COMNETS: COst-sensitive learning for throughput optimization in Multi-radio IoT NETworkS

Jothi Prasanna Shanmuga Sundaram*, Magzhan Gabidolla*, Miguel Á. Carreira-Perpiñán*, Alberto E. Cerpa*
*Computer Science and Engineering, University of California, Merced, CA 95343, USA

***Abstract*—Mesoscale IoT applications, such as P2P energy trade and real-time industrial control systems, demand high throughput and low latency, with a secondary emphasis on energy efficiency as they rely on grid power or large-capacity batteries. MARS, a multi-radio architecture, leverages ML to instantaneously select the optimal radio for transmission, outperforming single-radio systems. However, MARS encounters a significant issue with cost sensitivity, where high-cost errors account for 40% throughput loss. Current cost-sensitive ML algorithms assign a misclassification cost for each class, but not for each data sample. In MARS, each data sample has different costs, making it tedious to employ existing cost-sensitive ML algorithms. First, we address this issue by developing COMNETS, an ML-based radio selector using oblique trees optimized by (TAO). TAO incorporates sample-specific misclassification costs to avert high-cost errors, and achieves a 50% reduction in the decision tree size, making it more suitable for resource-constrained IoT devices. Second, we prove the stability property of TAO and leverage it to understand the critical factors affecting the radio-selection problem. Finally, our real-world evaluation of COMNETS at two different locations shows an average throughput gain of 20.83%, 17.39% than MARS.**

I. Introduction

The Internet of Things (IoT) is rapidly utilized in day-to-day lives for asset monitoring, home automation, and Industrial automation as it reduces the human effort involved. Conventional IoT applications were utilized in small-scale deployments like home automation, that span up to 100m. These applications utilize short-range radios like Zigbee and Bluetooth Low-Energy (BLE). Due to recent advancements in the IoT radio technology, long-range radios like WiFi-HaLow, LoRa, Sig-Fox, and NB-IoT were developed [1] for large-scale applications, like Microsoft Farmbeats [2], ranging between 1-5Km. The applications that range between 100-1000m, like Industrial automation[3], P2P energy-trade in smart meters [4], [5], are termed as the mesoscale IoT applications.

Mesoscale IoT applications are identified as the new and upcoming category of IoT applications [6]. Unlike the traditional IoT applications that have energy constraints, the mesoscale applications are either grid-powered [7], [8], [9] or have a large-battery reserve [5], [10], [4]. These mesoscale applications range between 100-1000m, with no specialized radios developed to cater to these applications [6]. To solve this issue, MARS [6] employs a multi-radio architecture comprising of multihop Zigbee and single hop LoRa radios with radio selection using axis-aligned trees optimized by the TAO [11]. MARS identifies the *gray-region*, 500-1200m from the gateway, where Zigbee and LoRa radios achieve competitive throughput. Due to the erratic link quality variations in the mesoscale environment, it is uncertain which radio will achieve better throughput at a given time instant. So, MARS formulated the radio selection problem as a classification problem with 0/1 loss, to maximize the throughput.

This work is partially supported by NSF award IIS–2007147.

Table I: Cost-sensitivity in radio selection

	T1	T2
Zigbee Throughput	7000	4600
LoRa Throughput	2000	4500
Throughput Loss	5000	100
Category	High-cost	Low-cost

Classification problems in machine learning focus on maximizing classification accuracy, assuming equal cost for all misclassification errors [12], [13], [14]. However, classification accuracy is not the only important factor in many real-world applications, and each misclassification error may incur different costs [15], [16].

For example, in brain tumor detection applications, wrongly identifying a healthy patient as a tumor patient has a different cost than identifying a tumor patient as a healthy patient. Churn modeling identifies the customers that are more likely to leave a service provider. Accurately identifying the churner is the task at hand but identifying the profitable churner and unprofitable churner is more important in this context [17]. In marketing applications, wrongly identifying that a specific customer will not accept the offer when they are ready to accept the offer costs money while the vice-versa prediction error costs time [18]. Finally, in intrusion detection applications, classifying malicious connections as benign has a different cost than the vice-versa classification.

Cost-sensitivity in radio selection. In the context of ML-based radio selection in multi-radio IoT networks, cost of misclassification can be defined as the difference in the instantaneous throughput of the two radios at a given time instance. It is to be noted that this equation is valid only when the ML model misclassifies low-throughput radio as high-throughput radio. This can be formulated as shown in equation 1 below:

$$C(t) = |Z(t) - L(t)|, \quad \forall t \in T \tag{1}$$

where T is the set of all time instances, $C(t)$ is the cost of misclassifying a low-throughput radio as the high-throughput radio at time instance t, $Z(t)$ is the throughput of a Zigbee radio at time instance t and $L(t)$ is the throughput of a LoRa radio at time instance t.

Table. I illustrates the importance of cost in the radio selection problem. In the radio-selector problem, when one radio is achieving higher throughput than the other, the difference in throughput will not be the same for all the instances. For example, at T1, let's assume that the throughput of the ZigBee radio is 7000 bps and the throughput of LoRa is 2000 bps. The throughput difference between these two radios at this time instance is 5000 bps. At T2, say the throughput of a Zigbee radio is 4600 bps and the throughput of a LoRa radio is 4500 bps. The throughput difference is only 100 bps. At T1, assuming that the radio selector algorithm misclassifies the low-throughput LoRa radio as the high-throughput radio costs 5000 bps, while a misclassification at T2 costs only 100 bps. Misclassification at time T2 is tolerable since there is little throughput gain, while the misclassification at time T1 is intolerable for the multi-radio system.

