

Multi-Objective Dynamic Data-Driven Electric Vehicle Route Planning

by

W.M.P.G Dhanushka Manjula Roshan

A thesis submitted to
The Faculty of Graduate and Postdoctoral Studies of
The University of Manitoba
in partial fulfillment of the requirements
of the degree of

Master of Science
Individual Interdisciplinary Studies

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada
August 2026

© Copyright 2026 by W.M.P.G Dhanushka Manjula Roshan

Thesis advisors

Prof. Parimala Thulasiraman and Prof. