Inferring Traffic Control Policies with Supervised Learning

A Case Study on Max Pressure

Abohariri, Robin; Tan, Chaopeng; Rinaldi, Marco; Van Lint, Hans

DOI
10.1109/MT-ITS68460.2025.11223570

Publication date
2025

Document Version
Final published version

Published in
2025 9th International Conference on Models and Technologies for Intelligent Transportation Systems, MT-ITS 2025

2025 9th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS) | 979-8-3315-8063-6/25/$31.00 ©2025 EU | DOI: 10.1109/MT-ITS68460.2025.11223570

INFERRING TRAFFIC CONTROL POLICIES WITH SUPERVISED LEARNING: A CASE STUDY ON MAX PRESSURE

Robin Abohariri[1,*], Chaopeng Tan[1], Marco Rinaldi[1], Hans van Lint[1]

1. Department of Transport and Planning, Delft University of Technology, Gebouw 23, Stevinweg 1, Delft, 2628CN, The Netherlands.
r.abohariri@tudelft.nl (R. Abohariri), tantantan951122@gmail.com (C. Tan), m.rinaldi@tudelft.nl (M. Rinaldi), j.w.c.vanlint@tudelft.nl (H. Lint)

* Corresponding author

Regular paper submitted for presentation at 9th International IEEE Conference on Models and Technologies for Intelligent Transportation Systems 2025

September 8-10, 2025, Kirchberg, Luxembourg

Abstract: Smart traffic systems, like those using well-established methods such as SCOOT, SCATS and TUC, aim to improve traffic flow by dynamically adjusting signal timings based on real-time traffic conditions. Traffic engineers need to understand the objective functions behind traffic signal control to analyze, improve, and optimize network performances. However, different jurisdictions, different operators and competing interests imply that the underlying objective functions governing traffic signal control might not be publicly known with sufficient detail (e.g. to preserve Intellectual Property Rights). A method for discovering these functions is therefore needed, particularly to enable better cooperation among stakeholders. In this work, we train computer models to mimic the decisions made by smart traffic light systems. Using data from a simulated traffic network (with virtual sensors tracking vehicles), we test a variety of supervised models, ranging from simple decision trees to more complex neural networks. Our results show these models can accurately mimic the underlying system's actions, achieving up to 99% accuracy. This work demonstrates that supervised learning can serve as a powerful tool for uncovering hidden traffic control functions by training models to replicate the system's decisions. By analyzing these models, we can then infer the key factors influencing signal control, thereby gaining insights into the underlying objective function.

Keywords: Traffic Signal Control; Supervised Learning; Feature Importance Analysis

INTRODUCTION

Traffic management systems rely on hidden or proprietary objective functions to optimize intersection signal control. In real-world deployments, these functions are often unknown to external stakeholders, such as neighboring jurisdictions' road authorities, public transport (PT) operators. While road agencies typically manage traffic signals, PT operators must adapt their operations based on how these systems prioritize public transport vehicles. However, since signal control strategies are often proprietary, PT operators may struggle to predict how changes in traffic management policies, such as modifications to bus priority levels—will impact service efficiency. For example, a road authority may adjust traffic signal priorities to either increase or decrease bus priority at intersections. If priority is reduced, PT operators might need to compensate by adjusting schedules, increasing fleet size, or rerouting services to maintain reliability. Conversely, if priority is enhanced, buses may experience fewer delays, leading to cost savings and improved service frequency. Understanding how an existing traffic control system makes these decisions is crucial for both road agencies and PT operators, yet proprietary signal control algorithms often obscure this information, leading operators to rely on real-time information, which limits the stakeholders' planning horizon and (in worst case scenarios) might impose constraints on the maximum reachable service quality. Furthermore, lack of insight in other stakeholders' objectives might prevent identifying opportunities for collaborative efforts (e.g. jointly optimizing priority and schedules, as opposed to reactive adjustment). This study investigates whether supervised learning can be used to uncover such hidden objective functions by training machine learning models to approximate a known traffic control function. To achieve this, we first deploy our recently introduced traffic management strategy—Max-Pressure Control(Varaiya (2013))—as a testbed for experimentation. MP optimally selects traffic signal phases based on vehicle states and queue lengths. We then collect training data from a simulated traffic network, capturing key traffic indicators such as number of vehicles, mean speed, and jam length. Using this dataset, we train various supervised learning models including Decision Trees, Random Forests, Gradient Boosting, SVM, and Neural Networks to approximate the Priority-MP decision-making process. If these models can successfully reconstruct MP's behavior, this suggests that a similar approach could be applied in real-world scenarios where the true objective function is unknown. This approach is inspired by previous studies demonstrating the power of supervised learning for function approximation in traffic systems. The work of J. S. Angarita-Zapata (2019) highlights how historical traffic data can be leveraged to model system behaviors, predicting future conditions based on