Cost-sensitive ML algorithms in the literature define a cost matrix associated with misclassifying each class. Here, the misclassification cost is always tied to a class instead of a specific data sample [15]. In the radio selector problem, the cost varies for each data sample. Cost-sensitive optimization for each data sample is tedious (refer §IV for more details). An intuitive way is to utilize the cost in the loss function of the ML models. Our experiments with five-fold cross-validation show that this method gives a model that is not well generalized for unseen data (refer §V-B).

First, we employed Oblique-trees optimized with TAO[11] algorithm to solve this problem. TAO[11] does this optimization gracefully to avoid high-cost errors as explained in §IV. To the best of our knowledge, we are the first to employ TAO[11] to optimize Oblique trees for this cost-based optimization. Compared to the other commonly used models like SVM, Logistic Regression and CART, TAO-Oblique trees achieve the best results. The TAO[11] optimization algorithm not only optimizes to avoid high-cost errors, but also makes the trees 50% smaller compared to the other models.

Second, we interpret the essence of the radio selection problem using the tree stability property of TAO[11] algorithm. TAO not only optimizes cost-sensitive oblique trees and makes them smaller, but it also provides stability to the tree when new data are added. In general, the decision trees (axis-aligned/Oblique) are easy to interpret. The reason they are not extensively used is that when new data are added, they induct a completely new tree with different rules [19], [20]. TAO[11] optimization on Oblique trees overcomes this issue by providing stability to the tree, i.e. TAO optimization reduces this drastic change in the tree structure even when new data are added. This helps to understand the essence of the radio selection problem at hand. Since TAO-Oblique trees use a linear combination of multiple features, it is tedious to extract a meaningful reason from the interpretations. So, we use TAO[11] optimized sparse oblique trees to solve this issue. Sparse-oblique trees reduce the number of parameters used in each decision node, making it easier to extract a meaningful reason for the interpretations.

Finally, the TAO[11]-Oblique trees are converted into IF...ELSE statements for easier deployment in resource-constrained IoT devices. Our real-world, large-scale evaluation of COMNETS, powered by TAO[11]-Oblique tree, shows a throughput gain of 20.83% & 17.39% over the state-of-the-art MARS, at locations A & B respectively.

In summary, the contributions of our work are:

- We showed the importance of cost-sensitivity in multi-radio networks through in-field experiments.
- We developed TAO[11]-Oblique trees that optimize to avert high-cost classification errors.
- We provided insights into the radio selection problem by exploiting the stability property of the TAO[11].

II. Related Work

Bahl et al. [21] showed that a multi-radio system is beneficial for wireless networks. A lot of multi-radio wireless systems [22], [23] has been developed to optimize the performance of the networks like energy-efficiency [24], [25], [26] and routing management [27]. Backpacking [28] was developed for high data rate sensor networks. Kusy et al. [25] developed a multi-radio architecture for WSN. They show that employing two radios with multi-hop network topology in the same node improves reliability with 3-33% energy overhead. LoRaCP [29] employs a ZigBee+LoRa multi-radio network for faster control packet transmissions in multi-hop WSN. Gummeson et al. [30] optimize energy consumption by employing a Reinforcement Learning (RL) based adaptive link layer to switch radios (CC2420+XE1205) based on channel dynamics. Using RL is detrimental because of: (i) Very high training data requirements and, (ii) inference latency[6]. Lymberopoulos et al. [26] switch radios (Zigbee+WiFi) with a threshold-based algorithm optimizing for energy efficiency.

While most of the above multi-radio systems developed for IoT networks optimize for energy efficiency over small-scale deployments, MARS [6] optimizes for throughput and latency on mesoscale applications. Initially, they identified the absence of a fully developed specialized radio for emerging mesoscale IoT applications. To close this gap, they first conducted a qualitative analysis on the suitability of all the available IoT radios for mesoscale application environments and identified that Zigbee2.4GHz and LoRa915MHz are the better candidates. Their analytic and experimental analysis with these two radios shows that Zigbee and LoRa achieve competitive throughput in the *gray-region*, 500-1200m from the gateway [6]. Since it is uncertain which radio will provide higher throughput at the time of transmission, they developed a Decision Tree (DT) model using axis-aligned trees to predict the high-throughput radio and further optimize this DT model with Tree-Alternating Optimization (TAO) algorithm[11]. This ML model needs the link quality of LoRa and the instantaneous path quality estimations of the Zigbee radios. The traditional path quality estimations of Zigbee are not instantaneous. Hence, they develop a DT-based instantaneous path quality estimation to provide instantaneous inputs for the ML model. On taking the instantaneous inputs, their ML model will output the high-throughput radio to transmit the scheduled packet.

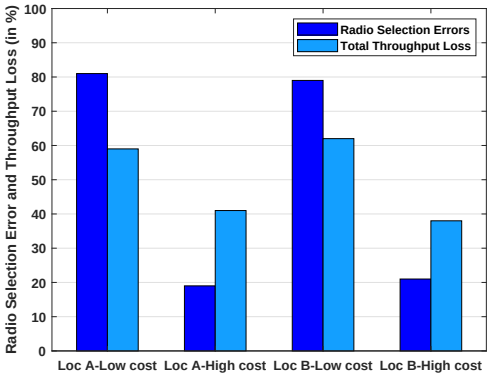


Figure 1: Motivation for cost-sensitive learning

Table II: Throughput gain and optimization margin

	Average throughput gain (in %)	
	Location A	Location B
MARS	49.79	48.20
PERFECT OPTIMIZER	73.23	69.50
OPTIMIZATION MARGIN	23.44	21.30

III. Motivation

We quantified this cost sensitivity with traces obtained from real-world experiments. From these traces, we categorized the radio selection errors as high-cost and low-cost errors. Low-cost errors have a throughput difference between two radios less than 200 bps. High-cost errors have a throughput difference greater than 200 bps.

