

QoS Prediction in V2X Networks Using Decision Tree Ensemble Methods Under Different Spatial Training Strategies

Rodrigo D'Amorim Barreto, Luis Alves Correia Filho and Marcela Silva Novo

Abstract—QoS prediction can improve reliability, adaptability and decision-making in V2X networks. In this paper five ensemble machine learning methods based on Decision Tree regressors were evaluated, analyzing their performance when applied to a QoS prediction problem, under three different spatial training split strategies, obtaining training and testing time, MAE and R^2 scores. Training and testing were performed using a filtered public dataset, with measurements collected from vehicles across different urban areas. All evaluated models are available in the Scikit-Learn platform. Results indicate that Histogram-Based Gradient Boosting (HGB) provides the best trade-off between computational cost and prediction accuracy for large V2X datasets, while balanced splitting strategies improve generalization when representative training samples are preserved.

Keywords—Decision Trees, Ensembles, Machine Learning, QoS prediction, V2X networks.

I. INTRODUCTION

Vehicle to Everything (V2X) networks are increasingly more diverse and widespread. This type of connection enables smarter and safer transport environments [1]. Prediction of Quality of Service (QoS) parameters can improve reliability, adaptability and decision-making capabilities when integrated in such networks [2].

Decision trees are widely used in machine learning applications due to their interpretability and robustness. In order to tackle complex problems, such as QoS in V2X networks, ensemble techniques that use multiple decision trees as predictors are widely used. This paper evaluates and compares decision tree ensemble methods supported by the Scikit-Learn framework for QoS prediction in heterogeneous V2X scenarios. Additionally, this work investigates how different spatial balancing strategies in the training dataset impact prediction accuracy, computational cost and model generalization.

The main contribution of this work is the evaluation of how spatial balancing strategies affect the trade-off between prediction performance, computational cost and dataset representativeness in V2X QoS prediction.

Section II describes the experimental configuration, including the computational environment, dataset preprocessing procedures, training split strategies and hyperparameter tuning process. Section III presents an overview of each evaluated

ensemble method, including the resulting hyperparameter configurations. Section IV presents and discusses the comparison metrics and experimental results. Section V concludes with practical recommendations based on the obtained results.

II. EXPERIMENT CONFIGURATIONS

The experiment was conducted using a computer equipped with Intel(R) Core(TM) i5 10400 CPU 2.90GHz processor and 16GB of RAM memory. The algorithms were implemented using the Scikit-Learn [3] (version 1.8.0) library for Python, in the Google COLAB workspace.

A. Dataset

A concise overview of the measurement campaign and dataset is presented here, with a detailed description available in Hernangómez et al. (2023) [4]. The Berlin V2X dataset was used for this experiment. It includes data obtained from four vehicles in two distinct configurations through the same set route: the four vehicles in tandem, and two separate pairs.

As shown in Figure 1, the selected route includes data collected from different areas of the city of Berlin (residential area, park, avenue, tunnel and highway), as such, the dataset provides a representative characterization of distinct vehicular network connectivity conditions.

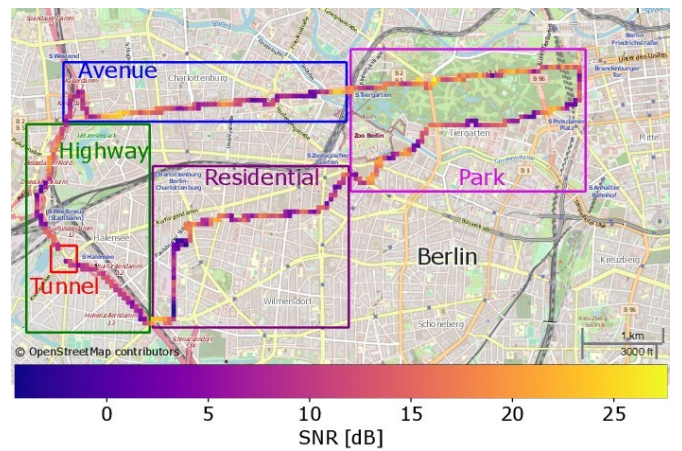


Fig. 1. A measurement route conducted in Berlin with SNR heatmap for operator1. Source: obtained from [4].

