Tabular and Time-Series Position Encodings in 6G Network Data

Investigating the Effects on Beam-Prediction Performance and Representation Quality

Pelayo Fernández Luengo[1]

Supervisors: Rihan Hai[1]**, Yuandou Wang**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Pelayo Fernández Luengo
Final project course: CSE3000 Research Project
Thesis committee: Rihan Hai, Yuandou Wang, Julián Urbano Merino

Abstract

Sixth-generation (6G) networks collect positioning data that must be transformed into a suitable representation before machine-learning models can use it effectively. The choice of this encoding is rarely treated as an experimental variable, yet it strongly shapes what information reaches the downstream model. This paper evaluates how tabular and time-series encoding techniques affect beam prediction performance and feature representation quality in nine scenarios from the DeepSense 6G dataset. Beam-prediction performance is measured using two downstream classifiers in a fixed multi-seed evaluation pipeline, while representation quality is assessed through invariance under positional noise. Encodings that represent the user equipment relative to the base station and include temporal context achieve the best performance. However, the representation analysis reveals that these geometry-aware encodings are less stable under positional noise. The findings suggest that, when position estimates are accurate, position and trajectory data should be encoded using base-station-relative distance, bearing and recent geometric change, whereas noisier settings may require additional preprocessing to preserve robustness.

1 Introduction

Sixth-generation (6G) networks are expected to rely increasingly on data-driven methods for communication tasks such as beam prediction [1; 2]. These tasks can be addressed using machine-learning models trained on measurements collected in experimental 6G scenarios. DeepSense 6G provides such data across diverse real-world settings, including positioning information and communication measurements used for beam prediction [3]. Before positioning data can support beam prediction, it must be encoded into model inputs. The same GPS measurements can make different information available depending on their representation, yet this step is often treated as fixed preprocessing rather than an experimental choice.

This work examines how different representations expose spatial and temporal context that may be useful for beam prediction. Figure 1 illustrates the distinction between the tabular and time-series settings considered in this study.

A similar distinction appears in the AI/ML beam management taxonomy considered in 3GPP TR 38.843 [4; 5], where beam-prediction settings also differ in whether they rely on current information or historical context. This paper applies this distinction to position-based beam prediction, linking the 3GPP standardisation context to the encoding choices evaluated in this work.

Prior work shows that position and trajectory information can support beam prediction, especially when it captures the relation between the user equipment (UE) and the base station (BS) [6; 7; 8]. However, it remains unclear how different current-position and history-aware encodings compare when the dataset, task, split and downstream models are fixed. This motivates the main research question:

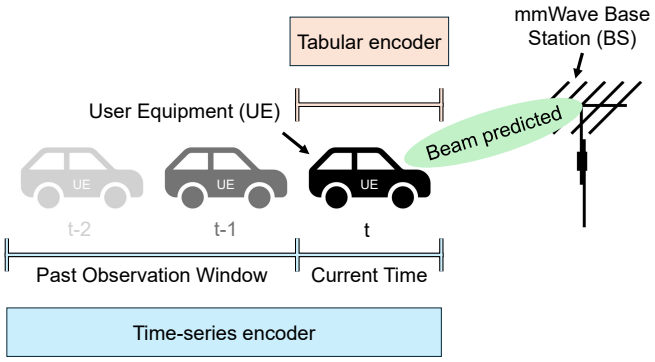

Figure 1: Comparison of tabular and time-series encoders in the Vehicle-to-Infrastructure (V2I) beam-prediction task. The tabular encoder uses only the current observation and excludes temporal information. The time-series encoder preserves the chronological order and may include the timestamp, with some configurations also incorporating previous observations to capture temporal context.

> *How do tabular and time-series position encoding techniques affect beam-prediction performance and feature representation quality in 6G network data?*

We investigate this question by comparing both encoding strategies for beam prediction using the DeepSense 6G dataset [3]. The evaluated configurations range from minimally processed position features to encodings that explicitly represent the geometry of the UE relative to the BS and encodings that additionally incorporate temporal context. We hold splits, models and seeds fixed so that observed outcome differences primarily reflect the encoding effect.

Across the evaluated encodings, the strongest beam-prediction results are obtained when the representation explicitly describes the geometry between the UE and the BS, especially when this geometry is combined with temporal context. However, these high-performing encodings are also less invariant to positional noise, revealing a trade-off between predictive usefulness and robustness.

This work makes three contributions. First, we provide a reproducible evaluation pipeline for position encodings. Second, we present a controlled comparison of tabular and time-series encodings across DeepSense 6G Scenarios 1-9 over multiple seeds. Third, we show that the choice of encoding affects both beam-prediction performance and the stability of the representation under positional noise.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 introduces the problem setting and encoding techniques. Section 4 describes the experimental pipeline. Section 5 details the experimental configuration. Section 6 reports the results. Section 7 interprets the findings and discusses threats to validity. Section 8 reflects on responsible research and Section 9 concludes and outlines future work.

2 Related Work

2.1 Encodings in 6G Data Context

Position information provides a low-overhead representation for beam prediction, and prior work differs mainly in how explicitly position is represented.

One approach is to use raw GPS coordinates directly, which has been shown to support beam prediction on real-world measurements [6]. However, this does not show whether raw coordinates are the most effective representation, because the model must still infer the UE-to-BS relation from latitude and longitude. Other approaches expose more geometric information per sample, for example by adding device orientation to the location [7] or by representing the UE position through its direction relative to the BS [8]. These works indicate that physical geometry is useful, but they do not isolate the encoding choice itself. This motivates the tabular encodings evaluated here, which range from raw position coordinates to BS-relative geometric features under the same experimental protocol.

Temporal context is commonly introduced by encoding a fixed window of recent observations. This has been studied with GPS data [8], LiDAR measurements [9] and camera or radar frames [10]. These studies show that recent observations can support beam prediction, but they do not determine whether the benefit comes from temporal access itself or other factors in the experimental setup. For this reason, the time-series encodings in this study use timestamp information, recent observations and rolling movement statistics to capture short-term positional context.

AI/ML-assisted beam management has been studied in 3GPP Release 18 as a way to reduce beam-measurement overhead. Spatial-domain beam prediction, or BM-Case1, predicts the best downlink beam or beam pair for the current spatial setting, while time-domain beam prediction, or BM-Case2, predicts the best beam for future time instances from historical measurements [11]. Although these cases use beam-measurement inputs rather than GPS-derived position features, the BM-Case1/BM-Case2 distinction motivates our comparison between current-position encodings and history-aware position encodings.