Figure 1 shows our motivation to pursue cost-sensitive learning for this radio selector problem. In Location A, the low-cost errors amount to 81% of the total radio selection errors costing 59% of the total throughput loss. The high-cost errors amount to 19% of the total radio selection errors costing 41% of the total throughput loss. Even though the difference between the number of high-cost and low-cost errors is very high, the difference between the total throughput loss between these two categories of errors is very low. A similar trend is seen at Location B. If the decision trees are optimized to avoid high-cost errors, nearly 20% of the radio selection errors can be avoided, leading to a reduction of 40% total throughput loss in the multi-radio system.

The throughput gain and optimization margin are tabulated in Table II. The average throughput gain of MARS in Location A and Location B is 49.79% and 48.2% respectively. Assuming that a perfect optimizer algorithm can avert all high-cost errors, the average throughput gain of the multi-radio system at Locations A & B will be increased to 73.23% and 69.50% giving an optimization margin of 23.44% and 21.30% respectively. This motivates us to pursue the cost-sensitivity problem for multi-radio IoT networks.

IV. Optimizing cost-sensitive loss

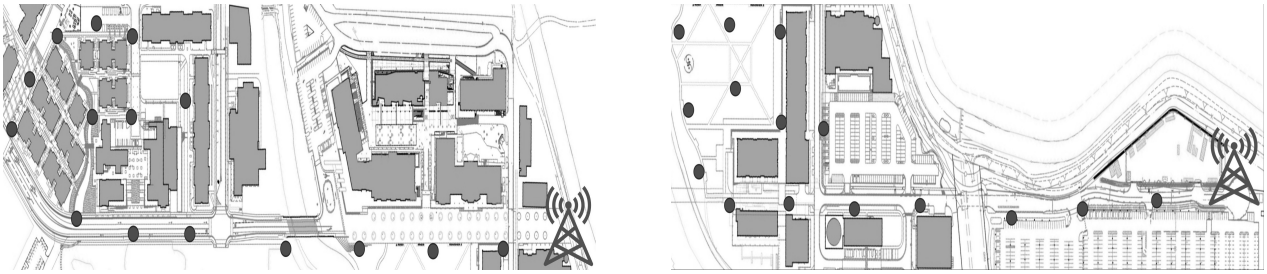
From a machine learning perspective, our problem consists of a cost-sensitive binary classification problem, where the two classes are LoRa, ZigBee radios. At any point in time for a given state of the system, if the ML model makes the correct prediction, i.e. chooses the radio with the highest throughput, then there is no corresponding loss for this case. But if the model makes an incorrect prediction at a given time instance, then there is a corresponding loss, which is the opportunity cost of choosing the radio with better throughput, whose amount is instance/time-specific. In order to train an ML model that optimizes for the best possible throughput, we first need to define a desired objective (loss) function.

Let $\{\mathbf{x}_n, y_n, c_n\}_{n=1}^N$ be our training set of N points, where $\mathbf{x} \in \mathbb{R}^D$ is a D-dimensional feature vector describing the current state of the system, $y \in \{0, 1\}$ is a class label indicating the radio with better throughput, and $c \in \mathbb{R}_{>0}$ is a cost (weight) of the instance obtained by the difference between two types of radio throughputs. Let $T(\mathbf{x}; \boldsymbol{\Theta})$: $\mathbb{R}^D \rightarrow \{0, 1\}$ be an oblique decision tree with learnable parameters $\boldsymbol{\Theta}$. Based on the aforementioned discussion on the importance of costs in our setting, we aim to optimize the loss function:

$$\min_{\boldsymbol{\Theta}} L(\boldsymbol{\Theta}) = \sum_{n=1}^{N} c_n \cdot L(y_n, T(\mathbf{x}_n; \boldsymbol{\Theta})) \tag{2}$$

where $L(\cdot, \cdot)$ is a 0/1 loss. With the inclusion of costs $\{c_n\}_{n=1}^N$, the objective function of (2) puts more importance on instances where there is a large gap between the throughputs of the two radio types while misclassifying instances with small throughput differences is less important. It captures exactly what we are after in our radio selection problem: better overall throughput. However, the loss function $L(\cdot, \cdot)$ is not differentiable, and with another highly non-differentiable function as a decision tree $T(\mathbf{x}; \boldsymbol{\Theta})$ makes the learning problem of (2) much harder to optimize.

To optimize a cost-sensitive 0/1 misclassification loss over a decision tree $T(\mathbf{x}; \boldsymbol{\Theta})$, we rely on the TAO algorithm. It is a general algorithmic framework that can optimize different types of loss functions over various tree-based methods, and we will apply it here to learn cost-sensitive oblique decision trees. At a higher level, the TAO algorithm operates very differently than traditional tree learning methods such as CART or C5.0. Instead of growing a tree greedily starting from a root node based on some impurity criteria, it takes an initial tree of some predetermined fixed structure and initial parameters (e.g. a complete tree of depth Δ or a greedily induced tree), and performs alternating optimization steps over each node parameters, and with each such step guaranteeing a monotonic decrease of an objective function. Conceptually, it operates similarly to how neural networks are learned: defining a model architecture (cf. tree structure) and initial parameters (cf. tree parameters), and optimizing them with gradient-based methods (cf. alternating optimization). Below we describe more formally the decision tree $T(\mathbf{x}; \boldsymbol{\Theta})$, and provide the specific details of optimization steps in TAO for the cost-sensitive loss. The predictive function $T(\mathbf{x}; \boldsymbol{\Theta})$ of a decision tree works by routing an instance $\mathbf{x}$ through a root-to-leaf path where each decision node $i \in \mathcal{D}$ ($\mathcal{D}$ is the set decision node indices, $\mathcal{L}$ is the set of leaf indices) in the path uses a