past patterns. Similar methodologies have been used in reverse-engineering complex 3D scanning and CAD modeling problems, where machine learning techniques infer underlying design intent from raw geometric data in Lim (2024). By applying these principles to traffic control, we aim to approximate the decision function behind a real-time signal system—without requiring direct access to its internal logic. This research contributes to the development of methods for reverse-engineering traffic signal control functions by demonstrating how machine learning can model the decision rules of an obfuscated objective function, such as Max pressure. By identifying key factors influencing traffic signal decisions, such as time loss, queue length, and vehicle numbers, this approach provides valuable insights for road authorities, PT operators, researchers and traffic engineers. By reverse-engineering MP using supervised learning, we show that data-driven models can serve as a proxy for unknown traffic optimization functions, enabling better decision-making in multi-stakeholder traffic management scenarios.

METHODOLOGY

We first introduce the core method we are attempting to reverse-engineer: Max-pressure (MP). The Max-pressure algorithm develops an acyclic, dynamic phase length traffic signal controller. The Max-pressure algorithm models vehicles in lanes as a substance in a pipe and enacts control in a manner which attempts to maximize the relief of pressure between incoming and outgoing lanesVaraiya (2013). For a given green phase p, the pressure is defined in (1).

$$\text{Pressure}(p) = \sum_{l \in L_{p,\text{inc}}} |V_l| - \sum_{l \in L_{p,\text{out}}} |V_l| \tag{1}$$

Where $L_{p,\text{inc}}$ represents the set of incoming lanes with green movements in phase p and $L_{p,\text{out}}$ represents the set of outgoing lanes from all incoming lanes in $L_{p,\text{inc}}$. Pseudo-code for the Max-pressure traffic signal controller is presented in Algorithm 1.

Algorithm 1 Max-pressure Algorithm

```
procedure MAXPRESSURE(g_min, t_p, P)
    if t_p < g_min then
        t_p ← t_p + 1
    else
        t_p ← 0    #Next phase has largest pressure
        Return arg max({Pressure(p) | p ∈ P})
    end if
end procedure
```

To evaluate whether machine learning can approximate MP, we follow a structured pipeline involving two experiments. In the first experiment, we use only the number of vehicles—the key feature employed by MP—as input to train multiple supervised classification models, including Decision Trees, Random Forests, Support Vector Machines (SVM), and Neural Networks, to predict MP-selected traffic phases. In the second experiment, we augment the feature set with additional variables collected from the traffic environment to reflect the complexity of real-world data, where the underlying traffic logic is unknown. We retrain the models to assess the impact of these extra features and conduct feature importance analysis to investigate whether the models continue to prioritize vehicle count as the dominant factor, consistent with the closed-form MP formulation. Both experiments are performed on a single-intersection simulation and repeated on a simplified toy network to evaluate robustness and generalization. The models are assessed based on their prediction accuracy and the alignment of their feature importance with MP's theoretical basis.

