Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 20 No. 21 (2026) | OPEN ACCESS

https://doi.org/10.3991/ijim.v20i21.62621

PAPER

Machine Learning for LoRaWAN Smart Parking: A Multi-Horizon Occupancy Prediction Benchmark

Avni Rexhepi, Vullnet Laniku (✉), Mihrije Kadriu, Nefise Alimusaj, Bujar Krasniqi

University of Prishtina, Prishtina, Kosovo

vullnet.laniku@uni-pr.edu

ABSTRACT

Smart parking systems guide drivers to free spaces through mobile interfaces whose usefulness depends on the accuracy and lead time of the underlying occupancy forecasts. We benchmark seven models—random forest (RF), XGBoost, LightGBM, a feed-forward neural network (NN), an LSTM, a temporal convolutional network (TCN), and a transformer—on a full year of real data: 112,830 observations from five LoRaWAN-instrumented spaces at the University of Prishtina, Kosovo (June 2024–June 2025). Departing from single-horizon nowcasting, we predict at 15-minute, 1-hour, and 2-hour lead times for both binary occupancy classification and continuous occupancy-ratio regression, under a leakage-free chronological split, with deep models trained to convergence and every result averaged over ten seeds with significance testing. The value of machine learning proves strongly horizon-dependent: because occupancy is highly autocorrelated, a zero-cost persistence baseline matches the best learned model up to one hour ahead, and only at two hours do six of the seven models beat it significantly (best long short-term memory (LSTM) 79.82% vs. 76.32%, $p < 10^{-10}$). For regression the sequence models lead at every horizon ($R^2 = 0.686$ at one hour) and the gradient-boosted trees outrank RF. An ablation shows historical (autocorrelation) features supply almost all of the signal, corroborated by SHAP and permutation importance. A resource analysis is counterintuitive: RF trains fastest yet yields the largest model (165 MB), while the compact sequence models (~0.2 MB) are far more deployable. These results give realistic, persistence-aware guidance for smart-parking services.

KEYWORDS

smart parking, occupancy forecasting, multi-horizon prediction, machine learning, LoRaWAN

1 INTRODUCTION

Parking management is a key element of smart city infrastructure, as drivers often spend considerable time and fuel searching for available spaces. This inefficiency not only increases traffic congestion but also contributes to higher emissions and urban pollution [1]. Smart parking systems have emerged as an effective solution, using IoT sensors and machine learning to provide real-time and predictive information on parking

Rexhepi, A., Laniku, V., Kadriu, M., Alimusaj, N., Krasniqi, B. (2026). Machine Learning for LoRaWAN Smart Parking: A Multi-Horizon Occupancy Prediction Benchmark. *International Journal of Interactive Mobile Technologies (iJIM)*, 20(21), pp. 150–172. https://doi.org/10.3991/ijim.v20i21.62621

Article submitted 2026-06-11. Revision uploaded 2026-07-25. Final acceptance 2026-08-31.

© 2026 by the authors of this article. Published under CC-BY.

availability [2], [3]. Through distributed sensors and cloud-based platforms, these systems enable proactive guidance for drivers and improve overall resource utilization.

Awan et al. [4] compare several ML/DL models for parking availability prediction and report that tree-based methods and KNN often outperform deeper architectures on real parking datasets. Traditional ML models, such as random forest (RF), provide reliability and computational efficiency, particularly when applied to structured datasets. In contrast, DL models such as long short-term memory (LSTM) networks are well suited for capturing temporal dependencies and complex usage patterns. Recent advancements, including hybrid architectures (e.g., GRU–LSTM stacks) and attention-based mechanisms, have further enhanced predictive accuracy [5], [6].

Recent work enriches occupancy prediction with spatio-temporal and geospatial features and IoT data streams [7]–[9] and connects it to energy-efficient LPWAN (notably LoRaWAN) and edge–cloud execution for city-scale deployments [10]–[12].

In this work, we report a real, year-long LoRaWAN smart-parking deployment at the University of Prishtina and use the data it produced to study a question of direct practical importance for driver-facing mobile services: *how far in advance can parking availability be predicted reliably, and which models are worth their computational cost?* Real-time availability is ultimately consumed through mobile and web interfaces that guide drivers to free spaces [13], [14]; the usefulness of such an interactive service depends entirely on the accuracy and the lead time of the forecasts that drive it. While prior studies have addressed individual aspects such as distributed learning [15] or statistical modeling of parking patterns [16], few provide a unified, leakage-aware evaluation of classification and regression models over a long observation period, and the trade-offs between predictive accuracy and resource constraints (latency, model size, energy) remain insufficiently explored.

We address these gaps through a strict chronological evaluation on a full year of real data. This paper makes four contributions: (1) a deployed LoRaWAN-to-cloud parking system that provides both the dataset and the operational context; (2) a leakage-free, *multi-horizon* benchmark of seven models—RF, XGBoost, LightGBM, a feed-forward neural network (NN), an LSTM, a Temporal Convolutional Network (TCN), and a Transformer—on both binary occupancy classification and continuous occupancy-ratio regression at 15-minute, 1-hour, and 2-hour lead times, complemented by a feature-group ablation, SHAP and permutation explainability, and a spatial neighbor-occupancy analysis; (3) a deliberately strong *persistence* baseline that exposes an effect underemphasized in the literature: short-horizon occupancy prediction is nearly trivial, and learning-based models earn a clear advantage only as the forecast horizon grows; and (4) a resource-aware comparison (training time, model size, and inference latency) that yields concrete, and sometimes counter-intuitive, model-selection guidance for memory- and energy-constrained deployments. The remainder of this paper is organized as follows: Section 2 reviews related work and positions our contribution; Section 3 describes the deployed LoRaWAN system; Section 4 details the data pipeline, features, models, and experimental protocol; Section 5 presents the multi-horizon results, ablation, explainability, spatial analysis, resource costs, and deployment workflow; and Section 6 concludes.

2 RELATED WORK

2.1 IoT and LPWAN smart parking

Surveys report that occupancy prediction can cut driver search time by 20–30% [3], [17], and integrated IoT–ML frameworks and mobile front-ends have been

demonstrated for city deployments [8], [13]. Among connectivity options, low-power wide-area networks (particularly LoRaWAN) offer the range and battery life needed for parking sensors, outperforming short-range Wi-Fi and Zigbee [18]. Real LoRaWAN parking deployments, sensor designs, coverage models, and energy-autonomous nodes have all been studied [10], [12], [19]–[22]; security analyses also note side-channel risks in such sensors [23]. Complete-system and edge/cloud designs, including anomaly detection for large-scale sensing, provide further architectural context [11], [14], [24]–[26].

2.2 Machine learning for occupancy prediction

Beyond classical statistical models, comparative studies find that tree-based methods often match or beat deeper architectures on real parking data [4], while LSTM and other sequence models exploit temporal patterns [27], [28]. More elaborate deep models (stacked GRU–LSTM, attention, and graph networks) together with rich spatio-temporal feature fusion further improve accuracy [5]–[7], [29], and machine learning has also been applied to optimize LoRaWAN itself [30].

2.3 Research gaps

Despite this progress, few studies offer a unified, leakage-aware comparison of multiple models under identical conditions; accuracy is rarely weighed against computational cost (model size, training time, latency); datasets often span only weeks to months, missing seasonal effects; and LPWAN communication is seldom integrated with ML analytics in one cohesive, deployed system. We address these gaps with a year-long real LoRaWAN deployment and a leakage-free, multi-horizon, resource-aware benchmark of seven models on both classification and regression.

Table 1. Positioning against representative LoRaWAN and ML smart-parking studies

Study	Link	Span	Task	Models	Pers.	Cost
Santana [10]	LoRa	weeks	nowcast	stat.	no	no
Jabbar [12]	LoRa	demo	detect	rules	no	no
Pous-S. [20]	LoRa	case	nowcast	thresh.	no	part.
Awan [4]	sensor	months	multi	RF+KNN	no	no
Zeng [5]	cam.	months	1-horiz	GRU+LSTM	no	no
Ye [6]	city	months	1-horiz	attn.	no	no
Zhang [7]	multi	months	1-horiz	ST-DL	no	no
Qu [15]	public	small	1-horiz	meta	no	no
This work	**LoRa**	**1 yr**	**clf+reg**	**7+GNN**	**yes**	**yes**

Notes: Link: connectivity; Span: dataset duration; Task: prediction target (this work spans 15 min–2 h horizons); Pers.: whether a persistence/naive baseline is reported; Cost: whether model size, latency, or energy is analyzed.

Table 1 positions this study against representative work. To our knowledge, no prior LoRaWAN smart-parking study combines (i) a full year of real deployment data,

(ii) a leakage-free *multi-horizon* evaluation of both classification and regression, (iii) a strong persistence baseline that quantifies *when* learning actually adds value, and (iv) a resource-cost analysis for constrained deployment. Prior LoRaWAN parking papers focus on connectivity, sensing, or system integration and report little or no predictive modelling, whereas the broader ML-forecasting literature rarely uses LPWAN data, seldom reports a persistence baseline, and typically evaluates a single horizon on weeks-to-months of data. Our contribution is therefore an honest, reproducible *benchmark* rather than a new architecture.

3 SYSTEM MODEL

The system was deployed at the Faculty of Electrical and Computer Engineering (FIEK), University of Prishtina, Kosovo, where five outdoor parking spaces were instrumented with LoRa-based occupancy sensors. The site experiences a continental climate (–10 to +35 °C) and serves university staff, students, and visitors, producing pronounced weekday peaks and quiet weekends, a realistic test of an institutional parking facility. The architecture follows a standard four-layer IoT design [24]: a *sensing* layer (LoRa occupancy sensors), a *communication* layer (LoRaWAN gateway), a *processing* layer (cloud server with storage, analytics, and machine learning), and an *application* layer (real-time dashboards and predictive services).

3.1 LoRa parking sensors

Each space is monitored by a battery-powered sensor that uses Chirp Spread Spectrum modulation for long-range, low-power communication. Occupancy detection combines a magnetometer (sensing magnetic-field changes) with a radar sensor (sensing surface reflections) [31]; this dual mechanism provides reliable detection, and the node transmits only on state changes to conserve energy. The compact, IP68-rated enclosure (see Figure 1) is built for outdoor use, and per the vendor datasheet, the sensor offers multi-year battery life with a configurable sampling interval [31].



Fig. 1. LoRa sensor

3.2 LoRaWAN gateway

The gateway bridges the sensors to the cloud over LoRaWAN. It uses a high-gain antenna and a rugged IP67 enclosure suited to dense outdoor deployments, with vendor-rated long-range urban coverage, support for many nodes per unit, and automatic failover to a backup gateway for resilience [32].

3.3 Cloud server and real-time monitoring

The cloud server runs the open-source ChirpStack (v4.8.1) network server: a Gateway Bridge translates gateway UDP traffic into MQTT [33], and the network server handles device sessions and persistence (PostgreSQL storage, Redis cache) over AES-128 encrypted transport. Its web dashboard (usable from desktop and mobile browsers) is the operator-facing, interactive monitoring interface (see Figure 2), showing device and gateway liveness, data-rate usage, and a gateway map.

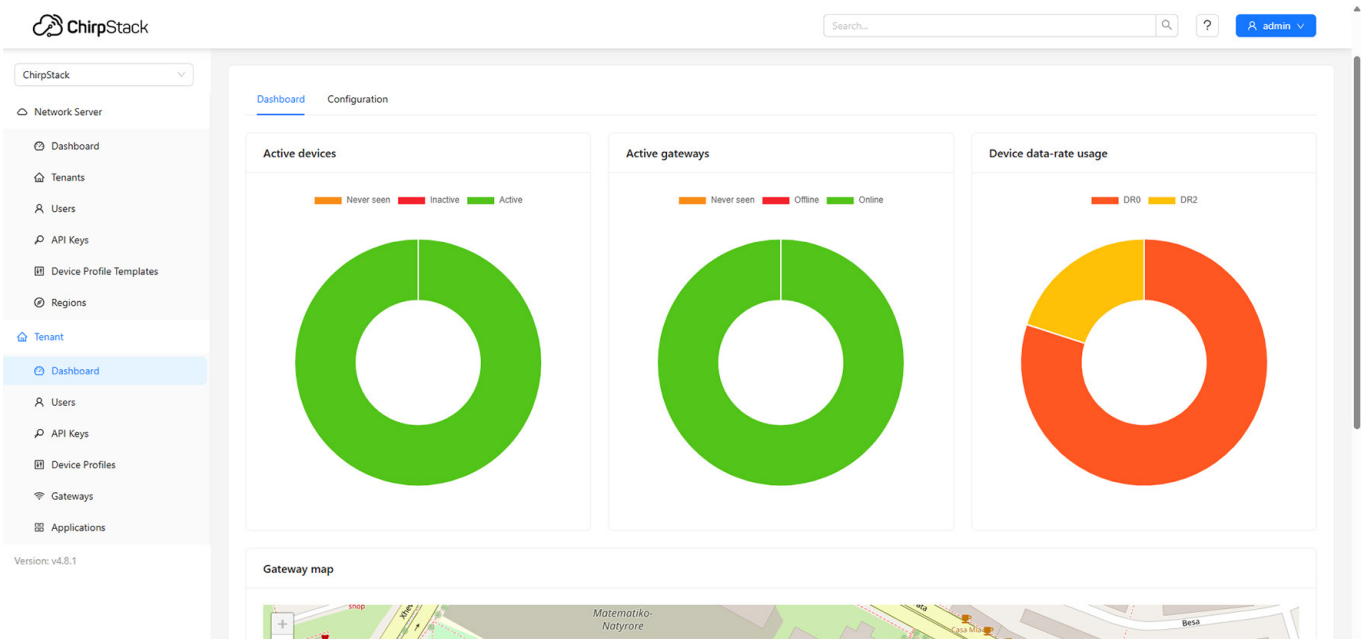


Fig. 2. ChirpStack network-server dashboard: device and gateway liveness, data-rate usage (DR0/DR2), and the gateway map

Throughout the year-long operation all five nodes remained *active* and the gateway *online*, indicating stable connectivity. The end-to-end pipeline (sensor uplink, gateway transmission, MQTT brokering, and cloud processing) is summarized in Figure 4, and raw events are aggregated into 15-minute bins for modeling. The vendor figures cited above (range, battery life) are datasheet specifications; to complement them with real link measurements, we ran a dedicated network-characterization campaign on the same deployment, reported next.

3.4 Measured network performance

Because the year-long production logs retained only occupancy state and not per-uplink radio metadata, we exported a dedicated link-characterization dataset from the network server over an 83-day campaign (April 21–July 13, 2026) on the same five-node, single-gateway deployment. The export contains 701 uplinks (501 state-change and 200 heartbeat frames) with per-frame RSSI, SNR, spreading factor, frame counter, and battery voltage. Table 2 and Figure 3 summarize the results; we report them as *measured*, distinct from the datasheet values and from the occupancy-forecasting dataset (a different, earlier period).

Three findings stand out. Link quality is adequate but not generous: RSSI averages –102.6 dBm and SNR +4.9 dB, about 20 dB above the SF7 sensitivity floor. Correcting an assumption in our earlier reporting, adaptive data-rate control did *not* settle on the most robust rate—uplinks split between SF7 (46%) and SF10 (50%), consistent with borderline links. Using frame-counter continuity, the apparent per-node delivery ratio is 59.7–80.9% (68.5% overall); this counts every counter gap as a loss and is thus a conservative lower bound (some gaps are server-side de-duplication), but indicates non-negligible loss on the more marginal nodes. Battery voltage stayed within 2.63–3.16 V. Event-driven sensing tolerates such loss well: occupancy is slowly varying, and each state is re-transmitted, so a dropped uplink does not corrupt the 15-minute series used for forecasting.

Table 2. Measured LoRaWAN link performance per node over the 83-day campaign (frame-counter-based delivery ratio, mean RSSI/SNR)

Node	Uplinks	Deliv. (%)	RSSI (dBm)	SNR (dB)
N1	102	79.1	–99.7	5.7
N2	106	63.9	–101.1	5.0
N3	127	80.9	–103.6	4.6
N4	176	59.7	–104.1	4.9
N5	190	68.6	–102.8	4.6
Overall	*701*	*68.5*	*–102.6*	*4.9*

Note: Nodes N1–N5 are the same five sensors, labelled independently for this later campaign.

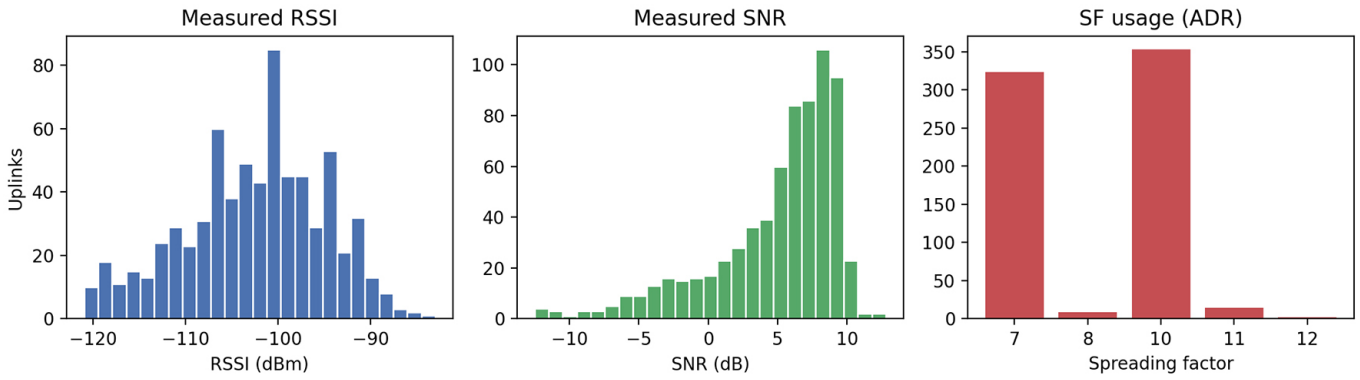


Fig. 3. Measured link statistics over the 83-day campaign: RSSI and SNR distributions and adaptive spreading-factor usage (predominantly SF7/SF10)