(a) Location A - Mesh topology (b) Location B - Mesh topology

Figure 2: Mesh topology set up at locations A and B

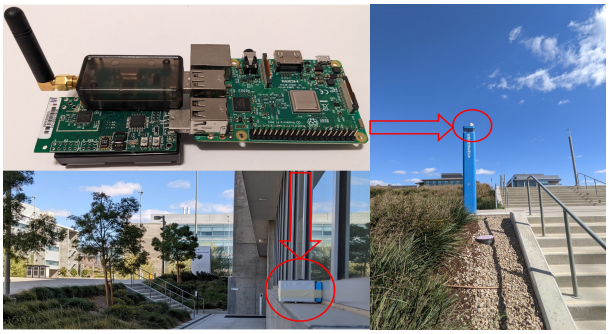


Figure 3: Multi-radio hardware

decision function $g_i(\mathbf{x};\boldsymbol{\theta}_i)$: $\mathbb{R}^D \rightarrow \{\text{left}_i, \text{right}_i\} \subset \mathcal{D} \cup \mathcal{L}$ to decide which child node the instance $\mathbf{x}$ will be sent next. In this paper, we consider oblique trees that use a linear model at decision nodes: $g_i(\mathbf{x};\boldsymbol{\theta}_i) = \mathbf{w}_i^T\mathbf{x} + \mathbf{w}_{i0}$ with learnable parameters $\boldsymbol{\theta}_i = \{\mathbf{w}_i, \mathbf{w}_{i0}\}$. The actual prediction of the tree $T(\mathbf{x};\boldsymbol{\Theta})$ happens in the leaf, where it just outputs the leaf's class label i.e. the radio type. So a leaf $j \in \mathcal{L}$ has only a single learnable parameter $\theta_j \in \{0, 1\}$. With learnable parameters explicitly defined, we now include regularization (penalty) terms on them in the objective function. The main purpose of this is to obtain models with better generalization performance as well as remaining small in the number of parameters and size. In particular, we include ℓ_1 norm penalty on decision node hyperplanes. This helps to produce sparse hyperplanes, i.e. those with few nonzero weights, and if the regularization term is high enough, then the entire weight vector will turn to zero. A decision node with few nonzero weights are typically more interpretable, and the one that has entirely zero hyperplane sends all instances to only one child ($g_i(\mathbf{x};\boldsymbol{\theta}_i) = \mathbf{0}^T\mathbf{x} + \mathbf{w}_{i0} = \mathbf{w}_{i0}$), whereby an entire other child subtree can be pruned. With regularization terms included, the final objective function that we optimize is the following:

$$\min_{\boldsymbol{\Theta}} E(\boldsymbol{\Theta}) = \sum_{n=1}^{N} c_n \cdot L(y_n, T(\mathbf{x}_n;\boldsymbol{\Theta})) + \lambda \sum_{i\in\mathcal{D}} \|\mathbf{w}_i\|_1 \quad (3)$$

Given some tree structure and initial parameters, the TAO algorithm optimizes the above objective by monotonically decreasing it in each step because of the following theorems:

Seperability. states that the objective function (3) separates over sets of any non-descendant nodes in the tree given the other fixed nodes. This stems from the fact that the tree makes hard decisions, and that the training points that reach any set of non-descendant nodes are disjoint. The implications of this theorem are that the non-descendant set of nodes can be independently optimized in parallel. The exact form of the node optimization depends on the node type (i.e. a decision node or a leaf) as explained below.

Reduced problem over decision nodes. Optimizing the top-level objective function (3) over the parameters of a decision node $i \in \mathcal{D}$ reduces to a simpler problem of a weighted 0/1 loss binary classification that depends only on the training instances that reach this node i. This follows from the fact that the only thing a decision node can do is to send a point $\mathbf{x}_n$ either to the left or to the right child subtrees. If both child subtrees classify the point $\mathbf{x}_n$ correctly (or incorrectly), then it does not matter where to send it, and so those points can be discarded from the reduced problem. But if one child subtree classifies it correctly, while the other does not, then we want the decision function $g_i(\mathbf{x}_n;\boldsymbol{\theta}_i)$ to learn to send it to the "correct" child subtree. Repeating this observation for all training points in the reduced set, we can form a binary classification problem, where the ground truth binary labels correspond to the "correct" child for each point. And the problem is weighted because each training instance carries the original costs from the top-level objective function. There is also an ℓ_1 penalty on decision node weights $\|\mathbf{w}_i\|_1$ that directly comes to the reduced problem from the top-level objective. Solving exactly a 0/1 misclassification loss over a linear model (and with ℓ_1 penalty) is in general NP-hard problem, but we can approximate this reduced problem by a convex surrogate such as logistic regression. In our implementation, we use an ℓ_1-regularized logistic regression solver inside the LIBLINEAR library [31]. To ensure a monotonic decrease of the objective, we can update the decision node parameters by the logistic regression solution only if it improves over the previous one in terms of the weighted 0/1 loss.

Reduced problem over leaf nodes. The top-level optimization problem (3) over a leaf node $j \in \mathcal{L}$ is simply the same loss function but over a leaf parameter θ_j and over the set of training points that reach this leaf. With simple constant leaf nodes, the optimal solution is just a weighted majority class.

The theorems underlying the TAO algorithm tell us how individual nodes must be optimized but do not prescribe the order the nodes must be visited. We follow the reverse breadth-first-order approach similar to the original paper on

TAO [11]: all nodes at the deepest level (from root) are optimized first, then their parents, and so on, until the root node. This is repeated multiple times (at most 20) until the objective function converges. As for the initial tree structure and parameters, we experiment with either a random complete tree of depth Δ or the one obtained from a CART algorithm and choose the best one (in terms of validation error).