1. Traffic Simulation and Data Collection

We conduct experiments on two scenarios: a single intersection and a toy network with multiple intersections. The single intersection consists of one junction with four legs. Each leg has three incoming lanes and two outgoing lanes (Fig. 1), with E2 detectors installed on all lanes, totaling 20 detectors. From each detector, we collect four features—vehicle counts, mean speed, jam length, and detector occupancy—resulting in an input matrix $\mathbf{X}$ with 80 columns.

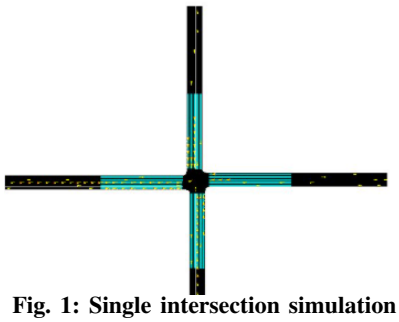
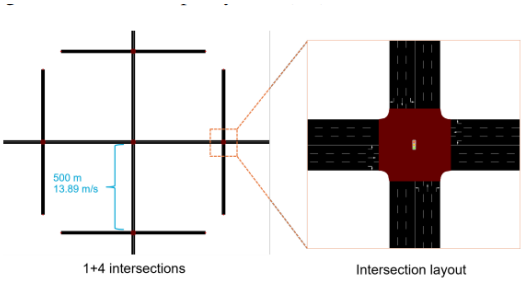


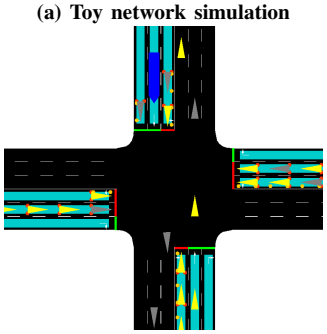
Fig. 1: Single intersection simulation

The toy network scenario includes five intersections: one central and four peripheral, each with four legs. Each link has three lanes designated for left turns, straight-through, and right turns (Fig. 2a). E2 detectors are deployed on 24 lanes, including outgoing lanes, simulating ANPR cameras (Fig. 2b). However, 8 lanes with constant green signals are excluded, leaving 16 lanes with detectors. From each detector, the same four features are collected, forming an input matrix $\mathbf{X}$ with 64 columns. Simulations are run using the microscopic SUMO framework Lopez et al. (2018) across various demand levels, with vehicle arrival probabilities of [0.3, 0.22, 0.20, 0.18, 0.16, 0.14, 0.12, 0.10, 0.05] representing the likelihood of a vehicle

arriving each second. Each scenario is simulated for 4,800 seconds, providing comprehensive data under varying congestion levels. We record the MP-selected traffic phase at each traffic signal phase change, which serves as the target variable **y** for supervised learning.



(a) Toy network simulation



(b) E2 detectors placed in the junction

Fig. 2

2. Machine Learning Models

We formulate the problem as a multi-class classification task, where each traffic phase is treated as a discrete class. The following models from Pedregosa et al. (2011) are trained to predict MP decisions: **Decision Tree**: A rule-based hierarchical model. **Random Forest**: An ensemble of decision trees that reduces variance and improves prediction performance. **Gradient Boosting**: A boosting approach that minimizes prediction errors. **Support Vector Machine (SVM)**: A non-linear classifier that seeks an optimal non-linear decision boundary. **Neural Network (NN)**: A multi-layer perceptron (MLP) for capturing complex non-linear relationships. Each model is trained using an 80:20 train-test split, with 5-fold cross-validation to improve generalization. To evaluate model performance, we use *test accuracy*, which is defined as the ratio of correctly predicted traffic phases to the total number of test samples. Mathematically, the test accuracy can be expressed as:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Test Samples}} = \frac{\sum_{i=1}^{N} \mathbf{1}(y_i = \hat{y}_i)}{N}$$

where y_i is the true label, $\hat{y}_i$ is the predicted label, and N is the total number of test samples. This provides an overall assessment of how well each model replicates the MP decisions. This provides an overall assessment of how well each model replicates the MP decisions.