Following previous filtering of this dataset for downlink data rate prediction [5], null or incomplete data were removed,

along with stationary measurements, where speed was measured as 0 km/h or null.

Before filtering, the dataset had 207434 measurement points, after filtering, it has 32060 points, corresponding to approximately 15.5% of the original dataset size.

The selected features for datarate prediction were the following:

- Physical layer (PHY): Signal-to-Noise Ratio (SNR), Reference Signal's Received Power (RSRP), Reference Signal's Received Quality (RSRQ), and Received Signal Strength Indicator (RSSI).
- Shared physical channels: assigned resource blocks, transport block size, and modulation and coding scheme (MCS).
- Radio Resource Control (RRC): cell identity, frequency, and bandwidth.

B. Training Splits

The dataset was split into training, testing and validation sets.

Three different training splits were developed for comparing. They are distinct regarding how many points from each area were randomly selected for the training split. From the 32060 points, 11698 are from the park area, 8580 are from the residential area, 7066 are from the avenue area, 4527 are from the highway area and only 189 are from the tunnel area.

1) *Split 1*: In the first split, every area is divided proportionally, 80% of each area is set for Training, 10% for Testing and the final 10% for Validation. As such, this split has more park points than any other area. This split has 25646 points for training, 3207 points for testing and 3207 points for validation.

2) *Split 2*: In the second split, every area appears evenly in the training set to avoid prediction bias. This strategy was designed to reduce spatial bias during training by enforcing equal representation among geographical areas. As such, 80% of the tunnel area appear in the training set, which equals 151 points, and every other area also has 151 points in the training set. The rest of the points for every area are evenly split between testing and validation. This split has 755 points for training, 15651 points for testing and 15651 points for validation.

3) *Split 3*: In this last split, every area but the tunnel area appears evenly in the training set. As the tunnel area has significantly less measurement points than any other (it is less than 5% the size of the highway area, the second smallest area), it could limit the training potential if we include it in the even splitting. Split 3 was designed as a hybrid balancing strategy, preserving spatial representativeness while avoiding severe training set reduction caused by the underrepresented tunnel measurements. In this split, 80% of the highway area appear in the training set, which equals 3621 points, and every other area besides the tunnel also has 3621 points in the training set. The tunnel has 80% of its points in the training set. The rest of the points for every area are as evenly split between testing and validation as possible without removing any points. This split has 14635 points for training, 8711 points for testing and 8714 points for validation.

C. Hyperparameter Tuning

The optimization metric adopted during hyperparameter tuning was the squared error loss. As it preserves the same ordering behavior as the R^2 metric, the selected hyperparameter configuration corresponded to the highest obtained R^2 score.

For tuning, it was first used the Random Search method, exploring a broad hyperparameter search space. Then it was used the Grid Search Method, exploring a narrower, more focused range of the selected hyperparameters, centered around the resulting configuration from the Random Search method. The results displayed in this paper had their hyperparameters chosen as the tuning result from this final Grid Search method.

The Random Search and Grid Search methods are available in the scikit learn library as `RandomizedSearchCV` and `GridSearchCV`, respectively.

III. DECISION TREES ENSEMBLES

Five different ensemble methods that use decision trees as predictors are compared in this paper. The hyperparameters omitted from Tables II, III and IV in this section were constant throughout all splits, and set to their default values from the Scikit-Learn library version 1.8.0.

A. Adaptive Boosting

Adaptive Boosting (ADA) is a machine learning algorithm where sequentially trained predictors iteratively adapt to correct errors produced by previous predictors [6].

Table I shows the selected hyperparameters tuned using the Grid Search method and their tuning result, used in each split for the experiment.

TABLE I
ADAPTIVE BOOSTING TUNING.