A separate line of work learns representations instead of designing them manually. Generalisable beam prediction combines position-derived features with beam-domain measurements to improve robustness across urban scenarios [12], while other approaches encode position and beam information as tokenised sequences for large language models [13]. These methods are useful for maximising predictive performance, but learned or fused representations make it harder to attribute performance differences to a specific encoding decision. In contrast, this work keeps the encodings explicit and deterministic, so that the different effects can be compared directly.

2.2 Gap in Current Work

Existing research shows that position, geometry and temporal context can support beam prediction, but it does not isolate the effect of the position encoding itself. Prior beam-prediction studies usually evaluate complete systems, so changes in representation are often mixed with changes in modality, architecture, task or experimental setting [6; 7; 8; 9; 10; 12; 13]. As a result, their reported gains cannot be attributed solely to the encoding choice.

This leaves a gap at the intersection of 6G data, tabular encoding and time-series encoding. Existing studies do not systematically compare how these encoding choices affect beam-prediction performance and representation quality under the same experimental conditions. We address this gap with a shared experimental protocol across all encodings.

3 Preliminaries

3.1 Problem Setting and Notation

The downstream task is beam-index prediction based on position [6]. Each observation contains position-related measurements from the UE, a timestamp and the received power of the available communication beams. The beam with the highest measured power is used as the prediction target.

For scenario s, observation i is represented as

$$\left(\mathbf{x}_i^{(s)}, \tau_i^{(s)}, y_i^{(s)}\right),$$

where $\mathbf{x}_i^{(s)}$ contains the raw position-related features, $\tau_i^{(s)}$ is the timestamp and $y_i^{(s)} \in \{1, \ldots, K\}$ is the optimal beam index among K candidate beams. An encoder transforms the available input into a feature vector

$$\mathbf{z}_i^{(s)} \in \mathbb{R}^d,$$

where d is the number of encoded features.

In the time-series setting, the timestamp and sample order are preserved. For a context window of length w, an encoder may use the current observation and up to $w-1$ preceding observations, provided that all observations belong to the same sequence. At the beginning of a sequence, where fewer than $w-1$ preceding observations exist, the unavailable context positions are left empty.

3.2 Relative to Base Station Geometry

BS Bearing, BS Geometry and BS Rolling describe the UE's position as seen from the BS. We first project the geographic coordinates onto a metric plane using the Universal Transverse Mercator (UTM) projection [14]. The UE's offsets from the fixed BS at time t are

$$e_t = E_t - E_{\mathrm{BS}}, \qquad n_t = N_t - N_{\mathrm{BS}}, \tag{1}$$

with $(E_t, N_t) = \Pi(\phi_t, \lambda_t)$ and $(E_{\mathrm{BS}}, N_{\mathrm{BS}}) = \Pi(\phi_{\mathrm{BS}}, \lambda_{\mathrm{BS}})$, where ϕ and λ denote latitude and longitude, respectively. The corresponding distance and compass bearing are

$$d_t = \sqrt{e_t^2 + n_t^2}, \qquad \theta_t = \mathrm{atan2}(e_t, n_t). \tag{2}$$

Here $\mathrm{atan2}$ denotes the two-argument arctangent. The argument order follows the compass convention, where $\theta_t = 0$ points north and positive angles increase clockwise.

3.3 Tabular Encodings

Raw Sample Encoding. Raw Sample represents each observation using the original position measurements provided by the dataset.

Latitude-Longitude Encoding (Lat/Lon). Lat/Lon represents the UE's current latitude and longitude, (ϕ_t, λ_t) [6].

Base Station Bearing Encoding (BS Bearing). BS Bearing represents the current UE position using the distance and bearing defined in Equation 2:

$$d_t, \quad \sin\theta_t, \quad \cos\theta_t, \quad \theta_t.$$

The sine and cosine terms provide a continuous representation of bearing across the boundary between $-\pi$ and π, while the raw angle is retained because beam directions are related to the direction of the UE around the BS.

Piecewise-Linear Encoding (PLE). PLE maps each continuous feature into quantile-based bins while preserving the relative magnitude and position of values within the feature's numerical range [15]. For bin edges $b_0 < b_1 < \cdots < b_B$ fitted on the training data, a value x is encoded as

$$e_\ell(x) = \mathrm{clip}\left(\frac{x - b_{\ell-1}}{b_\ell - b_{\ell-1}}, 0, 1\right), \qquad \ell = 1, \ldots, B. \quad (3)$$

Bins below x receive 1, the bin containing x receives a value in $(0, 1)$ and bins above x receive 0. The function $\mathrm{clip}(a, 0, 1)$ truncates a to the interval $[0, 1]$.

Periodic Fourier-Feature Encoding. Periodic encoding expands each continuous feature into sinusoidal components at multiple frequencies [15; 16]. Each feature value x is min-max scaled to $[0, 1]$ using parameters fitted on the training data and encoded as

$$[\sin(2\pi f_i x), \cos(2\pi f_i x)], \qquad i = 1, \ldots, n, \quad (4)$$

where each f_i is sampled once from $\mathcal{N}(0, \sigma_F^2)$ using a fixed seed. The Fourier components make high-frequency variation in continuous inputs easier for the neural network to represent [16].

3.4 Time-Series Encodings

Timestamp Encoding. Timestamp Encoding extends Raw Sample with elapsed time, time since the preceding observation and a cyclical representation of time of day. Elapsed time is measured from the beginning of each sequence, for sequence c in scenario s, this is defined as

$$T_i^{(s,c)} = \tau_i^{(s,c)} - \tau_1^{(s,c)}, \quad (5)$$

$$\Delta T_i^{(s,c)} = \begin{cases} 0, & i = 1, \\ \tau_i^{(s,c)} - \tau_{i-1}^{(s,c)}, & i > 1, \end{cases} \quad (6)$$

$$\omega_i^{(s)} = \frac{2\pi q_i^{(s)}}{86\,400}. \quad (7)$$

The resulting features are $T_i^{(s,c)}$, $\Delta T_i^{(s,c)}$, $\sin\omega_i^{(s)}$ and $\cos\omega_i^{(s)}$, where $q_i^{(s)}$ is the time of day in seconds. The sine and cosine terms keep times around midnight close in the representation.

Lag Window Encoding. Lag Window Encoding augments Timestamp Encoding with feature values from preceding observations in the same sequence. For the m-th feature and a lag of k observations, the lagged value is defined as

$$L_{i,k}^{(s,m)} = \begin{cases} x_{i-k}^{(s,m)}, & i > k, \\ \text{missing}, & i \le k. \end{cases} \quad (8)$$

where $x_{i-k}^{(s,m)}$ denotes the value of feature m at observation $i-k$ in scenario s. The value is missing when $i \le k$ because the required preceding observation does not exist.