3. Feature Importance Analysis

The MP model selects signal phases based on traffic pressure based vehicle count features. If the trained models—such as Decision Trees, Random Forests, and Logistic Regression—correctly approximate the behavior of MP, these features should emerge as the most important in feature importance analysis. Feature importance analysis serves as a tool to validate whether the trained models align with the expected behavior of the MP model. Each machine learning algorithm employs different methods to assess feature importance. Decision Trees determine importance based on how often and effectively a feature splits the data at decision nodes, with features near the root considered more important. Random Forests calculate feature importance by averaging the decrease in impurity (e.g., Gini) across ensembled decision trees, offering a more robust measure. Logistic Regression assesses importance through model coefficients, with larger absolute coefficients indicating a stronger influence on predictions. In Gradient Boosting Friedman (2001), feature importance is based on how much each feature reduces the loss function across all trees in the ensemble. By comparing the most important features identified by each model with those explicitly employed by the MP model, we can assess success in uncovering the underlying objective function.

Table III summarizes the top 16 most important features for both the single intersection and toy network scenarios, as identified by LR and NN. Here, vehN refers to vehicle count, out indicates outgoing lanes, and in represents incoming lanes.

RESULTS

1. Model Performance

To evaluate whether machine learning can approximate MP, we follow a structured pipeline involving two experiments. In the first experiment, we use only the number of vehicles—the key feature employed by MP—as input to train multiple supervised classification models, including Decision Trees, Random Forests, Support Vector Machines (SVM), and Neural Networks (NN), to predict MP-selected traffic phases. The second experiment augments the feature set with additional variables to reflect the complexity of real-world data, where the underlying traffic logic is unknown. We retrain the models to assess the impact of these extra features and conduct feature importance analysis to investigate whether the models continue to prioritize vehicle count as the dominant factor, consistent with the closed-form MP formulation. Both experiments are performed on a single-intersection simulation and repeated on a simplified toy

TABLE I: Model Accuracy with 95% Confidence Intervals on Seen and Unseen Demand Levels (Single Intersection and Toy Network)

Scenario	Logistic Reg.	RF	Grad. Boost	Dec. Tree	SVM	NN
Seen Demand Levels (Trained on [0.1, 0.12, 0.16, 0.20])						
Single Intersection	0.99 [0.99–1.00]	0.83 [0.83–0.84]	0.95 [0.94–0.95]	0.76 [0.76–0.77]	0.99 [0.99–0.993]	0.99 [0.994–0.999]
Toy Network	0.99 [0.99–1.00]	0.64 [0.62–0.66]	0.82 [0.81–0.84]	0.52 [0.51–0.54]	0.89 [0.88–0.90]	0.99 [0.994–0.999]
Unseen Demand Levels						
0.05 Single Intersection	0.74 [0.72–0.76]	0.20 [0.24–0.26]	0.46 [0.43–0.47]	0.30 [0.29–0.32]	0.64 [0.61–0.66]	0.75 [0.73–0.77]
0.05 Toy Network	1.00 [1.00–1.00]	0.58 [0.57–0.60]	0.82 [0.81–0.83]	0.46 [0.44–0.48]	0.85 [0.83–0.86]	0.99 [0.99–1.00]
0.1 Single Intersection	0.88 [0.86–0.89]	0.22 [0.22–0.24]	0.78 [0.77–0.80]	0.4 [0.39–0.42]	0.83 [0.82–0.84]	0.88 [0.87–0.89]
0.1 Toy Network	0.99 [0.99–1.00]	0.73 [0.71–0.74]	0.88 [0.87–0.89]	0.52 [0.51–0.54]	0.89 [0.88–0.90]	0.99 [0.99–1.0]
0.18 Single Intersection	0.99 [0.99–1.00]	0.77 [0.75–0.79]	0.92 [0.91–0.93]	0.65 [0.63–0.67]	0.99 [0.99–1.00]	0.99 [0.99–1.00]
0.18 Toy Network	0.99 [0.99–1.00]	0.56 [0.55–0.58]	0.76 [0.75–0.77]	0.42 [0.42–0.44]	0.91 [0.90–0.92]	0.99 [0.99–1.00]
0.22 Single Intersection	0.99 [0.99–1.00]	0.75 [0.73–0.76]	0.89 [0.88–0.90]	0.63 [0.61–0.65]	0.99 [0.99–1.00]	0.99 [0.99–1.00]
0.22 Toy Network	0.99 [0.99–1.00]	0.43 [0.42–0.44]	0.47 [0.45–0.48]	0.35 [0.34–0.36]	0.84 [0.83–0.85]	0.95 [0.95–0.96]
0.30 Single Intersection	0.99 [0.99–1.00]	0.80 [0.79–0.82]	0.94 [0.94–0.95]	0.65 [0.63–0.66]	0.99 [0.99–1.00]	0.99 [0.99–1.00]
0.30 Toy Network	0.98 [0.98–0.98]	0.28 [0.28–0.30]	0.29 [0.28–0.40]	0.39 [0.39–0.40]	0.82 [0.82–0.83]	0.94 [0.93–0.94]
Test Accuracy (Trained on All Demand Levels)						
Single Intersection	0.97 [0.96–0.97]	0.73 [0.72–0.73]	0.84 [0.84–0.85]	0.74 [0.73–0.75]	0.95 [0.95–0.96]	0.96 [0.96–0.97]
Toy Network	1.00 [1.00–1.00]	0.62 [0.61–0.63]	0.83 [0.82–0.84]	0.59 [0.58–0.6]	0.83 [0.82–0.84]	0.99 [0.99–‘1.00]