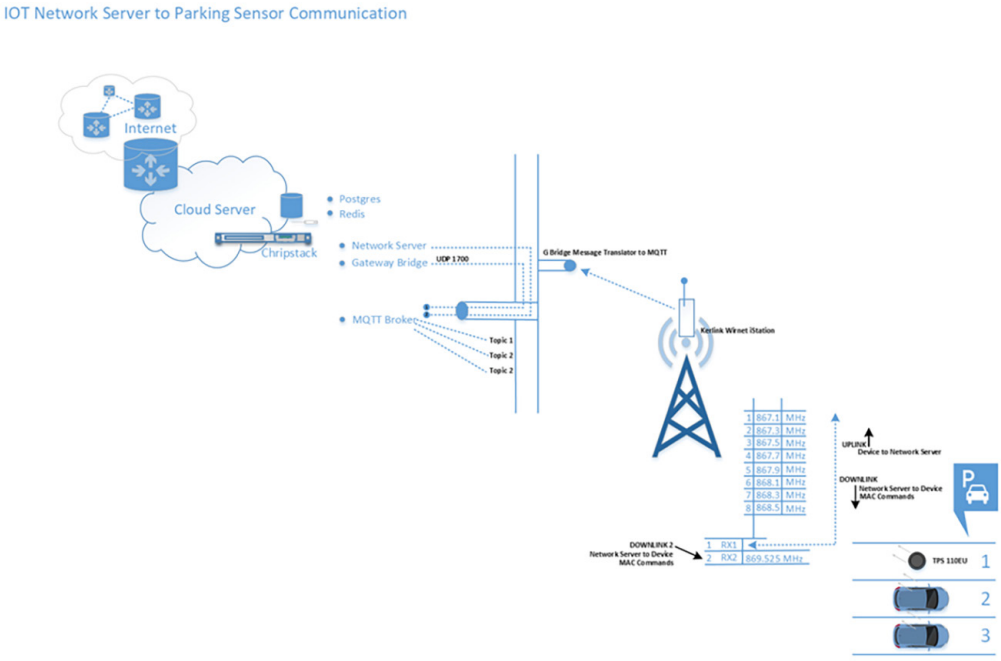


Fig. 4. System architecture

4 MACHINE LEARNING METHODOLOGY

The proposed framework employs a structured machine learning pipeline to predict parking occupancy, integrating data preprocessing, feature engineering, and model training.

4.1 Dataset description

We collected LoRaWAN smart-parking occupancy data continuously from the same deployment over a full year (June 15, 2024–June 14, 2025). No synthetic data were generated; all records were retrieved from the deployment backend logs. The dataset consists of 112,830 occupancy observations from 5 distinct parking spaces (labelled S1 through S5), with each space contributing approximately 22,500 records (about 62 per day). Sensors report event-driven occupancy changes; for modeling, the event stream is sorted chronologically per space and treated as a sequence of fixed 15-minute observations during operational hours (06:00–21:45). Each record contains a parking-space identifier, a 15-minute-resolution timestamp, and the binary occupancy state (0: vacant, 1: occupied).

The overall occupancy rate is 35.70%. Occupancy varies only modestly across seasons (winter 36.2%, spring 35.2%, summer 34.6%, fall 36.9%; monthly means range 31.9%–40.8%), but exhibits a pronounced weekday–weekend contrast typical of an academic facility: 42.06% on weekdays versus 19.99% on weekends. A key statistical property of the data, central to the analysis that follows, is its strong temporal autocorrelation: because vehicles remain parked for extended periods, an occupancy reading is identical to the one 15 minutes earlier in roughly 96% of cases. This makes a simple *persistence* predictor a deceptively strong baseline at short horizons and motivates our multi-horizon evaluation.

The five spaces are *not* homogeneous (refer to Table 3). Mean occupancy ranges from 29.8% to 39.8%, and the weekday–weekend gap varies markedly (e.g., S4 drops from 45.1% on weekdays to 17.7% on weekends, whereas S3 stays comparatively high at 27.2%). What the spaces *share* is the academic usage rhythm—a strong weekday peak in the late morning (10:00–11:00)—so the dataset combines a common temporal pattern with genuine per-space level differences. This heterogeneity is why the location features (Section 5.6) contribute, and it tempers over-generalization: the findings characterize one institutional lot, not arbitrary parking contexts (Section 5.12).

Table 3. Per-space usage statistics (full year)

Space	Overall	Weekday	Weekend	Peak Hour
S1	29.8	34.5	18.0	10
S2	39.1	46.2	21.7	10
S3	39.8	44.8	27.2	10
S4	37.2	45.1	17.7	10
S5	32.6	39.6	15.4	11

Notes: Occupancy rates in %; peak hour is the busiest hour of the day.

Forecasting tasks. We frame occupancy prediction as a *forecasting* problem evaluated at several lead times, rather than as nowcasting of the current state (which, given the autocorrelation above, is trivial). Let $x_t \in \{0,1\}$ denote occupancy at 15-minute bin t for a given space. We define two task families:

- **Classification:** predict the binary state x_{t+h} at horizon $h \in \{1,4,8\}$ bins, i.e., 15 minutes, 1 hour, and 2 hours ahead.
- **Regression:** predict the next-window occupancy ratio

$$y_t^{(h)} = \frac{1}{h}\sum_{k=1}^{h} x_{t+k}, \quad y_t^{(h)} \in [0,1],$$

for $h \in \{4, 8\}$ (the fraction of the next 1 or 2 hours the space is occupied). This yields calibrated occupancy scores rather than hard labels.

Evaluating multiple horizons lets us quantify *how far ahead* learning-based models provide value over the persistence baseline, which is the central question of this study.

4.2 From raw events to a regular time series

The sensors are event-driven: a node transmits only when it detects a state change, so the raw stream is irregularly sampled. We convert it to a regular series in four deterministic steps. *(1) Binning*. For each space we sort events chronologically and assign the prevailing occupancy state to fixed 15-minute bins over operational hours (06:00–21:45); the 15-minute resolution matches the typical minimum dwell time and keeps the series short enough for real-time inference. *(2) Forward-fill within gaps*. Because a node is silent while a state persists, a bin with no new event inherits the

last known state—the physically correct assumption for an event-driven occupancy sensor—applied only *within* an operational day and never across the overnight gap. *(3) Gap handling*. Bins in non-operational hours are excluded rather than imputed, and days with no transmissions from a node (e.g., maintenance) are left as genuine gaps so that no synthetic transitions are created. *(4) Anomaly screening*. Occupancy is binary by construction, so out-of-range values are impossible; we checked for and found no duplicate timestamps or physically implausible sub-bin oscillations beyond the sensor's debounced output. No records were synthetically generated or resampled beyond this binning.

4.3 Feature engineering

The pipeline transforms the raw timestamped stream into structured inputs while preventing leakage: *all predictive features are strictly causal*, computed at bin t from information available only up to bin $t - 1$, so none can reveal the predicted value. The tree-based and feed-forward models share 15 tabular features, while the LSTM consumes the raw sequence (Section 4.4). *Temporal* features capture cyclical patterns: hour, day of week, day of month, month, a weekend indicator, and four operational periods (Morning/Afternoon/Evening/Night); hour and day-of-week are additionally sine/cosine-encoded for the neural models. *Historical (causal)* features capture autocorrelation: the previous state x_{t-1}, time since the last reading, and one-step-lagged rolling summaries of past occupancy (3-hour and 24-hour means and a 24-hour variability term).

Location-specific features account for differences across spaces: a label-encoded space identifier and three historical occupancy rates (per space, per hour-of-day, and per day-of-week), all estimated *exclusively from the training period* and then mapped onto both partitions so that test predictions never benefit from future statistics.

4.4 Models and training

We evaluate *seven* models spanning three families, each applied to *both* the classification and regression tasks at every horizon. Four *feature-based* models operate on the shared 15-dimensional tabular feature set: a RF, two gradient-boosted tree ensembles (XGBoost and LightGBM), and a feed-forward NN. Three *sequence* models learn representations directly from the raw occupancy sequence with no hand-crafted features: a LSTM network, a TCN, and a Transformer encoder. To ensure a fair comparison, the feature-based models receive identical inputs, the three sequence models share the same input window, and all deep models are trained to convergence rather than for a fixed small budget (a limitation of our earlier pilot study).

Random Forest. Implemented with scikit-learn using 300 trees, a maximum depth of 15, a minimum of 5 samples to split and 2 per leaf. The same configuration is used for classification (`RandomForestClassifier`) and regression (`RandomForestRegressor`).

Gradient-boosted trees. XGBoost and LightGBM, each with 400 estimators, learning rate 0.05, row and column subsampling of 0.8; XGBoost uses a maximum depth of 6 and LightGBM uses 31 leaves. These state-of-the-art tabular learners are widely reported to be strong baselines on structured data and are included at the reviewers' suggestion.

Neural Network. A multilayer perceptron (scikit-learn `MLP`) with three hidden layers of sizes (128,64,32) for classification and (256,128,64) for regression, with ReLU activations, Adam, L_2 penalty $\alpha = 10^{-4}$, initial learning rate 10^{-3}, and mini-batches of 512. Training uses *early stopping* (10% internal validation split, patience of 15 iterations, up to 400 iterations), so the network trains until validation performance plateaus rather than for an arbitrary fixed number of epochs.

The three sequence models share the same input: the previous 12 bins (3 hours) of [*x*, *sin/cos* hour, *sin/cos* day, weekend], with inputs standardized and training sequences stride-2 subsampled for efficiency. All are trained with Adam (10^{-3}), binary cross-entropy (classification) or mean-squared error (regression), and early stopping on a held-out validation slice (patience 4, up to 30 epochs); a sigmoid bounds classification outputs to [0,1].

LSTM. A two-layer LSTM (hidden size 64, dropout 0.2) implemented in PyTorch, followed by a small feed-forward head ($64 \rightarrow 16 \rightarrow 1$).

TCN. A TCN of three causal dilated 1-D convolution blocks (48 channels, kernel size 3, dilations 1,2,4, ReLU, dropout 0.2) whose exponentially growing receptive field spans the input window, with a linear head on the last step. The TCN offers a multi-scale temporal view without recurrence.

Transformer. A two-layer Transformer encoder (model dimension 64, four attention heads, feed-forward dimension 128, dropout 0.2) with a learned positional embedding, followed by a linear head on the final position. Its self-attention provides the adaptive, content-dependent weighting of past bins that a fixed recurrent state cannot.

Scope of the model set. Beyond the seven models above, we additionally evaluated a spatiotemporal *Graph Neural Network* (a learnable-adjacency graph-convolution layer over the five spaces followed by a temporal recurrent unit) to test directly whether modeling inter-space structure helps; its result is reported in Section 5.7. We did *not* include Temporal Graph Networks or the Temporal Fusion Transformer: these are designed to exploit a rich spatial adjacency structure across many sensors, whereas with only five co-located spaces in a single small lot, the graph is essentially fully connected and degenerate. Fitting such high-capacity forecasters on five nodes would invite over-parameterization without a credible spatial prior; we treat them as future work once the deployment spans many geographically distributed lots.

4.5 Experimental protocol

Protocol. To respect causality, we use a single chronological holdout rather than random or *k*-fold splitting: the first ten months (through April 14, 2025) form the training set, and the final two months from the test set (93,948 training and 18,882 test instances for classification, comparable for regression), with no shuffling across the boundary and all leakage-prone statistics computed on training data only. Deep models are trained with early stopping (convergence verified from the loss curves, see Figure 7); to account for training stochasticity, every model is run with *ten* independent random seeds, and we report the mean ± standard deviation. Significance is assessed as follows: because the persistence and historical-average baselines are deterministic (a single value each), a model is compared against a baseline with a *one-sample t*-test of its ten-seed metric (accuracy or R^2) against that constant; two learned models are compared with a *paired t*-test across their shared seeds. All tests use $n = 10$, and we report exact *p*-values for the model-versus-persistence comparison at each horizon. The full pipeline was implemented in Python (scikit-learn and PyTorch) and is provided to reviewers via an anonymous repository (see Data Availability).

Baselines and metrics. We compare against a *Naive* majority-class predictor, a *Persistence* predictor ($\hat{x}_{t+h} = x_{t-1}$), and, for regression, an *Hourly Historical Average* (the mean next-window ratio for the matching space, hour, and day-of-week). Classification is scored by accuracy, precision, recall, F_1, and AUC-ROC; regression by R^2, MAE, and RMSE; and computational cost by training time, serialized model size, and per-sample inference latency.

5 RESULTS AND DISCUSSION

All models were trained on the first ten months of data and evaluated on the held-out final two months (April 14–June 14, 2025, i.e., a late-spring/early-summer window). Each headline metric is reported as the mean ± standard deviation over ten random seeds (the ablation and spatial analyses use five); baselines are the Naive majority-class predictor, the Persistence predictor ($\hat{x}_{t+h} = x_{t-1}$), and, for regression, the Hourly Historical Average.

5.1 Forecast accuracy across horizons

Table 4 and Figure 5 report classification accuracy at the three horizons for all seven models. The dominant feature of the results is the strength of the persistence baseline at short range. At 15 minutes the three sequence models match persistence exactly (93.47%; differences not significant, $p \geq 0.19$) while even the tree ensembles are within 0.2 points, confirming that nowcasting occupancy is essentially trivial. At one hour, persistence (85.28%) remains extremely competitive: it is *not* beaten by any model—the best learners (LSTM and TCN, both 85.32%) are statistically indistinguishable from it ($p = 0.32$ and 0.12), while RF, XGBoost, and LightGBM fall slightly but significantly below. Only at the two-hour horizon do the learned models open a clear, statistically significant gap over persistence: six of the seven models exceed persistence (76.32%)—all but the plain NN, which trails at 70.72%—led by the LSTM (79.82%) and TCN (79.78%), each with $p < 10^{-10}$. The gradient-boosted trees, though strong, do not surpass RF on this binary task.

Table 4. Classification accuracy (%) by forecast horizon (mean ± std over 10 seeds)

Model	15 min	One-Hour	Two-Hour
RF	93.41 ± .02	85.17 ± .06	79.27 ± .10
XGBoost	93.30 ± .06	84.22 ± .13	77.98 ± .12
LightGBM	93.37 ± .04	84.56 ± .15	78.82 ± .12
Neural Net	93.31 ± .24	78.06 ± 1.1	70.72 ± .80
LSTM	93.47 ± .01^{ns}	**85.32 ± .12**^{ns}	**79.82 ± .23**
TCN	**93.47 ± .00**^{ns}	85.32 ± .06^{ns}	79.78 ± .14
Transformer	93.47 ± .00^{ns}	85.30 ± .11^{ns}	79.62 ± .29
Persistence	*93.47*	*85.28*	*76.32*
Naive	*65.85*	*65.83*	*65.80*

Notes: Best model per column in bold; ^{ns} marks models statistically indistinguishable from persistence ($p > 0.05$).

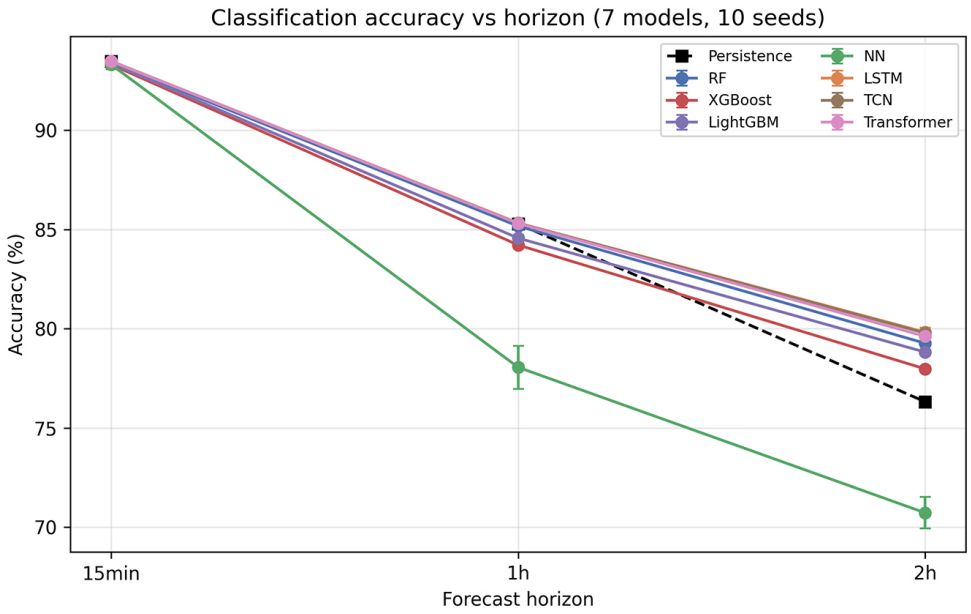


Fig. 5. Classification accuracy versus forecast horizon for all seven models

Notes: The learned models match a simple persistence baseline up to one hour ahead and pull clear of it only at two hours; error bars show ±1 std over 10 seeds.

5.2 Occupancy-ratio forecasting (regression)

The continuous next-window occupancy ratio is where learning clearly pays off. Table 5 and Figure 6 show that every learned model except the plain NN far exceeds the baselines at both horizons ($p < 10^{-19}$ against the historical average). The three sequence models are the strongest (1 h: LSTM and TCN both $R^2 = 0.686$; 2 h: TCN 0.612, LSTM 0.611), and—unlike the classification task—the gradient-boosted trees now clearly outrank the RF, with LightGBM (0.671/0.587) > XGBoost (0.664/0.577) > RF (0.658/0.567). The margin of the learned models over persistence *widens* with horizon, from $\Delta R^2 \approx 0.11$ at one hour to ≈ 0.24 at two hours, mirroring the classification trend and reinforcing the central message that the value of learning grows with lead time. The feed-forward NN is markedly weaker (and essentially degenerate at two hours, $R^2 \approx 0$); operating on the same tabular features as the tree ensembles, this gap is consistent with the well-documented difficulty of plain multilayer perceptrons on tabular data.