Rolling Statistics Encoding (Rolling). Rolling Statistics Encoding extends Timestamp Encoding with metric position, movement and short-term summary features. It first adds the current-sample features

$$a_i^{(s)}, \quad b_i^{(s)}, \quad r_i^{(s)}, \quad \Delta a_i^{(s)},$$
$$\Delta b_i^{(s)}, \quad m_i^{(s)}, \quad v_i^{(s)}, \quad \sin h_i^{(s)}, \quad \cos h_i^{(s)},$$

where $a_i^{(s)}$ and $b_i^{(s)}$ denote the UE's metric offsets from the mean training-set position, computed after projecting latitude and longitude to UTM coordinates [14], $r_i^{(s)}$ is its distance from the reference point, $\Delta a_i^{(s)}$ and $\Delta b_i^{(s)}$ are the changes in position since the preceding observation, set to zero when no preceding observation exists, $m_i^{(s)}$ is the movement distance, $v_i^{(s)}$ is the estimated speed and $\sin h_i^{(s)}$ and $\cos h_i^{(s)}$ represent the movement heading cyclically. The encoding additionally computes rolling means and standard deviations over a window w for

$$a_i^{(s)}, \quad b_i^{(s)}, \quad m_i^{(s)}, \quad v_i^{(s)}.$$

For a feature u, the rolling statistics are

$$\mu_i^{(s,w)}(u) = \frac{1}{w}\sum_{j=0}^{w-1} u_{i-j}^{(s)}, \quad (9)$$

$$\sigma_i^{(s,w)}(u) = \sqrt{\frac{1}{w-1}\sum_{j=0}^{w-1}\left(u_{i-j}^{(s)} - \mu_i^{(s,w)}(u)\right)^2}. \quad (10)$$

At the beginning of a sequence, rolling statistics are computed over the available observations only.

Base Station Geometry Encoding (BS Geometry). BS Geometry extends BS Bearing with the east and north offsets from Equation 1 and with the change in selected geometric quantities from the preceding observation:

$$\Delta p_t = p_t - p_{t-1}, \qquad p_t \in \{d_t, \sin\theta_t, \cos\theta_t\}. \quad (11)$$

For the first observation in each sequence, the change features are set to zero because no preceding observation is available. The resulting features describe both the current UE-to-BS geometry and its most recent change. BS Geometry is treated as a time-series encoding because it depends on the ordering of observations.

Base Station Rolling Encoding (BS Rolling). BS Rolling extends BS Bearing with rolling statistics that summarise the recent UE-to-BS geometry. For each window width w, it adds

$$\mu_i^{(s,w)}(d), \quad \sigma_i^{(s,w)}(d), \quad \mu_i^{(s,w)}(\sin\theta),$$
$$\sigma_i^{(s,w)}(\sin\theta), \quad \mu_i^{(s,w)}(\cos\theta), \quad \sigma_i^{(s,w)}(\cos\theta).$$

For a geometric feature u, the rolling mean and standard deviation are defined as in Equations 9 and 10, respectively.

The rolling means describe the recent distance and direction of the UE relative to the BS, while the rolling standard deviations describe how much this geometry varies over the window.

4 Experimental Design

The study compares encoding configurations under a shared protocol: scenario, task, split, model, hyperparameters, seed and metrics are fixed, making encoding the primary experimental variable.

The study is organised into two tracks. Each track is compared against its own baseline encoding, so that performance differences reflect the effect of the encoding rather than the effect of access to additional input information.

These tracks adapt the current-context versus history-aware distinction from 3GPP beam prediction to position-derived side information [4]. Unlike strict BM-Case1 and BM-Case2, this study does not use beam-measurement inputs. Instead, the tabular track evaluates how to encode the current UE position, while the time-series track evaluates whether recent position history and geometric change provide additional predictive information.

In the tabular track, each observation is treated independently. The model receives only the per-sample position features. The timestamp and the chronological order of the samples are not provided. The baseline for this track is Raw Sample, which applies only the numeric conversion required for the model to process the original features.

In the time-series track, the model is also given access to the time column and the order of the samples. This allows encoders to derive features from timestamps and from preceding observations. The baseline for this track is Timestamp, the simplest encoding that uses the time column. This makes it possible to isolate the effect of movement features from the effect of time access alone.

4.1 Experiment Pipeline

Each experimental run is defined by a configuration file that specifies the dataset, encoder, downstream model and output location. The same runner then executes every configuration in the comparison.

1. The dataset is loaded, cleaned and validated.
2. The dataset is partitioned into training, validation and test sets using a customisable split ratio.
3. The encoder is fitted on the training rows and then applied to all rows.
4. The model is trained on the encoded training set.
5. The model is evaluated on the validation and test sets.

Predictive metrics, representation statistics and the full configuration are stored for every run, so that any result can be traced to the exact settings that produced it. This configuration-driven design makes each run traceable and reproducible.

4.2 Downstream Model

Every encoding is evaluated using two fixed downstream models: a neural network and an XGBoost classifier [17]. The models are used as evaluation instruments, not as contributions. Their configurations are unchanged across encodings.

A neural network is included because it was previously developed and evaluated for the same task considered in this study, position-based beam prediction from a real-world 6G dataset [6].

XGBoost is included as a complementary tree-based model. Competitive performance has been reported for XGBoost in beam-selection tasks, including improvements over Support Vector Machines (SVM), K-Nearest Neighbours (KNN) and neural-network baselines under the evaluated experimental setting [18].

Using both models reduces the risk that an encoding appears effective only because it matches one learning algorithm. The exact model configurations and training parameters are reported in Section 5.4.

4.3 Evaluation Metrics

Each encoding is assessed along two dimensions: predictive performance and representation quality.

Predictive performance is measured using top-3 accuracy and power loss. Top-3 accuracy is the proportion of samples for which the optimal beam is included among the model's 3 highest-scored predictions. Power loss measures the received-power gap between the beam selected by the model and the optimal beam [6]. A lower power loss indicates better communication performance, with zero corresponding to no loss relative to the optimal beam. Power loss complements top-3 accuracy because an incorrectly classified beam may still retain received power close to that of the optimal beam.

Representation quality is evaluated through invariance, following Plachouras et al. [19]. Invariance measures how stable an encoding remains under controlled input perturbations. Here, Gaussian noise is added to the UE coordinates, the encoded feature vector is recomputed and the clean and perturbed representations are compared using cosine similarity. Higher similarity indicates greater invariance to positional noise.