TABLE II: Model Accuracy with 95% Confidence Intervals on Seen and Unseen Demand Levels After Feature Augmentation (Single Intersection and Toy Network)

Scenario	Logistic Reg.	RF	Grad. Boost	Dec. Tree	SVM	NN
Seen Demand Levels (Trained on [0.1, 0.12, 0.16, 0.20])						
Single Intersection	0.99 [0.99–1.00]	0.81 [0.80–0.81]	0.86 [0.86–0.87]	0.75 [0.74–0.76]	0.99 [0.98–0.99]	0.97 [0.97–0.98]
Toy Network	0.99 [0.99–1.00]	0.62 [0.60–0.64]	0.79 [0.78–0.80]	0.47 [0.46–0.49]	0.89 [0.88–0.90]	0.96 [0.95–0.97]
Unseen Demand Levels						
0.05 Single Intersection	0.74 [0.72–0.76]	0.28 [0.26–0.30]	0.41 [0.39–0.43]	0.27 [0.26–0.29]	0.65 [0.63–0.67]	0.62 [0.60–0.64]
0.05 Toy Network	1.00 [1.00–1.00]	0.58 [0.56–0.59]	0.77 [0.76–0.78]	0.42 [0.40–0.44]	0.86 [0.85–0.87]	0.86 [0.85–0.87]
0.10 Single Intersection	0.88 [0.87–0.90]	0.46 [0.44–0.48]	0.67 [0.65–0.70]	0.45 [0.44–0.48]	0.83 [0.81–0.84]	0.83 [0.81–0.85]
0.10 Toy Network	1.00 [1.00–1.00]	0.71 [0.70–0.72]	0.84 [0.83–0.85]	0.49 [0.48–0.51]	0.90 [0.90–0.91]	0.94 [0.93–0.95]
0.18 Single Intersection	0.99 [0.99–0.99]	0.77 [0.76–0.79]	0.85 [0.83–0.86]	0.66 [0.64–0.68]	0.99 [0.99–1.00]	0.97 [0.96–0.97]
0.18 Toy Network	0.99 [0.99–1.00]	0.57 [0.55–0.58]	0.72 [0.71–0.73]	0.41 [0.40–0.42]	0.89 [0.87–0.90]	0.94 [0.94–0.95]
0.22 Single Intersection	0.99 [0.99–1.00]	0.75 [0.73–0.76]	0.84 [0.83–0.85]	0.66 [0.64–0.68]	0.99 [0.99–1.00]	0.97 [0.97–0.98]
0.22 Toy Network	0.99 [0.99–0.99]	0.41 [0.40–0.42]	0.47 [0.45–0.48]	0.31 [0.30–0.33]	0.83 [0.82–0.84]	0.88 [0.87–0.89]
0.30 Single Intersection	0.99 [0.99–1.00]	0.80 [0.79–0.82]	0.87 [0.85–0.88]	0.67 [0.65–0.68]	0.99 [0.99–1.00]	0.98 [0.97–0.98]
0.30 Toy Network	0.98 [0.98–0.98]	0.27 [0.26–0.27]	0.28 [0.28–0.29]	0.29 [0.28–0.30]	0.81 [0.80–0.82]	0.82 [0.82–0.83]
Test Accuracy (Trained on All Demand Levels)						
Single Intersection	0.96 [0.96–0.97]	0.73 [0.72–0.74]	0.81 [0.81–0.82]	0.75 [0.73–0.76]	0.95 [0.95–0.96]	0.97 [0.97–0.98]
Toy Network	0.99 [0.99–1.00]	0.62 [0.60–0.64]	0.79 [0.77–0.80]	0.47 [0.45–0.48]	0.89 [0.88–0.90]	0.96 [0.95–0.97]