Split	estimators	learning rate
Split 1	49	1.09
Split 2	30	1.07
Split 3	62	1.09

The other hyperparameters were chosen to be constant throughout all splits. The max depth of the decision tree (using the `DecisionTreeRegressor` function from Scikit-Learn library) was set to 14 and the loss was set to linear.

B. Gradient Boosting

Gradient Boosting (GB) is a machine learning algorithm where predictors are sequentially combined to correct previous errors [7].

Table II shows the selected hyperparameters tuned using the Grid Search method and their tuning result, used in each split for the experiment.

TABLE II
GRADIENT BOOSTING TUNING.

Split	max depth	estimators
Split 1	5	108
Split 2	2	48
Split 3	2	199

C. Histogram Based Gradient Boosting

Histogram Based Gradient Boosting (HGB) is a machine learning algorithm that speeds up standard GB by bucketing continuous features into discrete bins (histograms), which significantly improves computational efficiency, especially for bigger datasets [8].

Table III shows the selected hyperparameters tuned using the Grid Search method and their tuning result, used in each split for the experiment.

 TABLE III
HISTOGRAM BASED GRADIENT BOOSTING TUNING.

Split	max iterations	max leaf nodes
Split 1	117	26
Split 2	43	25
Split 3	111	17

D. Random Forest

Random Forest (RF) is a machine learning algorithm where each decision tree is trained in a random subset of data and each node split considers a random subset of features [9].

Table IV shows the selected hyperparameters tuned using the Grid Search method and their tuning result, used in each split for the experiment.

 TABLE IV
RANDOM FOREST TUNING.

Split	max depth	estimators	max features
Split 1	56	460	0.5
Split 2	22	186	0.8
Split 3	9	195	0.7

E. Extremely Randomized Trees

Extremely Randomized Trees (XRT) is a machine learning algorithm similar to the Random Forest, where the main differences are the use of random thresholds for node splits instead of optimal split calculations, and the use of the entire training dataset instead of random data subsets.[10].

Table VI shows the selected hyperparameters tuned using the Grid Search method and their tuning result, used in each split for the experiment.

IV. COMPARISON RESULTS

Each algorithm was evaluated according to four different metrics:

- **Training Time:** The time it took to train the model on the data before it is ready for the test.
- **Prediction Time:** The time it took for the model to return its predictions on the test data.
- **R^2 score:** An important evaluation metric that measures the goodness of fit of a model with respect to the training data.
- **Mean Absolute Error (MAE):** A direct metric of the difference between real and predicted values, commonly used in regression analysis.

Be y the real value, $\hat{y}$ the predicted value and $\bar{y}$ the mean of the real values, we can calculate the R^2 score from a group of n points as follows in (1):

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{j=1}^n (y_j - \hat{y}_j)^2}{\sum_{j=1}^n (y_j - \bar{y})^2} \quad (1)$$

We can also calculate the MAE value from a group of n points as follows in (2):

$$MAE(y, \hat{y}) = \frac{1}{n} \sum_{j=0}^{n-1} |y_j - \hat{y}_j| \quad (2)$$

Table V shows the results of all configurations, every test was performed five times, and the results are presented in *mean* $\pm$ *standard deviation* format.

A. Comparing Splits

Comparing the performance of the same ensemble model between every split, we can determine that for every case, the training time was highest on the Split 1 and lowest on the Split 2. This is mainly because Split 1 has the biggest training set, while Split 2 has the smallest.

The prediction time is highest on the Split 2 and lowest on the Split 1 for ADA, GB and HGB. This is consistent with the size of each testing split, where Split 2 has the biggest test set and Split 1 has the smallest one. However, RF has its highest prediction time in the Split 1 and XRT has its biggest prediction time in Split 3. For RF, this is likely the case because the hyperparameter tuning for RF yielded 460 estimators for the first split, while yielding 186 and 195 for Splits 2 and 3, respectively. Similarly for XRT, it got 680 estimators for Split 3, while only 198 estimators for Split 2.