4.4 Multi-Seed Protocol

Each experiment is evaluated using Monte Carlo cross-validation over eleven random seeds [20]. Each seed fixes the random split and state of the downstream model, so that one seed defines one fully reproducible run. Results are reported as the mean and standard deviation across the eleven seeds, reducing the influence of any favourable or unfavourable split. For a given scenario and seed, the same partition is used for every encoding. The exact seeds used for the experiments can be found in Section 5.3.

4.5 Statistical Significance

To test whether the observed performance differences are consistent across scenarios, each encoding is compared with the baseline encoding of its own track using a two-sided Wilcoxon signed-rank test [21; 22]. Each paired observation is the per-scenario difference in mean top-3 accuracy between an encoding and its track baseline. This tests whether the median scenario-level difference differs from zero without assuming normally distributed differences.

Since four encodings are compared with the baseline in each track, Holm-Bonferroni correction is applied separately

for each downstream model [23]. Differences with corrected $p < 0.05$ are treated as statistically significant.

5 Experimental Configuration

5.1 Dataset and Evaluation Setup

The experiment uses Scenarios 1-9 of the DeepSense 6G dataset [3]. These scenarios represent real-world Vehicle-to-Infrastructure (V2I) communication illustrated in Figure 1, in which a moving vehicle-mounted UE communicates with a fixed BS. These scenarios contain timestamped GPS measurements collected from a moving UE, together with wireless measurements indicating the received power of the available beams. This work uses the position-related measurements as model inputs and the corresponding optimal beam index as the prediction target. The optimal beam is the beam with the highest measured received power for that observation. Scenario details are summarised in Appendix Table 3.

Each sample represents the UE at one timestamp. Samples remain ordered within their original sequence so that time-series encodings can use preceding observations without crossing sequence boundaries. For encodings that describe the UE relative to the BS, the BS coordinates associated with the corresponding scenario are used to derive distance, bearing and geometric change features.

The invariance analysis uses noise levels $\sigma \in \{1, 2, 5, 10\}$ metres. These values define a controlled sensitivity range from small coordinate perturbations to stronger localisation errors. The goal is not to model one specific GPS error distribution, but to compare how different encoders respond to the same positional perturbation. For each noise level, results are averaged over three independent noise realisations using seeds 0, 1 and 2.

5.2 Encoding Configuration

The encoding choices are intended to test distinct representation assumptions rather than to perform an exhaustive feature-engineering search.

In DeepSense 6G, the available position-related fields for the UE are latitude, longitude, Position Dilution of Precision (PDOP), Horizontal Dilution of Precision (HDOP), direction and number of satellites [3], although each encoder may retain only a subset of these fields or replace them with derived features.

PLE uses $B = 10$. This provides a moderate discretisation of each continuous feature, adding non-linearity without making the representation unnecessarily sparse. Periodic encoding uses $n = 6$ and $\sigma_F = 2.0$ to provide a compact multi-scale representation. Lag Window uses a window size of $w = 3$, while Rolling and BS Rolling use a window size of $w \in \{3, 5\}$. These short windows capture immediate movement while limiting feature growth and avoiding longer histories that may be less reliable when local beam conditions change.

5.3 Data Splitting and Random Seeds

The dataset is split at sequence level, so all observations from the same DeepSense sequence remain in a single partition. This prevents temporally adjacent observations from appearing in both training and evaluation, reducing the risk that the model is tested on samples that are almost identical to those seen during training. An approximate $70/15/15$ train-validation-test split is applied at the sequence level, with small deviations in observation counts because sequences differ in length. Similar split ratios were also considered but did not meaningfully change the results. Each configuration is evaluated using seeds $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 42\}$.

5.4 Model Configuration

The neural network follows Morais et al. [6], but adapts the input layer to encoder-dependent feature vectors of 2-35 features and replaces the original preprocessing with median imputation and standardisation. XGBoost uses 300 estimators and class-balanced sample weights, with all remaining hyperparameters kept at their library defaults. For both classifiers, the random seed is set to the experiment seed. The complete architecture and training configurations are reported in Appendix Table 2.

5.5 Hardware and Software Environment

All experiments were run on Windows 10 using an Intel Core i7-1165G7 CPU and 12 GB RAM. The software environment used Python 3.11.9, XGBoost 3.2.0, scikit-learn 1.7.2, pandas 2.3.3, NumPy 2.3.5, torch 2.12.0 and utm 0.8.1.

6 Results

6.1 Predictive Performance

Table 1 reports the test performance of all encodings averaged across the nine DeepSense 6G scenarios and eleven random seeds. The strongest results are obtained by encodings that describe the UE relative to the BS. In the tabular track, BS Bearing is the best encoding for both downstream models. Relative to Raw Sample, it increases top-3 accuracy from $62.6 \pm 2.6\%$ to $67.5 \pm 2.2\%$ and reduces power loss from 3.12 ± 0.28 dB to 2.78 ± 0.31 dB for the neural network. For XGBoost, it increases top-3 accuracy from $57.6 \pm 2.2\%$ to $61.1 \pm 1.9\%$ and reduces power loss from 3.32 ± 0.41 dB to 2.93 ± 0.82 dB. The remaining tabular encodings also improve over Raw Sample in average top-3 accuracy for both downstream models. For power loss, however, the improvement is less consistent, as PLE and Periodic increase it when XGBoost is used.

This pattern is more pronounced in the time-series track. BS Geometry reaches the best overall performance, with $70.7 \pm 2.5\%$ top-3 accuracy and 2.47 ± 0.37 dB power loss for the neural network, and $63.8 \pm 2.0\%$ top-3 accuracy and 2.74 ± 0.51 dB power loss for XGBoost. Compared with Timestamp, this corresponds to a top-3 improvement of 8.5 percentage points for the neural network and 4.5 percentage points for XGBoost. BS Rolling is the second-best time-series encoding, while BS Geometry achieves the highest aggregate performance across both downstream models and both predictive metrics.

Figure 2 shows that these aggregate gains are not uniform across scenarios. The clearest case is Scenario 6, where BS Geometry increases neural-network top-3 accuracy from

Table 1: Predictive performance per encoding on the test split. Values are mean ± standard deviation over eleven seeds after averaging across the nine scenarios. Encodings are grouped by track and compared against the baseline of their own track.