V. COMNETS ML MODEL

In this section, we explain the multi-radio system we built, the topology of our deployments for data collection, and the process of building the machine learning models.

Multi-radio hardware is shown in Figure 3. This consists of a Raspberry Pi 3B hosting two radios, (i)USB-based TelosB [32] Zigbee mote and (ii) a USB-based LoStik LoRa [33] radio. This multi-radio node is powered by a portable external power bank. We protected this node using a PVC case during deployments. The Raspberry Pi 3B will send a command to both the TelosB [32] and LoStik [33] radios, once every 3 seconds, to transmit a 29-byte packet, destined to the gateway, concurrently. LoRa radios employ ALOHA MAC while Zigbee radios use CSMA MAC. Link-level acknowledgments are disabled in the TelosB [32] Zigbee motes.

Topology setup is done with 15 multi-radio end nodes and one gateway. LoRa radios form a single-hop network where a LoRa radio can directly communicate with the gateway and Zigbee radios form a multi-hop mesh topology to reach the gateway. Multi-hop Zigbee network uses a distance-vector routing [34] protocol for multi-hop routing. Since our region of interest is the *gray-region* [6], most of the multi-radio nodes are populated in the *gray-region* (0.5-1.2Km from gateway) at two different locations A (Fig. 2a) and B (Fig. 2b).

Data Collection is done on the mesh topology deployed in both the above-mentioned locations. Data is collected from the nodes populated in the *gray-region* [6]. A total of 25,500 data packets were recorded. This comprehensive data set covers all the different dynamics of the deployed environment. The throughput of each transmitted data packet is recorded. A Python script labels this data set to identify the high-throughput radio and its associated loss for each transmitted data packet. The associated loss is calculated as the difference in the throughput of a high and low-throughput radio. We formulate this radio selection problem as a classification problem and develop multiple classification models, that take in the input feature vector engineered with domain knowledge, following the same procedure as done in MARS [6].

Problem formulation, Feature Selection and Engineering. We formulated this radio selection problem as a classification problem and developed multiple classification models, that take in the input feature vector engineered with domain knowledge, following the same procedure of MARS [6].

The classification model takes the input features E2E path quality of LoRa radios (E2E-PQ_{LoRa}), and the E2E path quality of Zigbee radios (E2E-PQ_{Zigbee}) to output a high throughput radio. The input feature vector is expressed as:

$$Input_i = [HN_Z, E2E\_RSSI_L, E2E\_PRR_Z, E2E\_RNP_Z] \tag{4}$$

HN is the Hop Number that denotes a node's distance from the gateway in terms of hops. This value is obtained from the multi-hop Zigbee radios running a DV routing protocol [34] for multi-hop Zigbee communications. $E2E\_RSSI_{LoRa}$ is the End-to-End RSSI of LoRa radios. This is obtained by exploiting channel reciprocity as done in MARS[6]. $E2E\_PRR_{Zigbee}$ is the end-to-end packet reception ratio of Zigbee radios and $E2E\_RNP_{Zigbee}$ is the Required Number of Packets [35] obtained from Zigbee radios.

The output of the machine learning model is the radio predicted to have higher instantaneous throughput. This can be expressed as:

$$Output_i = [Zigbee|LoRa] \tag{5}$$

A. Cost-weighted accuracy metric

The decision trees use the traditional 0/1 accuracy to calculate the train/test accuracy of the models. Since we develop models that are optimized to avert all the high-cost errors, we use the Cost-Weighted Accuracy (CWA) metric as explained below:

$$CWA = \frac{\sum_n c_n I(y_n, T(x_n))}{\sum_n c_n}.100 \tag{6}$$

where c is the cost. In the radio selector problem, the cost is the throughput loss incurred as the result of choosing a low-throughput radio instead of a high-throughput radio. I(p,q) is an indicator function that is equal to one when p = q, y_n is the ground truth, T(x_n) is the classification model that takes input x_n and returns an output.

B. Models and their prediction accuracies

The test and train CWA of three widely used models SVM [36], Logistic Regression [37] and CART [38] decision trees are tabulated in Table III. These models are trained with 1500 samples like MARS [6].

The accuracies of these models are modified to account for loss as shown in equation 6. The training and testing accuracies averaged over a five-fold cross-validation are tabulated in Table III. SVM shows a higher testing accuracy than the training accuracy. This means the model is not well-generalized for unseen data. While Logistic Regression shows a similar trend in Location A, its training and testing accuracies are very low for Location B. The CART model achieves better results than SVM and LR. However, the difference between training and testing CWAs of CART is high, meaning that the model is not generalized for unseen data. Optimizing the Axis-aligned (CART) tree with TAO [11] solves this problem, but the Oblique tree [39] optimized with TAO algorithm [11] achieves better CWA than axis-aligned trees. TAO-Oblique trees achieve lesser training accuracy than CART in Location B, but the testing CWA is higher than CART. This means that

Table III: Cost-weighted accuracies of different machine learning models and optimizations.

ML Models & Optimizations	Location A				Location B			
	Training CWA (in %)	Testing CWA (in %)	Depth	No. of Leaves	Training CWA (in %)	Testing CWA (in %)	Depth	No. of Leaves
SVM	92.48 ± 0.33	92.66 ± 0.66	-	-	80.12 ± 0.54	82.74 ± 0.80	-	-
LR	92.64 ± 0.35	92.78 ± 0.44	-	-	80.94 ± 0.52	80.40 ± 1.37	-	-
CART	***97.68 ± 0.42***	93.56 ± 1.11	8.0 ± 0.6	31.6 ± 5.7	***92.25 ± 2.27***	85.88 ± 1.49	8.8 ± 4.1	33.8 ± 24.1
TAO-Axis Aligned	96.36 ± 1.09	94.67 ± 1.68	7.2 ± 1.3	25.2 ± 4.8	91.61 ± 1.92	86.14 ± 1.57	8.0 ± 3.8	24.2 ± 23.5
TAO-Oblique	97.11 ± 0.59	***95.47 ± 1.06***	***6.8 ± 1.2***	***18.8 ± 5.8***	89.90 ± 1.56	***87.87 ± 1.50***	***5.2 ± 2.2***	***13.0 ± 7.1***

Table IV: Decision rules of the sparse tree shown in Fig. 4.