network to evaluate robustness and generalization. Model performance is assessed based on prediction accuracy and the alignment of feature importance with MP's theoretical basis. We focus first on the results from the initial experiment using only vehicle count features. All trained models demonstrated high predictive accuracy when evaluated on the same demand levels used during training ([0.1, 0.12, 0.16, 0.20]), confirming that machine learning models can effectively learn MP's phase selection principles under familiar conditions. For example, Neural Networks and Logistic Regression achieved near-perfect accuracy across both the single-intersection and toy network scenarios under seen conditions. To assess generalization, the models were tested on unseen demand levels: 0.05, 0.1, 0.18, 0.22, and 0.30. As shown in Table I, model performance generally declined as the demand levels deviated further from the training range, with a pronounced effect on more complex scenarios like the toy network. Random Forests and Decision Trees, in particular, exhibited poor generalization, with accuracy dropping sharply at out-of-range demand levels. For instance, on the toy network at demand level 0.30, Random Forest accuracy fell to 28%, and Decision Trees to 39%, highlighting their limited ability to extrapolate beyond familiar conditions. Logistic Regression demonstrated notable robustness, maintaining high accuracy on most unseen demand levels, even for the toy network. Gradient Boosting and SVM, while performing well on seen demand levels, struggled to generalize effectively and showed instability, with significant accuracy drops

TABLE III: Top 16 Most Important Features from Logistic Regression (LR) and Neural Networks (NN) for Single Intersection and Toy Network