		Neural network		XGBoost	
Track	Encoding	Top-3 (%)	Power Loss (dB)	Top-3 (%)	Power Loss (dB)
Tabular	Raw Sample (baseline)	62.6 ± 2.6	3.12 ± 0.28	57.6 ± 2.2	3.32 ± 0.41
	Lat/Lon	66.5 ± 2.4	2.85 ± 0.33	59.5 ± 2.2	3.05 ± 0.34
	PLE	64.6 ± 2.4	2.80 ± 0.34	59.9 ± 2.2	3.38 ± 0.23
	Periodic	64.3 ± 1.6	3.00 ± 0.31	57.9 ± 1.7	3.51 ± 0.26
	BS Bearing	$\mathbf{67.5 \pm 2.2}$	$\mathbf{2.78 \pm 0.31}$	$\mathbf{61.1 \pm 1.9}$	$\mathbf{2.93 \pm 0.82}$
Time-series	Timestamp (baseline)	62.2 ± 2.1	3.08 ± 0.46	59.3 ± 2.7	3.41 ± 0.31
	Lag Window	60.8 ± 1.7	3.07 ± 0.59	60.8 ± 2.2	3.01 ± 0.46
	Rolling	63.6 ± 2.1	3.09 ± 0.40	60.3 ± 2.4	3.24 ± 0.26
	BS Geometry	$\mathbf{70.7 \pm 2.5}$	$\mathbf{2.47 \pm 0.37}$	$\mathbf{63.8 \pm 2.0}$	$\mathbf{2.74 \pm 0.51}$
	BS Rolling	68.7 ± 2.7	2.72 ± 0.49	63.3 ± 1.9	2.92 ± 0.34

38.02% with Timestamp to 79.57%, a gain of 41.55 percentage points. For XGBoost, the same comparison increases top-3 accuracy from 55.73% to 64.30%, a gain of 8.57 percentage points. Other scenarios show smaller or less consistent gains, especially for encodings that rely on recent temporal context.

PLE and Periodic add feature transformations, but remain below BS Bearing in top-3 accuracy. Averaged over the two downstream models, PLE is 2.1 percentage points lower than BS Bearing, while Periodic is 3.2 percentage points lower. Similarly, Lag Window and Rolling add temporal information, but remain below BS Geometry by 6.5 and 5.3 percentage points on average, respectively. Across both models, the BS-relative encodings achieve the highest top-3 accuracy within their respective tracks. Possible reasons for the scenario-level variation are discussed in Section 7.

The complete statistical comparisons for test top-3 accuracy are reported in Appendix Table 4. After Holm-Bonferroni correction within each track, BS Bearing improves significantly over Raw Sample for both downstream models, and BS Geometry and BS Rolling improve significantly over Timestamp for both models. Lat/Lon improves significantly for the neural network but not for XGBoost, consistent with the smaller tabular gains observed for the tree model. By contrast, Lag Window and Rolling do not differ significantly from Timestamp for either model, and the numerical embeddings yield at most a small significant gain, limited to PLE on XGBoost.

6.2 Representation Quality

Predictive performance shows how well each encoding supports the downstream beam-prediction task, but it does not indicate how sensitive the encoding is to perturbations. To analyse this, representation quality is evaluated through invariance, which refers to the stability of an encoded feature vector when controlled positional noise is added to the UE coordinates.

Figure 3 reports the cosine similarity between each clean representation and the representation recomputed after the same controlled Gaussian perturbations. The most stable encodings are the simpler current-sample representations. Timestamp, Raw Sample and Lag Window remain relatively stable as the noise level increases, with mean cosine similarities of 0.659, 0.585 and 0.584 at $\sigma = 10\ m$. Among the geometry-aware encodings, BS Bearing degrades the least under small perturbations, retaining a cosine similarity of 0.966 at $\sigma = 1\ m$, but it falls to 0.378 at $\sigma = 10\ m$.

The lowest cosine similarities occur for encodings that derive movement or rolling geometric statistics from the coordinates. At $\sigma = 10\ m$, Rolling, BS Geometry and BS Rolling reach mean cosine similarities of 0.050, 0.094 and 0.164, respectively. At this noise level, the encodings with the highest predictive performance are also among those with the lowest cosine similarity. The exact values are reported in Appendix Table 5.

7 Discussion

The results indicate that encoding affects beam prediction because it determines which physical structure is made explicit to the downstream model. The strongest encodings are not simply larger versions of the raw input, they expose the UE-to-BS relation on which beam selection depends.

BS-relative geometry improves performance because it removes a geometric inference step from the model. Raw latitude and longitude locate the UE, but do not explicitly describe its relation to the BS. BS Bearing provides this relation directly through distance and bearing, which explains why it is the strongest tabular encoding for both downstream models. This is consistent with prior work showing that GPS position supports beam prediction, but refines that result by showing that position is more useful when represented relative to the BS [6]. Location- and orientation-aided beam-selection work supports the same interpretation, since these methods also expose physical information that constrains the beam direction [7; 8].

Temporal context helps only when it describes relevant movement. Timestamp and Lag Window add time-related or historical information, but they do not significantly improve over the Timestamp reference. In contrast, BS Geometry and BS Rolling combine sequence order with BS-relative position and give the strongest time-series results. This suggests that the useful part of temporal context is not the presence of