Tree Node IDs	Weights for linear combination of the features				
	HN	**RSSI**	**PRR**	**RNP**	**Regularization Constant**
Node 0	0.958733267484	1.025309937395	-0.074761701442	0.077471341337	-0.122153674469
Node 1	1.415681867042	0	0	0	0.143560158843
Node 4	0	-1.298779855192	0	-0.211013927858	-1.448720963148

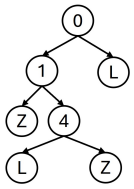


Figure 4: Sparse oblique tree

the TAO [11] algorithm optimizes the decision tree and makes it suitable for real-world deployments. On average, the TAO [11] algorithm can reduce the space complexity of the trees by 50%. TAO [11] algorithm can optimize the decision trees trained with lesser training data to perform better in the real world while reducing the space complexity of the trees to be deployed. This makes the TAO [11] algorithm more suitable for IoT deployments requiring less data collection effort. So, we use TAO-oblique trees for our experiments.

VI. Understanding radio selection via TAO's structural stability property

TAO-optimized sparse-oblique trees enable the extraction of meaningful insights from the interpretations by reducing the number of features used at the decision nodes [39]. Different combinations of all the feature vectors are tried non-homogeneously at each decision node. A tree providing better insights with reasonable accuracy is utilized for interpretations. The structure of the TAO-optimized sparse oblique tree is shown in Figure 4 and the corresponding weights are shown in Table IV.

At Node 0, higher emphasis is given to the Hop number and RSSI to divide the dataset into two parts. If the data sample has a higher hop number, i.e. farther away from the gateway, with the best RSSI, the tree always chooses LoRa.

At Node 1, we have the data samples that are not farther away from the gateway with better RSSI. This decision node emphasizes only on the Hop number. If the hop number is lower, i.e., the nodes are closer to the gateway. The tree always chooses the Zigbee radio to transmit as it provides higher throughput at closer distances.

At Node 4, we have the data samples that are in between farther and closer nodes. At this distance, higher emphasis is given to the RSSI of the LoRa radio and RNP [35] of the Zigbee radio. This means that RNP and RSSI are the two important features required to identify the high-throughput radio in the gray-region. Note that PDR is not as useful as RNP for the multi-hop Zigbee radios. We infer that this may be the case due to the RNP metric providing a quantitative measurement of the underlying distribution of packet losses instead of just an average number like PDR, leading to a better estimate of expected throughput.

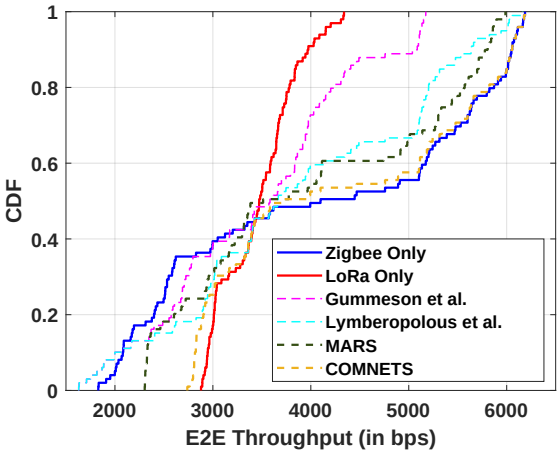
VII. Large-scale evaluations

COMNETS is evaluated on a real-world, large-scale mesh topology at two different locations as shown in Figures 2a and 2b. The deployments were done in complex environments that consisted of different building materials, human influx, and fleeting reflectors. The end devices will schedule a packet every three seconds periodically, destined to the gateway. When the packet is scheduled, the TAO[11]-oblique tree-based radio selector will select a radio for transmission. 10,400 data packets were transmitted for these experiments.

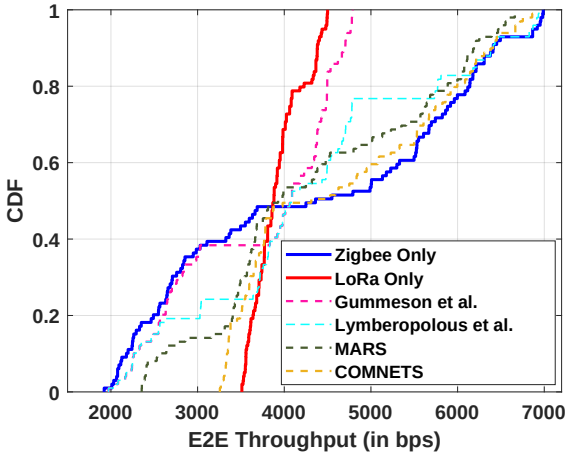
Benchmarks. The MARS [6] multi-radio system already outperforms Gummeson et al. [30] and the threshold-based [26] multi-radio system. However, we compare COMNETS with LoRa and Zigbee single radio systems and all the above-mentioned multi-radio systems.

Metrics. We evaluated the performance of COMNETS at two different locations on the following metrics: (i) Throughput, (ii) Latency, and performance ratio on different packet generation intervals.