Table 5. Regression R^2 by forecast horizon (mean ± std over 10 seeds)

Model	One-Hour	Two-Hour
RF	0.658 ± .001	0.567 ± .001
XGBoost	0.664 ± .002	0.577 ± .002
LightGBM	0.671 ± .002	0.587 ± .003
Neural Net	0.186 ± .019	0.011 ± .059
LSTM	**0.686 ± .003**	0.611 ± .003
TCN	0.686 ± .001	**0.612 ± .002**
Transformer	0.684 ± .002	0.609 ± .003
Persistence	*0.577*	*0.374*
Hourly-hist.	*0.241*	*0.256*

Note: Best model per column in bold.

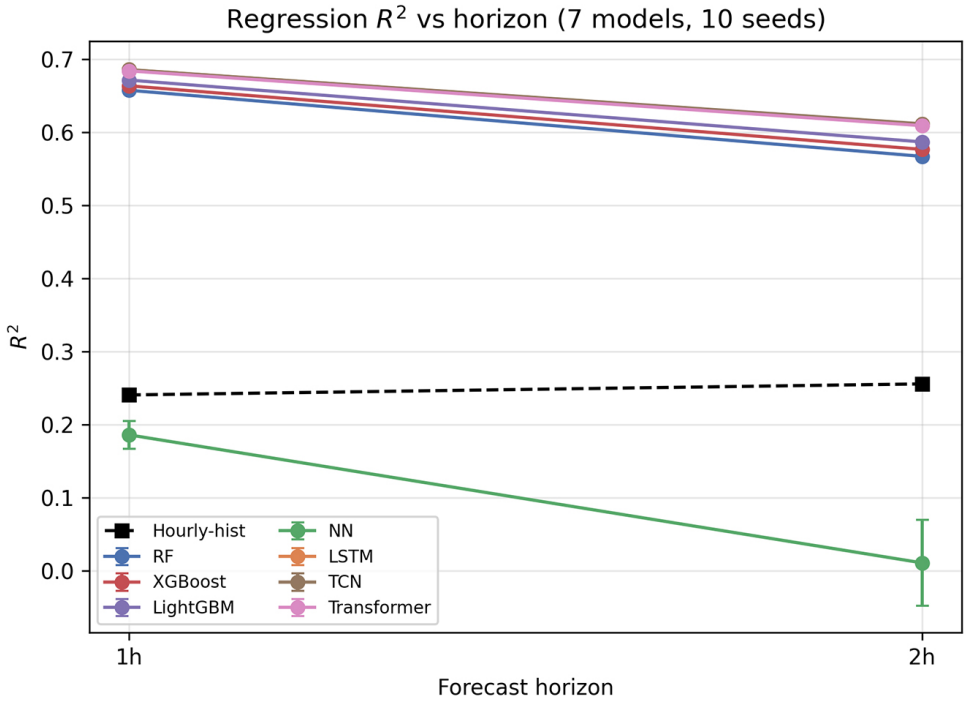


Fig. 6. Regression R^2 versus forecast horizon for all seven models

Notes: Sequence models are best; the learned models' advantage over the baselines grows with lead time. Error bars show ±1 std over ten seeds.

5.3 Detailed metrics at the one-hour horizon

Tables 6 and 7 give the full metric set for all seven models at the operationally important one-hour horizon. For classification, the three sequence models are the strongest and near-identical (AUC-ROC 0.917), with RF close behind (0.911) and the NN clearly lagging (0.837). For regression, the sequence models again lead on every error metric, with the Transformer attaining the lowest MAE (0.150) and TCN the lowest RMSE (0.253). Figure 7 confirms that the deep models converge well before the early-stopping limit, validating the fair-comparison protocol.

Table 6. Detailed classification metrics at the 1-hour horizon (mean over 10 seeds)

Model	Acc.	Prec.	Rec.	F_1	AUC
RF	85.17	**0.774**	0.799	0.786	0.911
XGBoost	84.22	0.762	0.782	0.772	0.908
LightGBM	84.56	0.764	0.794	0.778	0.912
Neural Net	78.05	0.676	0.687	0.681	0.837
LSTM	**85.32**	0.771	0.812	0.791	**0.917**
TCN	85.32	0.768	**0.817**	**0.792**	0.917
Transformer	85.30	0.770	0.812	0.791	0.917

Notes: Accuracy in %. Best per column in bold.

Table 7. Detailed regression metrics at the 1-hour horizon (mean over 10 seeds)

Model	R^2	MAE	RMSE
RF	0.658	0.166	0.265
XGBoost	0.664	0.169	0.262
LightGBM	0.671	0.165	0.259
Neural Net	0.186	0.270	0.408
LSTM	**0.686**	0.159	0.254
TCN	0.686	0.154	**0.253**
Transformer	0.684	**0.150**	0.254

Note: Best per column in bold.

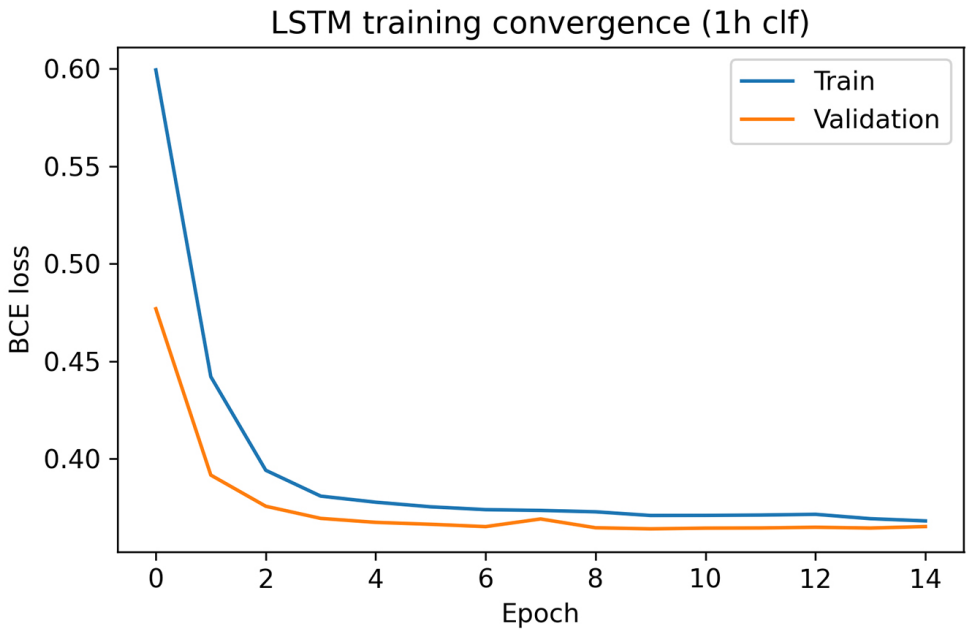


Fig. 7. LSTM training and validation loss (1-hour classification)

Notes: Training converges and is halted by early stopping, ensuring the deep models are not under-trained relative to RF.

5.4 Comparison with prior reported results

Direct numerical comparison with prior work is complicated by differing datasets and horizons, but a quantitative anchoring is instructive (refer to Table 8). On the Santander on-street dataset, Awan et al. [4] report (5-fold CV) 92.3% (Decision Tree) and 86.5% (RF) at 10 minutes, falling to 87.7%/82.5% at 20 minutes. Our accuracies sit in the same regime (93.5% at 15 minutes, 85.3% at one hour). The decisive difference is methodological: like most of the parking-prediction literature [5], [6], [27], [28], that study reports *no persistence baseline*—yet at these short horizons a zero-cost persistence rule already attains ≈ 93%, so the high headline accuracies commonly reported largely reflect occupancy autocorrelation rather than model skill.

Our contribution is therefore not a marginally higher number but a corrected interpretation: significant gains over the trivial baseline emerge only at two hours and in the continuous-ratio task—an effect prior numbers, lacking a persistence reference, cannot expose.

Table 8. Quantitative anchoring against a representative comparative study

Study/Setting	Horizon	Best Acc.	Persist.?
Awan *et al.* [4] (Santander)	10 min	92.3%	no
Awan *et al.* [4] (Santander)	20 min	87.7%	no
This work (university LoRaWAN)	15 min	93.5%	**yes (93.5%)**
This work	1 h	85.3%	**yes (85.3%)**
This work	2 h	79.8%	**yes (76.3%)**

Notes: Accuracies is not directly comparable (different datasets) but illustrate the regime; the key column is whether a persistence baseline is reported.