The R^2 score is highest on the Split 3 and lowest on the Split 2 for all models, while the MAE is highest on the Split 2 and lowest on the Split 1 for all models except for XRT, which has its lowest MAE in Split 3. These results indicate that enforcing strict spatial balancing may reduce training bias, but can significantly impair generalization capability when the resulting training set becomes excessively small. These results also suggest that excessively proportional training distributions may increase the risk of spatial overfitting, while excessively balanced splits may impair generalization due to insufficient training diversity. Split 3, however, achieved a better trade-off between spatial balancing and training representativeness

TABLE V
RESULTS OBTAINED FROM EACH CONFIGURATION

Model	Training Time (s)	Prediction Time (s)	R^2	MAE (Mbps)
SPLIT 1 - Proportional Split				
ADA	15.422 ± 2.491	0.186 ± 0.003	0.931 ± 0.002	5.546 ± 0.019
GB	33.532 ± 0.422	0.017 ± 0.003	0.934 ± 0.000	5.417 ± 0.007
HGB	1.372 ± 0.423	0.070 ± 0.045	0.937 ± 0.001	5.376 ± 0.039
RF	158.996 ± 1.724	0.732 ± 0.035	0.933 ± 0.000	5.418 ± 0.005
XRT	44.255 ± 4.312	0.401 ± 0.057	0.933 ± 0.000	5.377 ± 0.004
SPLIT 2 - Even Split				
ADA	1.036 ± 0.490	1.081 ± 0.104	0.916 ± 0.001	6.472 ± 0.035
GB	0.244 ± 0.004	0.074 ± 0.011	0.926 ± 0.000	6.147 ± 0.003
HGB	0.491 ± 0.128	0.498 ± 0.142	0.920 ± 0.000	6.383 ± 0.000
RF	2.470 ± 0.964	0.491 ± 0.147	0.929 ± 0.000	5.941 ± 0.009
XRT	1.030 ± 0.193	0.481 ± 0.070	0.931 ± 0.000	5.828 ± 0.007
SPLIT 3 - Even Split, Proportional Tunnel Split				
ADA	8.886 ± 0.964	0.697 ± 0.206	0.938 ± 0.000	5.706 ± 0.035
GB	14.769 ± 0.720	0.033 ± 0.001	0.940 ± 0.000	5.445 ± 0.001
HGB	0.514 ± 0.233	0.091 ± 0.028	0.940 ± 0.001	5.466 ± 0.023
RF	26.650 ± 1.366	0.230 ± 0.033	0.940 ± 0.000	5.426 ± 0.009
XRT	16.885 ± 0.546	0.697 ± 0.122	0.941 ± 0.000	5.365 ± 0.005

TABLE VI
EXTREMELY RANDOMIZED TREES TUNING.

Split	max depth	estimators	max features
Split 1	10	700	1.0
Split 2	12	198	1.0
Split 3	9	680	1.0

regarding the R^2 score, but it also had a higher MAE, which suggests that Split 3 tends to develop models that better capture general prediction trends, but cannot predict extreme outliers as accurately.

B. Comparing Models

The fastest training model is HGB, having the lowest training time in all splits, with the difference becoming more pronounced as the training set size increases. RF is the slowest training model in all splits.

GB is the fastest predicting model, being quickest in all the splits. Thus, the Histogram Based approach trades prediction speed for faster training performance in these cases.

For the Split 3, all models had very similar R^2 , with only a 0.003 variation between the lowest and highest values. ADA had the worst R^2 score in all splits, and also the highest MAE, indicating that it was the model with the largest prediction error among the evaluated approaches.

C. Additional Evaluation

To further evaluate the proposed spatial training strategy, all splits were evaluated using the same test set, obtained from a *Split 1* partition, with Histogram-Based Gradient Boosting.

Using the same dataset, a fourth splitting strategy (*Split 4*) was implemented. In this strategy, only the two areas with the

largest number of data points (park and residential) are equally represented in the training set. Thus, except for the park area, 80% of the data points from every other area are included in the training set.

Additionally, the dataset was repartitioned into new groups, still following the previous split divisions (*Split 1, 2 and 3*). These splits are different however, because the group of data points from each area was distributed differently into the different sets, building new training and validation sets.