Results. *The throughput gain* of COMNETS is plotted in Figure 5a and 5b for both the locations A and B. At Location A (Fig. 5a), Zigbee achieves high throughput for 42% of the transmissions and LoRa achieves higher throughput for the rest of the transmissions. MARS follows the high throughput radio all the time with slight errors for the first 20% of the transmissions and for the transmission between 50-



(a) Throughput - Location A



(b) Throughput - Location B

Figure 5: Throughput gain of COMNETS

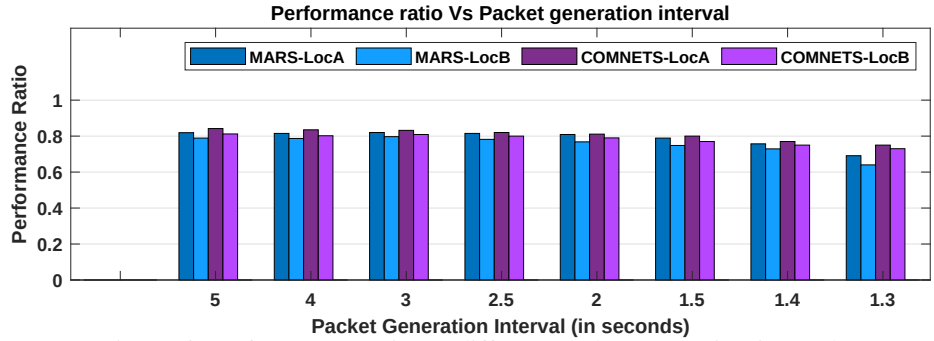


Figure 6: Performance Ratio on different packet generation intervals

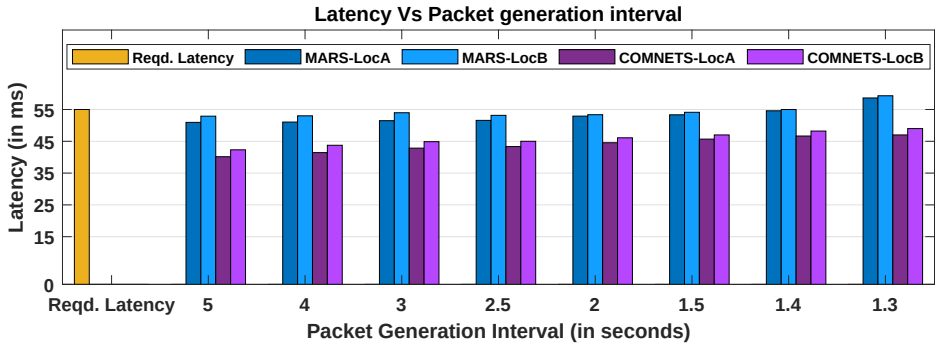


Figure 7: Latency on different packet generation intervals

55% of the transmissions. This is happening because MARS does not discriminate low and high throughput errors. At both locations, MARS outperforms Gummeson et al. [30] and Lymberopolous et al. [26]. At Location B, Gummeson et al. [30] and Lymberopolous et al. [26] tend to cross the solid red line. Theoretically, this is impossible as the multi-radio systems cannot cross the performance of the single radio system. However, in practice, these systems were evaluated at different times, when the channel quality (uncontrollable factor) would be different. This leads to a small jitter in throughput, making the performance go beyond the single-radio system. COMNETS optimizing to avoid high-throughput errors achieves an average throughput gain of 20.83% and 17.39% over MARS in Locations A and B respectively.

Performance Ratio of COMNETS on different packet generation intervals is plotted in Figure 6. The performance ratio is calculated as the average per-packet throughput of COMNETS over the average per-packet throughput of the best radio. In this figure, we can see that the performance ratio of MARS slightly decreases with a decrease in packet generation interval from 5-1.5s. A significant decrease in the performance of MARS happens for the packet generation intervals 1.4s and 1.3s. This significant decrease is due to the queuing delays caused by the increase in data packets. This queuing delay also affects the beacon packets that are used for path quality estimation and the packets that carry

this information to other nodes to calculate path quality. We follow the same path quality estimation as done in MARS [6]. The high-cost errors happening due to this phenomenon cause this significant dip in performance ratio. COMNETS follows similar trends with a decrease in the packet generation interval. However, COMNETS powered by TAO[11]-oblique trees, optimized to avert high-cost errors, achieves a higher performance ratio than MARS as it can avert most of the high-throughput errors. This cost-sensitive optimization helps to improve the overall performance of multi-radio networks even with inevitable detriments.

Latency of COMNETS on different packet generation intervals is plotted in Figure 7. According to 5G America's report [40], the average required latency for mesoscale IoT applications is 55ms. The latency of MARS slightly increases with a decrease in packet generation interval. The latency of MARS goes beyond the required 55ms when the packet generation interval is 1.3s. This is because MARS does not optimize for high-cost errors. However, COMNETS powered by TAO[11]-Oblique trees optimized to avert high-cost errors can keep the latency of the system within the required bounds.

VIII. Conclusion

We presented COMNETS, a cost-sensitive ML-based radio selection algorithm to predict high throughput radio for mesoscale IoT applications. We showed that MARS suffers from the problem of cost sensitivity, giving a greater margin for throughput optimization. The current cost-sensitive ML algorithms set a misclassification cost for each class while the radio selection problem for multi-radio networks entails different misclassification costs for each data sample. Optimizing such cost-sensitive costs is tedious. First, we leverage the TAO algorithm to overcome this issue. TAO not only optimizes the cost for each data sample, it also optimizes the tree for unseen data along with size reduction, making it efficient for deployment in resource-constrained IoT devices. Second, we leverage the structural stability property of TAO to understand the essence of the radio selection problem. By using TAO-optimized sparse oblique trees, we gain a more fundamental understanding of the importance of the different input features, and how COMNETS makes the classification decisions at different regions in the network. Having a tree structure allows for easier interpretability than other ML techniques (e.g. neural networks). Finally, our evaluations on large-scale, real-world deployments show that COMNETS achieves a throughput gain of 20.83% and 17.39% than MARS at two locations A and B, respectively.