5.5 Feature importance and explainability

To avoid relying on any single attribution, we characterize the one-hour RF classifier with three methods: impurity importance, permutation importance (accuracy drop when a feature is shuffled on the test set), and SHAP (TreeExplainer). All three agree (refer to Table 9, see Figure 8): recent occupancy overwhelmingly dominates. The previous state accounts for 36.3% of impurity and a 21.6-point permutation accuracy drop—an order of magnitude above any other feature—which is why persistence is so strong. The 3-hour rolling average, hour-of-day, and the leakage-free hourly historical rate provide the incremental signal that lets models pull ahead at longer horizons. Permutation and SHAP are more concentrated than impurity (which spreads credit across correlated lags), the expected difference that motivates reporting all three.

Table 9. RF feature attribution at the 1-hour horizon by three methods (top features)

Feature	Impurity	Permut.	SHAP
Previous Occupancy State	36.25	21.56	0.191
3h Rolling Occupancy	14.38	1.22	0.060
Hour of Day	8.86	0.33	0.066
Hourly Historical Rate	7.77	0.16	0.038
Day-of-Week Hist. Rate	4.02	0.14	0.023
Day of Month	4.86	0.06	0.010
24h Rolling Occupancy	4.60	0.02	0.009

Notes: Impurity and permutation are in %; SHAP is mean $|\text{value}|$.

Rankings agree on the dominance of the recent state; the remaining features are small under all three methods.

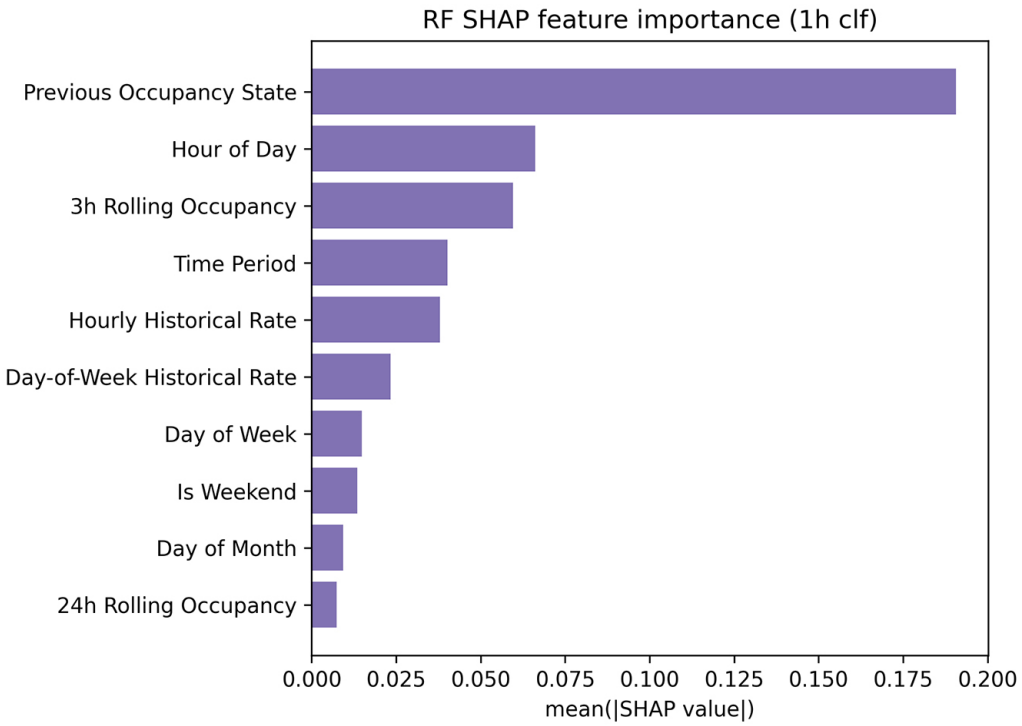


Fig. 8. SHAP feature importance (mean absolute SHAP value) for the 1-hour RF classifier, confirming that recent occupancy state dominates, followed by hour-of-day and short-range rolling occupancy

5.6 Feature-group ablation

To isolate *which* feature groups drive performance, we retrain the RF at the one-hour horizon while cumulatively adding groups (refer to Table 10). The result is unambiguous: calendar features alone reach only 74.3% accuracy and $R^2 = 0.19$ (worse than persistence), whereas adding the historical group jumps accuracy to 84.9% and R^2 to 0.66—the autocorrelation features supply essentially all of the predictive signal. Location features add +0.3 points and the neighbour feature +0.05 (next). This confirms the models exploit genuine occupancy dynamics rather than calendar coincidences.

Table 10. Feature-group ablation for the RF at the 1-hour horizon (mean over 5 seeds), adding groups cumulatively

Feature Groups	Accuracy (%)	Regression R^2
Temporal only	74.29	0.194
+ Historical	84.87	0.660
+ Location (all 15)	85.15	0.658
+ Neighbour occupancy	85.20	0.660

5.7 Spatial (neighboring-space) interaction

Because the five spaces share one small lot, occupancy in neighboring spaces could carry a predictive signal. We added a strictly causal *neighbor-occupancy* feature—the one-bin-lagged mean occupancy of the *other* four spaces—and re-evaluated the RF with and without it (refer to Table 11). The effect is small but consistent

and, notably, grows with horizon: +0.05 points at one hour and +0.11 at two hours. The modest size is expected: co-located spaces are driven by a common lot-level demand signal that each space's own recent history already largely encodes, so the marginal information in neighbors is limited. The horizon-growing trend nonetheless suggests neighbor context becomes more useful precisely where single-space autocorrelation weakens—motivating richer spatial models once the deployment spans multiple, geographically separated lots.

To test the strongest form of spatial modelling directly, we also evaluated a spatiotemporal graph neural network (GNN): a graph-convolution layer with a *learnable* 5×5 adjacency over the co-registered nodes, followed by a shared temporal LSTM and a per-node head, trained on a regular 15-minute grid (five seeds). The GNN reaches 85.8% ± 0.1 at one hour and 80.2% ± 0.1 at two hours—statistically on par with the best single-space sequence models (refer to Table 4) and following the same horizon pattern (tying grid-persistence at one hour, 85.7%, and beating it at two, 76.8%). Explicit, learnable access to all five spaces thus yields *no material improvement* over per-space history: as the ablation and small neighbor effect already indicated, the co-located spaces share a common lot-level signal that per-space temporal features capture, so graph capacity is not the bottleneck. This supports treating graph architectures as future work for geographically distributed, many-lot deployments.

Table 11. Effect of adding the neighbor-occupancy feature to the RF classifier (accuracy %, mean over 5 seeds)

Horizon	Without	With Neighbor	Gain
1 h	85.15	85.20	+0.05
2 h	79.19	79.30	+0.11

5.8 Random-forest configuration robustness

To confirm the results are not an artifact of a single lucky hyperparameter choice, we swept the number of trees {100,300,500} and maximum depth {8,15,25} for the one-hour classifier (refer to Table 12). Accuracy stays within a narrow 84.2–85.5% band across all nine configurations. Depth matters more than tree count: shallow trees (depth 8) are actually best (85.3–85.5%) and the deepest trees (depth 25) mildly overfit (≈ 84.3%), while beyond 100 trees the count is nearly irrelevant. The configuration reported throughout (300 trees, depth 15) is representative rather than cherry-picked.

Sequence-model input length. We similarly verified that the sequence models are insensitive to the input-window length: sweeping the window over 6, 12, and 24 bins (1.5, 3, and 6 hours) at the one-hour horizon changes accuracy by less than 0.2 points (LSTM 85.2–85.3%, TCN 85.2–85.4%), with the 12-bin (3-hour) window used throughout at or near the optimum. The fixed window is therefore not a limiting design choice, and a longer receptive field yields no benefit—consistent with the dominance of very recent occupancy shown by the importance analysis.

Table 12. RF 1-hour accuracy (%) across tree count and maximum depth (single seed)

Trees	Depth 8	Depth 15	Depth 25
100	85.34	85.07	84.21
300	**85.45**	85.09	84.29
500	85.42	85.10	84.31

5.9 Error analysis

Examining the one-hour RF classifier on the test window, accuracy is markedly higher on weekends (91.6%) than weekdays (82.6%), reflecting the more predictable (consistently low) weekend occupancy of an academic facility. Within operational hours, errors concentrate in the morning and afternoon (error rate ≈18%), when arrivals and departures drive most state transitions, and are lowest in the evening (≈5%), when spaces are reliably vacant. Across the spring and summer months spanned by the test set, accuracy is stable (85.2% and 84.8%). These patterns indicate that residual error is concentrated at genuine transition events rather than distributed randomly.

5.10 Computational cost and model selection

Table 13 reports computational cost for all seven models. The results overturn a common assumption. RF trains fast (2.5 s) but is by far the *largest* model—165 MB for 300 depth-15 trees on a year of data—two orders of magnitude bigger than every other model. The gradient-boosted ensembles dominate the cost–accuracy frontier for the tabular models: LightGBM and XGBoost train fastest (<1.1 s), occupy only ~1.4–1.8 MB, and (for regression) are more accurate than RF. The sequence models are the smallest of all—the TCN is just 0.06 MB—but the slowest to train (37–91 s), and the Transformer has the highest inference latency. Inference is sub-millisecond per space for every model.