*Rank	Single Intersection		Toy Network	
	LR	NN	LR	NN
1	vehN_out_-_gneE7_0 (1.75)	vehN_inc_gneE12_2 (0.074)	vehN_intM_S_out_1 (4.35)	vehN_intM_S_in_1 (0.105)
2	vehN_out_-_gneE12_0 (1.73)	vehN_inc_-_gneE8_1 (0.074)	vehN_intM_N_out_1 (4.31)	vehN_intM_S_out_1 (0.101)
3	vehN_out_-_gneE7_1 (1.72)	vehN_inc_gneE7_1 (0.070)	vehN_intM_E_out_1 (4.28)	vehN_intM_E_in_1 (0.097)
4	vehN_out_gneE10_1 (1.58)	vehN_inc_gneE7_2 (0.068)	vehN_intM_W_out_1 (4.28)	vehN_intM_W_in_1 (0.092)
5	vehN_out_gneE8_0 (1.57)	vehN_inc_gneE12_1 (0.068)	vehN_intM_W_out_2 (4.26)	vehN_intM_W_out_1 (0.087)
6	vehN_out_gneE10_0 (1.50)	vehN_inc_gneE7_0 (0.067)	vehN_intM_S_out_2 (4.25)	vehN_intM_N_out_1 (0.082)
7	vehN_out_gneE8_1 (1.41)	vehN_-_inc_gneE8_2 (0.066)	vehN_intM_N_out_2 (4.21)	vehN_intM_N_in_1 (0.081)
8	vehN_inc_gneE7_1 (1.38)	vehN_inc_gneE12_0 (0.065)	vehN_intM_E_out_2 (4.20)	vehN_intM_S_out_2 (0.078)
9	vehN_inc_-_gneE10_1 (1.35)	vehN_inc_-_gneE10_1 (0.065)	vehN_intM_W_in_1 (3.30)	vehN_intM_E_in_2 (0.068)
10	vehN_inc_-_gneE10_2 (1.34)	vehN_inc_-_gneE8_0 (0.064)	vehN_intM_E_in_1 (3.25)	vehN_intM_N_in_2 (0.066)
11	vehN_out_-_gneE12_1 (1.33)	vehN_inc_-_gneE10_2 (0.061)	vehN_intM_S_in_1 (3.24)	vehN_intM_W_out_2 (0.065)
12	vehN_inc_-_gneE10_0 (1.30)	detOcc_gneE12_0 (0.005)	vehN_intM_N_in_1 (3.20)	vehN_intM_W_in_2 (0.064)
13	vehN_inc_gneE12_0 (1.27)	vehN_out_gneE8_0 (0.004)	vehN_intM_S_in_2 (3.14)	vehN_intM_N_out_2 (0.054)
14	vehN_inc_gneE7_2 (1.25)	vehN_out_gneE12_0 (0.004)	vehN_intM_E_in_2 (3.14)	vehN_intM_S_in_2 (0.051)
15	vehN_inc_-_gneE8_0 (1.23)	vehN_out_gneE7_0 (0.004)	vehN_intM_W_in_2 (3.12)	vehN_intM_E_out_1 (0.033)
16	vehN_inc_-_gneE8_1 (1.23)	vehN_out_gneE10_1 (0.004)	vehN_intM_N_in_2 (3.09)	vehN_intM_E_out_2 (0.030)

on out-of-range demand levels, especially in complex scenarios. In contrast, Neural Networks consistently delivered stable performance across both simple and complex scenarios, showing the best generalization among the tested models. Overall, these results indicate that while machine learning models can learn MP's logic under known conditions, their generalization ability varies significantly. Logistic Regression and Neural Networks perform well because MP's decision-making is largely based on linear relationships in vehicle counts, which these models capture effectively—Logistic Regression through its linear formulation, and Neural Networks via their ability to approximate complex functions including linear patterns. In contrast, tree-based models and SVM are more prone to performance degradation, especially as the tested demand levels deviate further from the training range. In the second experiment, we augmented the input feature space with additional variables—such as turning ratios, queue lengths, and signal states—to reflect a more realistic setting where the functional form of MP is unknown and the model must infer it from richer data. The goal was to assess whether machine learning models could still approximate MP effectively, and whether their reliance on vehicle count persisted or shifted when more contextual information became available. As shown in Table II, feature augmentation led to mixed outcomes. On one hand, models like Logistic Regression and Neural Networks remained highly accurate, demonstrating their ability to integrate new inputs without overfitting or losing sight of the core logic. Their performance on both seen and unseen demand levels in the augmented setting was comparable—if not slightly improved—relative to the first experiment, particularly in the Toy Network. For example, Neural Networks maintained strong generalization even in more complex settings, with accuracies often exceeding 0.85 on unseen demand levels. On the other hand, tree-based models continued to show limited generalization capacity, with their performance deteriorating on the Toy Network. In some cases, performance worsened relative to the first experiment, likely due to overfitting on the richer feature space. For instance, Random Forest accuracy on the Toy Network at demand level 0.30 dropped to 27%, matching its performance in the simpler feature setting—indicating that the extra input variables did not help generalization, and may have exacerbated sensitivity to feature-specific patterns. These findings suggest that while augmenting the input space introduces greater realism and model flexibility, it also raises the risk of overfitting for models that rely heavily on discrete splits or margin-based classification. Conversely, models that already aligned closely with MP's underlying logic—namely, Logistic Regression and Neural Networks—were better able to incorporate the new features without compromising generalization. This reinforces the idea that when approximating policies like MP, which are grounded in linear or threshold-based rules, simpler or flexible models that generalize well under structural assumptions may outperform more complex alternatives.