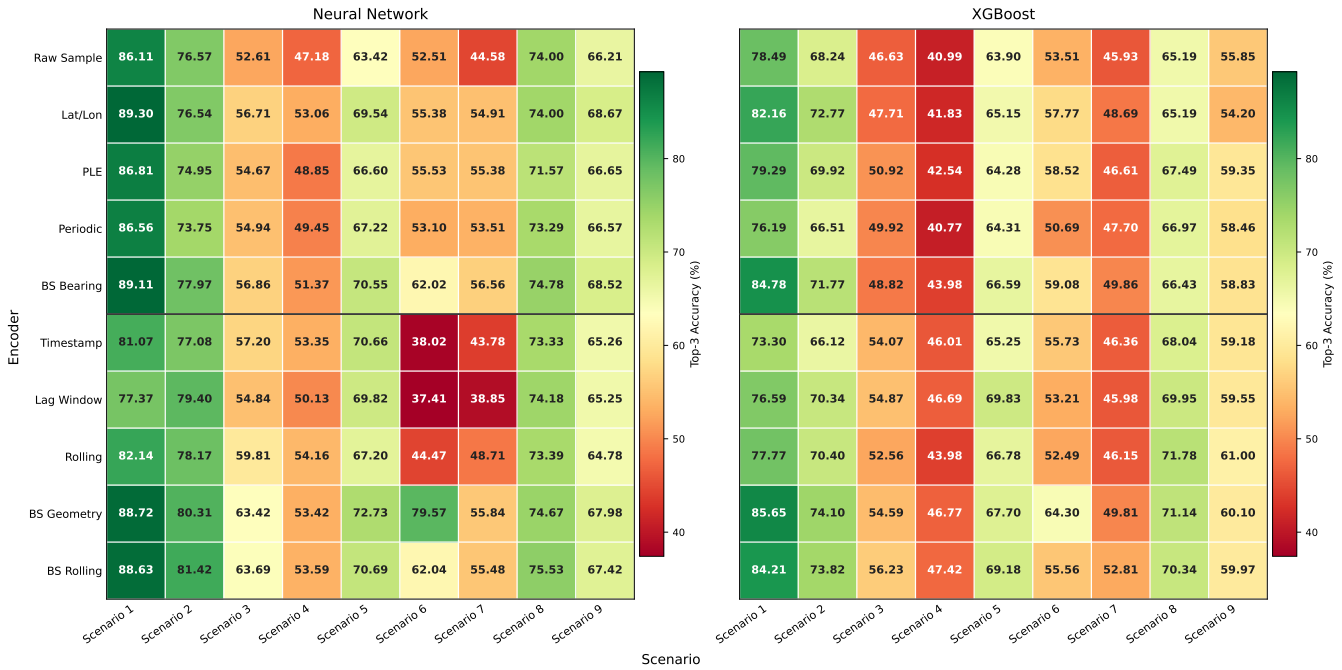

Figure 2: Per-scenario top-3 accuracy mean for every encoding over eleven seeds.

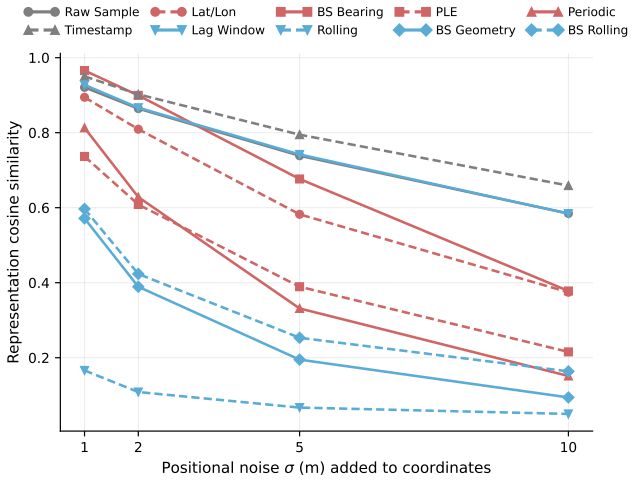

Figure 3: Encoding invariance to synthetic GPS noise, measured by mean cosine similarity across Scenarios 1-9.

previous rows by itself, but the change in the UE-to-BS relation. This is consistent with sequence-based beam-prediction and beam-tracking studies, where recent observations help because they describe user movement or future beam evolution [8; 9; 10].

The scenario-level results suggest that this temporal benefit depends on the physical setting. The largest gain appears in Scenario 6, while Scenarios 3, 4 and 7 show weaker or less consistent gains from temporal encodings. One plausible explanation is the road speed limit in the measurement area: Scenarios 1, 2, 5, 6, 8 and 9 were measured on roads with a maximum speed limit of 25 miles per hour, whereas Scenarios 3 and 4 were measured on roads with a 45 miles per hour limit and Scenario 7 on roads with a 55 miles per hour limit. At higher permitted speeds, a short history window may describe a less stable local context for the current beam decision [24].

Scenario 6 also has the largest distance between the vehicle trajectory and the BS, which may make explicit BS-relative geometry especially valuable. However, Scenarios 6 and 7 also contain fewer samples than the other scenarios, so part of the observed variation may reflect higher estimation uncertainty rather than only physical differences. These scenario-level observations therefore indicate possible explanations for the results, but they do not isolate a single cause.

The comparison also separates expressiveness from usefulness. PLE and Periodic increase feature dimensionality, and Lag Window and Rolling add temporal information, but none of them outperforms the BS-relative encodings. Numerical embeddings can make continuous features easier for a model to use [15; 16], but they do not add the missing physical relation required by the task. In this experiment, a smaller encoding that exposes the relevant communication geometry is more effective than a larger encoding that mainly increases representation dimensionality.

This distinction matters for AI/ML-assisted beam management. 3GPP studies distinguish between current-context and history-aware prediction settings [4; 5], but the present results show that the representation of side information also matters. Treating position as raw coordinates, BS-relative geometry or recent geometric change leads to different downstream behaviour, even when the dataset, split, model and seeds are fixed. Future benchmarks should therefore specify not only

which side information is available, such as position or trajectory, but also how that information is represented before it is used.

The invariance results show the main cost of making geometry explicit. BS Geometry and BS Rolling improve prediction, but they are less stable under positional noise than simpler encodings. This is expected because bearing, movement differences and rolling statistics can amplify coordinate perturbations. A small change in latitude and longitude can become a larger change after computing angular features, consecutive differences or statistics over a short window. In deployment, geometry-aware encodings should therefore be paired with positioning-quality checks, smoothing or noise-aware training. When position estimates are noisy, a simpler but more stable encoding may be preferable to the encoding that performs best under clean GPS conditions.

7.1 Threats to Validity

Several factors limit the scope of these findings.

The study uses a single dataset, DeepSense 6G. Although it spans nine real-world scenarios of differing difficulty, other 6G datasets may exhibit different geometry and noise characteristics.

Only a selected set of encodings is evaluated. Other representations, such as learned embeddings or convolutional transforms, may behave differently and are left to future work.

Because models and hyperparameters are fixed, the comparison measures performance under a shared protocol rather than each encoding's individually tuned optimum.

The temporal conclusions depend on the chosen time-series design. Lag Window, Rolling and BS Rolling use a short fixed history window. Longer windows or different sampling rates may change the usefulness of temporal context.

The significance tests reduce the risk of over-interpreting aggregate differences, but they are based on at most nine paired scenarios and therefore have limited statistical power. Non-significant results should therefore be interpreted as insufficient evidence of a consistent scenario-level effect, not as evidence that no effect exists.

The invariance comparison is also affected by differences in feature composition. Some encodings contain several features that are independent of latitude and longitude, while others consist almost entirely of GPS-derived features. Consequently, the cosine similarity considered in this study may favour representations containing more unchanged features.

Finally, the invariance analysis uses synthetic Gaussian noise on UE coordinates. Real positioning errors may be environment-dependent and biased.

8 Responsible Research

8.1 Reproducibility

Reproducibility is supported by configuration files, stored run outputs, fixed seeds, a public dataset and released source code. For each run, the pipeline stores the predictive metrics, representation statistics and runtime measurements, allowing any reported number to be traced to the settings that produced it. Randomness is controlled by a user-selected seed, which determines the split, classifier state and stochastic encoder components, so that a run can be repeated exactly. At the time of publication, the dataset was publicly available, with no stated expiry date [3]. The experiment source code, instructions for reproducing the runs and dashboard for visualising the results are available in the GitHub repository [25].