Table 13. Computational cost (1-hour classification, single seed)

Model	Train (s)	Size (MB)	Infer (ms/samp.)
RF	2.5	165	0.005
XGBoost	1.1	1.83	**0.001**
LightGBM	**0.8**	1.38	0.001
Neural Net	82.9	0.31	0.001
LSTM	36.3	0.22	0.010
TCN	68.4	**0.06**	0.007
Transformer	90.8	0.28	0.024

Note: Best per column in bold.

Combining accuracy, robustness, and cost yields clear guidance. For a short-horizon binary state (≤1 h), no machine learning is warranted: persistence matches the best-learned model at zero cost. For longer horizons (≥2 h), the learned models give a real, statistically significant gain, and among them the gradient-boosted trees are attractive server-side—sub-second retraining and a ~1.5 MB artifact—while the sequence models give the best regression accuracy in a footprint (~0.06–0.28 MB) small enough for an edge gateway. The RF, despite its training speed, is the largest artifact to ship (165 MB) and needs pruning for constrained nodes. Two broader lessons follow for mobile smart-parking services: a deployed predictor's headline accuracy must always be read against persistence, and the lead time a guidance app promises should be chosen deliberately, since useful value over the trivial baseline emerges only around the two-hour horizon and in the continuous-ratio formulation.

5.11 Deployment and mobile-application workflow

The forecasting models are intended to run *server-side* rather than on the battery-powered sensor. In the deployed pipeline, uplinks reach the ChirpStack network server, are binned into 15-minute observations, and a scheduled job refreshes forecasts; because RF retrains in seconds and all models infer in well under a millisecond per space (see Table 13), a lot of this size can be re-scored continuously at negligible cost. A driver-facing mobile or web client then queries a lightweight availability API. The choice of horizon is an application decision: a 15-minute or 1-hour forecast is adequate for a driver already approaching the lot (for whom persistence is essentially as good), whereas a *two-hour* forecast is the operationally useful one for pre-trip planning—deciding whether to drive at all or whether to pre-book—which is precisely the regime where the learned models beat persistence by a statistically significant margin. The compact LSTM (~0.2 MB) can alternatively be shipped to an edge gateway for offline operation, whereas the 165 MB RF is impractical to embed and is better kept in the cloud.

5.12 Limitations

Several limitations apply. The evaluation covers five spaces in a single university lot; although these spaces are individually heterogeneous (refer to Table 3), they share one institutional demand pattern, so the results should be read as a rigorous benchmark on one real deployment rather than evidence of generalization to commercial, residential, or on-street parking, which remains untested. The chronological holdout, while sound, places the entire test set in a single late-spring/early-summer window (training spans the full year). The feature set omits external signals such as weather and academic-calendar events that could help at transition periods. On the networking side, we now report *measured* radio statistics (RSSI, SNR, spreading factor, frame-counter delivery ratio, battery voltage) from a dedicated 83-day campaign (Section 3.4), which supersedes the earlier datasheet-only reporting; however, this campaign is shorter than the year-long occupancy record and was collected in a later period, uses a single gateway, and does not include a direct end-to-end latency measurement, so range and latency remain datasheet values. Because the radio campaign and the forecasting window do not overlap, we also do not yet fuse communication-aware features (e.g., SF/RSSI) into the predictor, which we leave as future work. Models were evaluated offline, so live effects (communication latency, sensor faults, and concept drift) are not captured. Nonetheless, the year-long training data and the consistent, statistically tested findings across seven models and multiple horizons support our conclusions about *when* learning-based forecasting adds value.

6 CONCLUSION

We presented a multi-horizon, resource-aware, and leakage-free benchmark of parking-occupancy forecasting, grounded in a real year-long LoRaWAN deployment. Using 112,830 observations from five instrumented spaces over June 2024–June 2025, we evaluated seven models—RF, XGBoost, LightGBM, a feed-forward NN, an LSTM, a TCN, and a Transformer (each deep model trained to convergence and every model repeated over ten seeds)—on binary occupancy classification (15-minute, 1-hour, and 2-hour horizons) and continuous occupancy-ratio regression (1- and 2-hour horizons), against naive, persistence, and historical-average baselines,

complemented by a feature-group ablation, SHAP and permutation explainability, a spatial neighbor-occupancy analysis, and a hyperparameter-robustness study.

Our central finding is that the value of machine learning for this problem is *horizon-dependent*. Because parking occupancy is highly autocorrelated, a zero-cost persistence rule is statistically as accurate as the best learned model up to one hour ahead; only at the two-hour horizon do six of the seven models (all but the plain NN) beat persistence by a clear and statistically significant margin (best: LSTM 79.82% vs. 76.32% accuracy, $p < 10^{-10}$). For the continuous occupancy-ratio task, the sequence models are best at every horizon ($R^2 = 0.686$ at one hour, 0.612 at two), the gradient-boosted trees outrank the RF, and the advantage of the learned models over the baselines widens with lead time. A feature-group ablation shows that historical (autocorrelation) features supply almost all of the predictive signal. The plain NN is consistently the weakest, underscoring that architecture choice matters more than a simple "deep vs. classical" dichotomy.

On the resource side, our measurements correct a common assumption: although RF trains fastest and infers with the lowest latency, on a full year of data it produces by far the *largest* model (165 MB), whereas the gradient-boosted trees (~1.5 MB) and the sequence models (~0.06–0.28 MB) are far more deployable and, for regression, at least as accurate. For memory-constrained mobile and edge deployments a compact model is therefore the better engineering choice, and short-horizon services may need no learned model at all.

Future work. We plan to (i) expand the deployment to additional lots and on-street spaces to test cross-context generalization; (ii) incorporate weather and academic-calendar features expected to help at transition periods; (iii) extend the 83-day measured network campaign to a full year aligned with the occupancy record, add end-to-end latency instrumentation, and fuse the resulting communication-aware features (SF, RSSI, delivery ratio) into the predictor; and (iv) deploy a compact forecasting model on-device to drive a low-latency, driver-facing mobile availability service. We hope this honest, persistence-aware methodology helps other universities and municipalities set realistic expectations when adopting IoT-based parking analytics.

7 USE OF AI TOOLS

The authors used a generative AI assistant (a large language model) to support implementation of the analysis pipeline (code refactoring and consolidation), language editing, and preparation of result figures and tables. The study design, experimental protocol, and all reported results were specified and verified by the authors, who reviewed every output and take full responsibility for the content of this paper.

8 DATA AVAILABILITY

The dataset and the complete source code used in this study are publicly available at https://github.com/vlaniku/lorawan-smart-parking-benchmark.

9 ACKNOWLEDGMENT

The authors would like to acknowledge the support of the Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ) GmbH, Kosovo, under Grant Agreement No. 81298267, for the implementation of the IoT-based smart parking system in front of the FIEK, University of Prishtina.

10 REFERENCES

[1] M. R. Alam, S. Saha, M. B. Bostami, M. S. Islam, M. S. Aadeeb, and A. K. M. M. Islam, "A survey on IoT driven smart parking management system: Approaches, limitations and future research agenda," *IEEE Access*, vol. 11, pp. 119523–119543, 2023. https://doi.org/10.1109/ACCESS.2023.3327306

[2] S. S. Channamallu, S. Kermanshachi, J. M. Rosenberger, and A. Pamidimukkala, "Parking occupancy prediction and analysis – a comprehensive study," *Transportation Research Procedia*, vol. 73, pp. 297–304, 2023. https://doi.org/10.1016/j.trpro.2023.11.921

[3] X. Xiao, Z. Peng, Y. Lin, and G. Mao, "Parking prediction in smart cities: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 10, pp. 10302–10326, 2023. https://doi.org/10.1109/TITS.2023.3279024

[4] F. M. Awan, Y. Saleem, R. Minerva, and N. Crespi, "A comparative analysis of machine/deep learning models for parking space availability prediction," *Sensors*, vol. 20, no. 1, Art. no. 322, 2020. https://doi.org/10.3390/s20010322

[5] C. Zeng *et al.*, "Parking occupancy prediction method based on multi factors and stacked GRU-LSTM," *IEEE Access*, vol. 10, pp. 47361–47370, 2022. https://doi.org/10.1109/ACCESS.2022.3171330

[6] W. Ye, H. Kuang, J. Li, X. Lai, and H. Qu, "A parking occupancy prediction method incorporating time series decomposition and temporal pattern attention mechanism," *IET Intelligent Transport Systems*, vol. 18, no. 1, pp. 58–71, 2024. https://doi.org/10.1049/itr2.12433

[7] L. Zhang, B. Wang, Q. Zhang, S. Zhu, and Y. Ma, "Parking lot traffic prediction based on fusion of multifaceted spatio-temporal features," *Sensors*, vol. 24, no. 15, Art. no. 4971, 2024. https://doi.org/10.3390/s24154971

[8] M. Alymani, L. A. Almoqhem, D. A. Alabdulwahab, A. A. Alghamdi, H. Alshahrani, and K. Raza, "Enabling smart parking for smart cities using Internet of Things (IoT) and machine learning," *PeerJ Computer Science*, vol. 11, Art. no. e2544, 2025. https://doi.org/10.7717/peerj-cs.2544