References

[1] S. et al., "A survey on lora networking: Research problems, current solutions, and open issues," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 371–388, 2019.

[2] V. et al., "{FarmBeats}: an {IoT} platform for {Data-Driven} agriculture," in *14th USENIX NSDI 17*, 2017.

[3] F. et al., "Comprehensive optimal network scheduling strategies for wireless control systems," *ACM TCPS*, 2024.

[4] M. Electric, ""BLEnDer®"," http://tinyurl.com/3k435ymu, 2024.

[5] ——, "Smart meter system," http://tinyurl.com/4z2dy6nc, 2024.

[6] S. et al., "Mars: Multi-radio architecture with radio selection using decision trees for emerging mesoscale cps/iot applications," *arXiv preprint arXiv:2409.18043*, 2024.

[7] N. et al., "Improve ieee 802.15. 4 network reliability by suspendable csma/ca," in *2024 IEEE WCNC*, 2024, pp. 1–6.

[8] Y. et al., ": Rateless-enabled link adaptation for lora networking," *IEEE/ACM TON*, 2024.

[9] T. et al., "Resilience bounds of sensing-based network clock synchronization," in *IEEE ICPADS*, 2018.

[10] M. Electric, "Large-capacity battery control system," http://tinyurl.com/2ryvbcxv, 2024.

[11] C.-P. et al., "Alternating optimization of decision trees, with application to learning sparse oblique trees," *In NeurIPS '18*, 2018.

[12] J. et al., "A simple methodology for soft cost-sensitive classification," in *ACM SIGKDD*, 2012, pp. 141–149.

[13] L. et al., "Advances in cost-sensitive multiclass and multilabel classification," in *ACM SIGKDD*, 2019.

[14] D. et al., "Metacost: A general method for making classifiers cost-sensitive," in *ACM SIGKDD*, 1999.

[15] E. et al., "The foundations of cost-sensitive learning," in *IJCAI*, 2001.

[16] K. et al., "Classification cost: An empirical comparison among traditional classifier, cost-sensitive classifier, and metacost," *Expert Systems with Applications*, vol. 39, no. 4, pp. 4013–4019, 2012.

[17] G. et al., "Modeling churn using customer lifetime value," *European journal of operational research*, vol. 197, no. 1, pp. 402–411, 2009.

[18] Z. et al., "Cost-sensitive learning by cost-proportionate example weighting," in *IEEE ICDM*, 2003.

[19] T. et al., "Bias and the quantification of stability," *Machine Learning*, vol. 20, pp. 23–33, 1995.

[20] L. et al., "Instability of decision tree classification algorithms," in *ACM SIGKDD*, 2002.

[21] B. et al., "Reconsidering wireless systems with multiple radios," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 5, pp. 39–46, 2004.

[22] A. et al., "Blue-fi: enhancing wi-fi performance using bluetooth signals," in *ACM MobiSys*, 2009.

[23] S. et al., "Wifi-assisted 60 ghz wireless networks," in *ACM MobiCom, 2017*.

[24] J. et al., "Wizi-cloud: Application-transparent dual zigbee-wifi radios for low power internet access," in *IEEE INFOCOM, 2011*.

[25] K. et al., "Radio diversity for reliable communication in sensor networks," *ACM TOSN*, vol. 10, no. 2, pp. 1–29, 2014.

[26] L. et al., "Towards energy efficient design of multi-radio platforms for wireless sensor networks," in *ACM/IEEE IPSN*, 2008.

[27] D. et al., "Routing in multi-radio, multi-hop wireless mesh networks," in *ACM MobiCom*, 2004.

[28] A. A. Al Islam, M. S. Hossain, V. Raghunathan, and Y. C. Hu, "Backpacking: Deployment of heterogeneous radios in high data rate sensor networks," in *ICCCN*. IEEE, 2011.

[29] G. et al., "One-hop out-of-band control planes for multi-hop wireless sensor networks," *ACM TOSN*, vol. 15, no. 4, pp. 1–29, 2019.

[30] ——, "An adaptive link layer for range diversity in multi-radio mobile sensor networks," in *IEEE INFOCOM*, 2009.

[31] e. a. Fan, "Liblinear: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, p. 1871–1874, Jun. 2008.

[32] C. Technology, "TelosB," http://tinyurl.com/2svnbhjs, 2021.

[33] R. LLC, "LoStik," https://github.com/ronoth/lostik.

[34] C. et al., "A loop-free extended bellman-ford routing protocol without bouncing effect," *ACM SIGCOMM Computer Communication Review*, vol. 19, no. 4, pp. 224–236, 1989.

[35] ——, "Temporal properties of low power wireless links: modeling and implications on multi-hop routing," in *ACM MobiHoc*, 2005.

[36] C. Cortes, "Support-vector networks," *Machine Learning*, 1995.

[37] J. Friedman, "The elements of statistical learning: Data mining, inference, and prediction," *(No Title)*, 2009.

[38] P. et al., "Classification and regression trees: Leo breiman, jerome h. friedman, richard a. olshen and charles j. stone the wadsworth statistics/probability series, wadsworth, belmont, 1984, x+ 358 pages," 1985.

[39] H. et al., "Sparse oblique decision trees: A tool to understand and manipulate neural net features," *Data Mining and Knowledge Discovery*, vol. 38, no. 5, pp. 2863–2902, 2024.

[40] G. Americas. (2019) 5G - The future of IoT. [Online]. Available: http://tinyurl.com/26y5epsp