For each split, hyperparameter tuning was performed again using Grid Search. The search space was based on Table III, with max iterations values between 40 and 150, and max leaf nodes values between 15 and 31.

Table VII shows a slight increase in R^2 for *Split 4*, and lower MAE for *Split 1*, in accordance with previous analysis. *Split 3* has a worse performance when compared to 1 and 4, which suggests that limiting the number of data points from the highway area hindered predictions for the park and residential areas in this new split, while evenly balancing only the residential and park areas improved generalization. This also suggests the bigger training sets improved the R^2 score in the Table V.

TABLE VII
HISTOGRAM BASED GRADIENT BOOSTING PERFORMANCE ON THE SAME TESTING SET AND UNDER NEW SPLITS.

Split	R^2	MAE (Mbps)
Split 1	0.9348 ± 0.0006	5.4069 ± 0.0282
Split 2	0.9047 ± 0.0000	6.5934 ± 0.0000
Split 3	0.9306 ± 0.0006	5.5219 ± 0.0212
Split 4	0.9350 ± 0.0007	5.4220 ± 0.0180

V. CONCLUSIONS

All decision tree ensembles are capable of obtaining high prediction performance in data rate estimation tasks. In larger V2X datasets, it is recommended to use Histogram Based Gradient Boosting since it should yield smaller training time with still quick and accurate predictions. Among the evaluated approaches, HGB demonstrated the best balance between scalability, computational efficiency and prediction accuracy. HGB seems to be the most suitable model among the evaluated approaches for large-scale V2X and intelligent transportation system applications with increasing data availability.

Evenly splitting in order to minimize prediction bias is only recommended when the involved groups can still be adequately represented in the resulting training set. The reduced number of tunnel measurements after filtering negatively impacted the representativeness of balanced training strategies, resulting in more challenging prediction scenarios.

Future work includes the evaluation of deep learning approaches, additional V2X datasets and temporal sequence-aware prediction strategies in heterogeneous vehicular environments.

ACKNOWLEDGEMENTS

The authors thank the Federal Institute of Education, Science and Technology of Bahia (IFBA) and the Federal University of Bahia (UFBA).

REFERENCES

- [1] D. C. Moreira, I. M. Guerreiro, W. Sun, C. C. Cavalcante and D. A. Sousa, "QoS Predictability in V2X Communication with Machine Learning", *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, Antwerp, Belgium, 2020, pp. 1-5.
- [2] Barmounakis, Sokratis, et al. "AI-driven, QoS prediction for V2X communications in beyond 5G systems", *Computer Networks*, 2022, 217, p. 109341.
- [3] K. Jolly. "Machine learning with scikit-learn quick start guide: classification, regression, and clustering techniques in Python", *Packt Publishing Ltd*, 2018.
- [4] R. Hernangómez et al., "Berlin V2X: A Machine Learning Dataset from Multiple Vehicles and Radio Access Technologies", *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, Florence, Italy, 2023, pp. 1-5.
- [5] L. A. C. Filho, A. L. C. Ottoni and M. S. Novo, "Predição de Qualidade de Serviço em Redes V2X Utilizando Técnicas de Aprendizado de Máquina", *XLII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais- SBt*, 2024, BELÉM, PA.
- [6] H. Drucker, "Improving Regressors using Boosting Techniques", in *Proceedings of the 14th International Conference on Machine Learning (ICML)*, San Francisco, CA, USA, 1997, pp. 107-115.
- [7] J. H. Friedman, "Stochastic Gradient Boosting", *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367-378, 2002.
- [8] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T. Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree", in *Advances in Neural Information Processing Systems (NIPS)*, vol. 30, Long Beach, CA, USA, 2017, pp. 3146-3154.
- [9] BREIMAN, Leo. Random forests. *Machine Learning*, v. 45, n. 1, p. 5-32, 2001.
- [10] P. Geurts, D. Ernst and L. Wehenkel, "Extremely randomized trees", *Machine Learning*, vol. 63, no. 1, pp. 3-42, 2006.