols, network security, and machine learning.



AHMAD MOHAMAD MEZHER received the M.S. degree in signals and systems from the Central University "Marta Abreu" of Las Villas, Santa Clara, Cuba, in 2011, and the Ph.D. degree in network engineering from Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 2016. He currently holds the McCain Postdoctoral Fellowship of innovation with the Electrical and Computer Engineering Department, University of New Brunswick (UNB). His research interests include smart grid communications, vehicular ad hoc networks, data privacy, and load forecasting. He was a recipient of a FI-AGAUR Fellowship from the Generalitat de Catalunya and the Social European Budget.

JUAN PABLO ASTUDILLO LEÓN received the Ph.D. degree (Hons.) in network engineering from Universitat Politècnica de Catalunya (UPC), Spain, in 2020. He was a Postdoctoral Fellow with the University of New Brunswick (UNB), Canada. He worked as an Undergraduate with the Universidad Politècnica Salesiana (UPS), Ecuador. He is currently a full-time Professor with Yachay Tech University, Urcuqui, Ecuador, and a Postgraduate Professor with Pontificia Universidad Católica del Ecuador (PUCE), Sede Manabí, Ecuador. He has been involved in several national and international projects and has authored international publications in conferences and journals. His research interests include the application of artificial intelligence in wireless multi-hop networks and the development of the IoT smart services. He received the Best Ph.D. Thesis Prize of Information and Communication Technologies from Escola de Doctorat, UPC.

JULIAN CARDENAS BARRERA received the Ph.D. degree in electrical engineering from Universidad Central Marta Abreu de Las Villas, Santa Clara, Cuba, in 1994. He is currently an Associate Professor with the Department of Electrical and Computer Engineering and the NB Power Industrial Research Chair with the Emera and NB Power Research Center for Smart Grid Technologies, University of New Brunswick. His research interests include digital signal processing, the optimization of renewable energy systems, energy forecasting, and smart grid communications. He is a registered Professional Engineer with the Province of New Brunswick, Canada.



EDUARDO CASTILLO GUERRA (Senior Member, IEEE) received the B.Sc. degree in electrical engineering and the M.Sc. degree in telecommunication from the University of "Marta Abreu" Las Villas, Santa Clara, Cuba, in 1992 and 1996, respectively, and the Ph.D. degree in electrical engineering from the University of New Brunswick in 2003. He is currently a Professor with the Department of Electrical and Computer Engineering, University of New Brunswick. His research interests include modeling digital systems, artificial intelligence, digital signal and speech processing, the optimization of renewable energy systems, digital circuit and sensor design, voice authentication, and secure communications. He is a registered Professional Engineer in the Province of New Brunswick, Canada.

JULIAN MENG (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Queen University, Kingston, ON, Canada, in 1993. His research interests include adaptive signal estimation, nonlinear signal processing, renewable energy, and intelligent systems. He is a registered Professional Engineer of the Association of Professional Engineers and Geoscientists NB.