[9] M. Balmer, R. Weibel, and H. Huang, "Value of incorporating geospatial information into the prediction of on-street parking occupancy – A case study," *Geo-Spatial Information Science*, vol. 24, no. 3, pp. 438–457, 2021. https://doi.org/10.1080/10095020.2021.1937337

[10] J. R. Santana, P. Sotres, J. Pérez, L. Sánchez, J. Lanza, and L. Muñoz, "LoRaWAN-based smart parking service: Deployment and performance evaluation," in *Proc. 19th ACM Int. Symp. Perform. Eval. Wireless Ad Hoc, Sens. Ubiquitous Netw. (PE-WASUN '22)*, Montreal, QC, Canada, 2022, pp. 107–114. https://doi.org/10.1145/3551663.3558685

[11] P. Raith, S. Nastic, and S. Dustdar, "Serverless edge computing—Where we are and what lies ahead," *IEEE Internet Computing*, vol. 27, no. 3, pp. 50–64, 2023. https://doi.org/10.1109/MIC.2023.3260939

[12] W. A. Jabbar, L. Y. Tiew, and N. Y. A. Shah, "Internet of things enabled parking management system using long range wide area network for smart city," *Internet of Things and Cyber-Physical Systems*, vol. 4, pp. 82–98, 2024. https://doi.org/10.1016/j.iotcps.2023.09.001

[13] A. Saeliw, W. Hualkasin, S. Puttinaovarat, and K. Khaimook, "Smart car parking mobile application based on RFID and IoT," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 13, no. 5, pp. 4–14, 2019. https://doi.org/10.3991/ijim.v13i05.10096

[14] H. Th. S. Alrikabi and N. A. Jasim, "Design and implementation of smart city applications based on the internet of things," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 15, no. 13, pp. 4–15, 2021. https://doi.org/10.3991/ijim.v15i13.22331

[15] H. Qu, S. Liu, J. Li, Y. Zhou, and R. Liu, "Adaptation and learning to learn (ALL): An integrated approach for small-sample parking occupancy prediction," *Mathematics*, vol. 10, no. 12, Art. no. 2039, 2022. https://doi.org/10.3390/math10122039

[16] M. Schneble and G. Kauermann, "Statistical modelling of on-street parking spot occupancy in smart cities," *Journal of the Royal Statistical Society: Series C*, vol. 74, no. 4, pp. 1128–1149, 2025. https://doi.org/10.1093/jrsssc/qlaf017

[17] A. Raj and S. D. Shetty, "Smart parking systems technologies, tools, and challenges for implementing in a smart city environment: A survey based on IoT & ML perspective," *International Journal of Machine Learning and Cybernetics*, vol. 15, pp. 2673–2694, 2024. https://doi.org/10.1007/s13042-023-02056-5

[18] C. Bayilmis, M. A. Ebleme, Ü. Çavuşoğlu, K. Küçük, and A. Sevin, "A survey on communication protocols and performance evaluations for Internet of Things," *Digital Communications and Networks*, vol. 8, no. 6, pp. 1094–1104, 2022. https://doi.org/10.1016/j.dcan.2022.03.013

[19] J. R. Santana, P. Sotres, J. Pérez, L. Sánchez, J. Lanza, and L. Muñoz, "Assessing LoRaWAN radio propagation for smart parking service: An experimental study," *Computer Networks*, vol. 235, Art. no. 109962, 2023. https://doi.org/10.1016/j.comnet.2023.109962

[20] C. Pous-Sabadí, J. Meléndez, R. Casas, and J. T. Peinado, "Technology assessment for LoRaWAN-based time-limited smart parking: A case study," *IEEE Access*, vol. 12, pp. 158446–158470, 2024. https://doi.org/10.1109/ACCESS.2024.3484758

[21] B. P. R. Bhavanam and P. Ragam, "LoRa sense: Sensing and optimization of LoRa link behavior using path-loss models in open-cast mines," *CMES – Computer Modeling in Engineering and Sciences*, vol. 142, no. 1, pp. 425–466, 2024. https://doi.org/10.32604/cmes.2024.052355

[22] P. Šolić, A. Leoni, R. Colella, T. Perkovic, L. Catarinucci, and V. Stornelli, "IoT-ready energy-autonomous parking sensor device," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4830–4840, 2021. https://doi.org/10.1109/JIOT.2020.3031088

[23] L. D. Rodić, T. Perković, M. Škiljo, and P. Šolić, "Privacy leakage of LoRaWAN smart parking occupancy sensors," *Future Generation Computer Systems*, vol. 138, pp. 142–159, 2023. https://doi.org/10.1016/j.future.2022.08.007

[24] Z. Alavikia and M. Shabro, "A comprehensive layered approach for implementing internet of things-enabled smart grid: A survey," *Digital Communications and Networks*, vol. 8, no. 3, pp. 388–410, 2022. https://doi.org/10.1016/j.dcan.2022.01.002

[25] S. Husein, D. Mudhafar, and B. Saravana Balaji, "The impact of a collaborative IoT framework for smart cities and environmental monitoring," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 17, no. 11, pp. 68–82, 2023. https://doi.org/10.3991/ijim.v17i11.36277

[26] Z. Hasani, S. Krrabaj, and M. Krasniqi, "Proposed model for real-time anomaly detection in big IoT sensor data for smart city," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 18, no. 3, pp. 32–44, 2024. https://doi.org/10.3991/ijim.v18i03.44467

[27] G. Ali *et al.*, "IoT based smart parking system using deep long short memory network," *Electronics*, vol. 9, no. 10, Art. no. 1696, 2020. https://doi.org/10.3390/electronics9101696

[28] R. K. Kasera and T. Acharjee, "Parking slot occupancy prediction using LSTM," *Innovations in Systems and Software Engineering*, vol. 18, no. 3, pp. 313–321, 2022. https://doi.org/10.1007/s11334-022-00481-3

[29] W. Ye, H. Kuang, X. Lai, and J. Li, "A multi-view approach for regional parking occupancy prediction with attention mechanisms," *Mathematics*, vol. 11, no. 21, Art. no. 4510, 2023. https://doi.org/10.3390/math11214510

[30] A. Farhad *et al.*, "LoRaWAN meets ML: A survey on enhancing performance with machine learning," *Sensors*, vol. 23, no. 15, Art. no. 6851, 2023. https://doi.org/10.3390/s23156851

[31] Bosch Connected Devices and Solutions GmbH, "Parking sensor: Wireless sensors for detecting parking space occupancy," Datasheet, 2018. [Online]. Available: https://lora-alliance.org/wp-content/uploads/2019/07/parking-lot-sensor-datasheet.pdf [Accessed: Dec. 2024].

[32] Kerlink, "Wirnet iFemtoCell LoRaWAN gateway," Product Specification. [Online]. Available: https://www.kerlink.com/product/wirnet-ifemtocell/ [Accessed: Dec. 2024].
[33] ChirpStack, "MQTT integration," ChirpStack Documentation, Version 4. [Online]. Available: https://www.chirpstack.io/docs/chirpstack/integrations/mqtt.html [Accessed: Dec. 2024].

11 AUTHORS

Avni Rexhepi is a Professor at the University of Prishtina, Faculty of Electrical and Computer Engineering, and head of the "Computer and Software Engineering" department. His fields of interest encompass Programming, Data Structures and Algorithms, Machine Learning, and Artificial Intelligence (E-mail: avni.rexhepi@uni-pr.edu).

Vullnet Laniku is a Teaching Assistant and Network Engineer at the University of Prishtina "Hasan Prishtina," Faculty of Electrical and Computer Engineering, where he is currently pursuing the Ph.D. degree in Computer and Electrical Engineering. His research interests include 5G/6G systems, integrated sensing and communication, the Internet of Things, and machine learning for communications (E-mail: vullnet.laniku@uni-pr.edu).

Mihrije Kadriu is a Teaching Assistant at the University of Prishtina "Hasan Prishtina," Faculty of Electrical and Computer Engineering. Her research interests include IoT security, cloud computing, Artificial Intelligence, and machine learning, with publications in these areas. She has also contributed to various cybersecurity and IoT-security research projects (E-mail: mihrije.kadriu@uni-pr.edu).

Nefise Alimusaj is currently pursuing her Ph.D. at the University of Prishtina "Hasan Prishtina," Faculty of Electrical and Computer Engineering. She works as a Senior Network Engineer in the telecommunications industry. Her research interests include machine learning, artificial intelligence, semantic communication, and radio resource management (E-mail: nefise.alimusaj@student.uni-pr.edu).

Bujar Krasniqi is an Associate Professor at the Faculty of Electrical and Computer Engineering, University of Prishtina. He received his Ph.D. in Electrical Engineering and Information Technology from TU Wien, Austria, in 2011. His research interests include 5G, the Internet of Things, radio planning and optimization, and machine learning for wireless communication (E-mail: bujar.krasniqi@uni-pr.edu).