8.2 Transparency and Honest Reporting

All results are reported regardless of outcome. Negative and inconclusive findings are kept rather than removed. The encoding and preprocessing choices are stated explicitly so that the comparison can be judged on its design.

8.3 Ethical Considerations

This research presents limited ethical risks, but two are worth addressing.

First, the dataset contains positional measurements, which can be sensitive even when released for research. The data are used only to train and evaluate classifiers, with no attempt to identify individuals or to reconstruct the movement of any person.

Second, generative AI tools, including ChatGPT, were used as assistive tools during the writing and development process. Their use was limited to improving wording, structure and clarity, as well as providing implementation-level feedback such as debugging suggestions and code readability improvements. These tools were not used to define the research question, choose the experimental design, implement the core contribution, generate results, select reported outcomes, alter numerical values, or replace the author's interpretation. Any AI-generated suggestion was reviewed, verified against the source code, dataset documentation, cited literature and stored experimental outputs, and revised before inclusion. No sensitive, confidential or proprietary data were entered into generative AI tools. The author retains full responsibility for the final text, code, analysis, interpretation and academic integrity of the work.

9 Conclusion and Future Work

This paper investigated how different encoding techniques affect beam-prediction performance and the quality of feature representations derived from 6G network data. A controlled experiment on nine DeepSense 6G scenarios isolates how encoding affects performance.

The results show that encoding choices affect beam-prediction performance. BS Bearing achieves the strongest tabular performance, while BS Geometry achieves the strongest time-series and overall performance. These findings indicate that BS-relative distance, bearing and recent geometric change provide useful information for beam selection, whereas PLE and Periodic, which mainly expand the inputs, provide less consistent improvements.

The representation analysis identifies a trade-off between predictive performance and sensitivity to positional perturbations. BS Geometry and BS Rolling achieve strong predictive performance but relatively low cosine similarity under increasing positional noise. Simpler encodings, including Raw

Sample and Timestamp, remain more invariant. This suggests that predictive performance under clean conditions and representation invariance capture distinct properties and should be considered separately when selecting an encoding.

Future work could test whether the same pattern holds beyond the current setting. This includes evaluating additional real-world 6G datasets and simulated ray-tracing scenarios, for example using tools such as DeepMIMO [26] and Sionna [27], as well as testing other beam-prediction tasks or model families. Future work could also investigate representations based on additional sensing modalities, including LiDAR, camera and radar data, both individually and in combination with position information. Another direction is to incorporate robustness to GPS noise into encoding design and evaluate its effect on downstream predictive performance. Such extensions would test the generalisability of the findings across a wider range of BS layouts, user trajectories and propagation environments.

References

[1] Q. Xue, C. Ji, S. Ma, J. Guo, Y. Xu, Q. Chen, and W. Zhang, "A survey of beam management for mmwave and THz communications towards 6G," *IEEE Communications Surveys & Tutorials*, vol. 26, no. 3, p. 1520–1559, 2024. [Online]. Available: http://dx.doi.org/10.1109/COMST.2024.3361991

[2] Y. Shi, L. Lian, Y. Shi, Z. Wang, Y. Zhou, L. Fu, L. Bai, J. Zhang, and W. Zhang, "Machine learning for large-scale optimization in 6G wireless networks," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2088–2132, 2023. [Online]. Available: http://dx.doi.org/10.1109/COMST.2023.3300664

[3] A. Alkhateeb, G. Charan, T. Osman, A. Hredzak, J. Morais, U. Demirhan, and N. Srinivas, "DeepSense 6G: A large-scale real-world multi-modal sensing and communication dataset," 2023. [Online]. Available: https://arxiv.org/abs/2211.09769

[4] 3GPP, "Study on Artificial Intelligence (AI)/Machine Learning (ML) for NR Air Interface," 3rd Generation Partnership Project (3GPP), Technical Report TR 38.843, Dec. 2023, Release 18. [Online]. Available: https://www.3gpp.org/dynareport/38843.htm

[5] X. Lin, "An overview of the 3GPP study on artificial intelligence for 5G new radio," 2023. [Online]. Available: https://arxiv.org/abs/2308.05315

[6] J. Morais, A. Behboodi, H. Pezeshki, and A. Alkhateeb, "Position aided beam prediction in the real world: How useful GPS locations actually are?" 2022. [Online]. Available: https://arxiv.org/abs/2205.09054

[7] S. Rezaie, E. de Carvalho, and C. N. Manchón, "A deep learning approach to location- and orientation-aided 3D beam selection for mmWave communications," 2021. [Online]. Available: https://arxiv.org/abs/2110.06859

[8] V. Ardianto Nugroho and B. Moo Lee, "GPS-aided deep learning for beam prediction and tracking in UAV mmwave communication," *IEEE Access*, vol. 13, p. 117065–117077, 2025. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2025.3586594

[9] S. Jiang, G. Charan, and A. Alkhateeb, "LiDAR aided future beam prediction in real-world millimeter wave V2I communications," 2022. [Online]. Available: https://arxiv.org/abs/2203.05548

[10] M. Ma, I. Welgamage, A. Alkhateeb, A. L. Swindlehurst, M. Juntti, and N. T. Nguyen, "Knowledge distillation for lightweight multimodal sensing-aided mmwave beam tracking," 2026. [Online]. Available: https://arxiv.org/abs/2604.16708

[11] N. Jayaweera, A. Bonfante, M. Schamberger, A. M. A. Tehrani, T. Sanguanpuak, P. Tilak, K. Jayasinghe, F. W. Vook, and N. Rajatheva, "5G-advanced AI/ML beam management: Performance evaluation with integrated ML models," 2024. [Online]. Available: https://arxiv.org/abs/2404.15326

[12] Y. Jin, Y. Li, J. Jun, Y. Gao, S. Liu, J. Du, Z. Yang, and S. Xu, "Generalizable and robust beam prediction for 6G networks: An deep-learning framework with positioning feature fusion," 2026. [Online]. Available: https://arxiv.org/abs/2602.09685

[13] Y. Sheng, K. Huang, L. Liang, P. Liu, S. Jin, and G. Y. Li, "Beam prediction based on Large Language Models," 2025. [Online]. Available: https://arxiv.org/abs/2408.08707

[14] T. Bieniek, B. van Andel, and T. I. Bø, "utm: Bidirectional UTM-WGS84 converter for Python," https://github.com/Turbo87/utm, 2025.