2. Feature Importance Analysis

To further assess whether machine learning models capture the underlying logic of Max Pressure (MP), we performed a feature importance analysis for the second experiment. We selected Logistic Regression (LR) and Neural Networks (NN), both trained on all demand levels, as they consistently achieved the highest prediction accuracy. The goal is to examine whether these models

prioritize the features used by MP—specifically, vehicle counts on incoming and outgoing lanes. As shown in Table III, the feature importance results confirm that both Logistic Regression and Neural Networks rely heavily on vehicle count features, consistent with the core logic of MP, which bases its phase selection primarily on vehicle accumulation at intersections. In the toy network, which contains 16 lanes, the top-ranked features for both LR and NN overwhelmingly correspond to vehicle counts on these lanes, covering all directions (North, South, East, and West) and both incoming and outgoing lanes. This provides strong evidence that both models learned to prioritize the same information that MP uses for decision-making, even in a more complex network setting.These findings validate that Logistic Regression and Neural Networks not only achieve high predictive accuracy but also capture the fundamental operational principles of MP by focusing on vehicle counts—the key input in MP's closed-form formulation. This demonstrates their suitability for approximating MP logic under both simple and more complex traffic network scenarios.

DISCUSSION AND CONCLUSION

This study investigated whether supervised learning can approximate the decision logic of Max Pressure (MP), a traffic signal control strategy, using only observable data. By training models to mimic MP's phase selections, we found that certain models—particularly Logistic Regression and Neural Networks—accurately reproduced MP's behavior and generalized well, even when the feature space was expanded to simulate real-world uncertainty. These models consistently prioritized vehicle counts, aligning with MP's underlying logic. In contrast, tree-based models and SVM struggled to generalize and often overfit to noise. Overall, the findings show that supervised learning not only enables accurate imitation of traffic control decisions but also offers interpretable insights into their guiding principles. This approach could help stakeholders better understand and anticipate the behavior of proprietary systems, fostering more transparent and cooperative traffic management. Future work could extend this framework to real-world traffic data and more complex control strategies (e.g., SCOOT, SCATS, or hybrid systems). Another promising direction is to combine supervised learning with causal inference or reinforcement learning to not only mimic behavior but also understand the trade-offs embedded in the control policy. Additionally, exploring methods to extract symbolic rules or interpretable decision trees from high-performing models may further bridge the gap between black-box learning and actionable insight.

ACKNOWLEDGMENTS

The authors acknowledge the financial support of the EU Horizon Europe Research and Innovation Programme (Grant Agreement No. 101103808 ACUMEN). In addition, Chat-GPT was used to enhance readability, but no sentence was entirely generated by the tool. The authors reviewed and edited all content as needed and take full responsibility for the final publication.

REFERENCES

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232.

J. S. Angarita-Zapata, A. D. Masegosa, I. T. (2019). A taxonomy of traffic forecasting regression problems from a supervised learning perspective. 7(61462-61479).

Lim, S. (2024). Environment mapping based classification for reverse engineering using supervised learning. *Preprints*.

Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wießner, E. (2018). Microscopic traffic simulation using sumo. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582. IEEE.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., and et al. (2011). Scikit-learn: Machine learning in python. Available online: https://scikit-learn.org (accessed on 20 December 2021).

Varaiya, P. (2013). The max-pressure controller for arbitrary networks of signalized intersections. In *Advances in Dynamic Network Modeling in Complex Transportation Systems*, pages 27–66. Springer.