[15] Y. Gorishniy, I. Rubachev, and A. Babenko, "On embeddings for numerical features in tabular deep learning," 2023. [Online]. Available: https://arxiv.org/abs/2203.05556

[16] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," 2020. [Online]. Available: https://arxiv.org/abs/2006.10739

[17] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. ACM, Aug. 2016, p. 785–794. [Online]. Available: http://dx.doi.org/10.1145/2939672.2939785

[18] Z. Zarei, F. Tilahun, and C. Kang, "Camera-assisted beam selection via unified learning in 6G vehicle-to-infrastructure (V2I) scenarios," *IEEE Access*, vol. PP, pp. 1–1, 01 2026. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2026.3671352

[19] C. Plachouras, J. Guinot, G. Fazekas, E. Quinton, E. Benetos, and J. Pauwels, "Towards a unified representation evaluation framework beyond downstream tasks," 2025. [Online]. Available: https://arxiv.org/abs/2505.06224

[20] Q.-S. Xu and Y.-Z. Liang, "Monte Carlo cross validation," *Chemometrics and Intelligent Laboratory Systems*, vol. 56, no. 1, pp. 1–11, 2001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169743900001222

[21] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945. [Online]. Available: http://www.jstor.org/stable/3001968

[22] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. 1, pp. 1–30, 2006. [Online]. Available: http://jmlr.org/papers/v7/demsar06a.html

[23] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979. [Online]. Available: http://www.jstor.org/stable/4615733

[24] DeepSense 6G, "Scenarios," https://www.deepsense6g.net/scenarios, accessed: Jun. 21, 2026.

[25] P. Fernandez Luengo, "pelayofdez/cse3000-encoding-exp-and-dash: 6G beam prediction encoding experiments and dashboard," 2026. [Online]. Available: https://doi.org/10.5281/zenodo.20784948

[26] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," 2019. [Online]. Available: https://arxiv.org/abs/1902.06435

[27] J. Hoydis, S. Cammerer, F. A. Aoudia, A. Vem, N. Binder, G. Marcus, and A. Keller, "Sionna: An open-source library for next-generation physical layer research," 2023. [Online]. Available: https://arxiv.org/abs/2203.11854

A Additional Tables

Table 2: Architecture and training configuration of the downstream models.

Neural Network (Morais Adaptation)

Setting	Value
Input	Encoding-dependent (2-35 features)
Preprocessing	Median imputation and standardisation
Hidden layers	3
Units per hidden layer	256
Activation	ReLU
Output size	Training-observed classes (57-62)
Loss	Cross-entropy
Optimizer	Adam
Initial learning rate	0.01
Learning-rate decay	Epochs 20 and 40
Reduction factor	0.2
Batch size	32
Training epochs	60
Adam β_1, β_2	0.9, 0.999 (default)
Adam ϵ	10^{-8} (default)
Weight decay	0 (default)
Weight initialisation	Kaiming-uniform (default)
Random state	Experiment seed

XGBoost

Setting	Value
Number of estimators	300
Tree method	`hist`
Class balancing	Balanced sample weights
Missing-value handling	Median imputation
Booster	`gbtree` (default)
Objective	`multi:softprob` (default)
Learning rate (η)	0.3 (default)
Max tree depth	6 (default)
Min child weight	1 (default)
Min split loss (γ)	0 (default)
Subsample	1.0 (default)
Column subsample (per tree)	1.0 (default)
L2 regularisation (λ)	1 (default)
L1 regularisation (α)	0 (default)
Max bins	256 (default)
Random state	Experiment seed

Table 3: DeepSense 6G scenarios used in the experiments. "GNSS quality" indicates whether satellite-count, PDOP, HDOP, fix-type or DGPS measurements are available. Scenario 4 does not provide `unit1_beam_index`, the target is derived as the argmax of the `unit1_pwr_60ghz` array.

Scenario	Samples	Sequences	UE GPS used	GNSS quality	Beam target
1	2,411	29	Raw	Yes	Provided
2	2,974	34	Raw	Yes	Provided
3	1,487	52	Raw	Yes	Provided
4	1,867	72	Raw	Yes	Derived
5	2,300	29	Raw	Yes	Provided
6	915	12	Raw	Yes	Provided
7	854	63	Raw	Yes	Provided
8	4,043	83	Calibrated	No	Provided
9	5,964	136	Calibrated	Yes	Provided

Table 4: Per-track statistical significance for test top-3 accuracy. Each encoding is compared with its track baseline. Δ is the median per-scenario difference in percentage points. Bold indicates corrected $p < 0.05$.

		Neural network			XGBoost		
Track	**Encoding**	$\mathbf{\Delta}$	p_{raw}	p_{Holm}	$\mathbf{\Delta}$	p_{raw}	p_{Holm}
Tabular	Lat/Lon	$+3.64$	0.016	**0.047**	$+2.00$	0.055	0.109
	PLE	$+1.87$	0.039	0.078	$+1.62$	0.008	**0.031**
	Periodic	$+1.43$	0.109	0.109	$+0.09$	0.844	0.844
	BS Bearing	$+4.23$	0.008	**0.031**	$+3.26$	0.008	**0.031**
Time-series	Lag Window	-0.85	0.129	0.258	$+0.80$	0.098	0.195
	Rolling	$+1.07$	0.129	0.258	$+1.53$	0.359	0.359
	BS Geometry	$+3.23$	0.004	**0.016**	$+3.10$	0.004	**0.016**
	BS Rolling	$+4.34$	0.004	**0.016**	$+2.30$	0.008	**0.023**

Table 5: Encoding invariance under positional noise. Values are mean cosine similarities between clean and perturbed test representations, averaged over three noise realisations within each scenario and then equally across the nine scenarios.

Track	**Encoding**	$\boldsymbol{\sigma = 1\,\mathrm{m}}$	$\boldsymbol{\sigma = 2\,\mathrm{m}}$	$\boldsymbol{\sigma = 5\,\mathrm{m}}$	$\boldsymbol{\sigma = 10\,\mathrm{m}}$
Tabular	Raw Sample	0.921	0.864	**0.739**	**0.585**
	Lat/Lon	0.894	0.809	0.582	0.375
	PLE	0.737	0.608	0.389	0.216
	Periodic	0.813	0.628	0.331	0.151
	BS Bearing	**0.966**	**0.899**	0.677	0.378
Time-series	Timestamp	**0.950**	**0.902**	**0.795**	**0.659**
	Lag Window	0.927	0.867	0.742	0.584
	Rolling	0.166	0.109	0.067	0.050
	BS Geometry	0.572	0.389	0.195	0.094
	BS Rolling	0.597	0.423	0.253	0.164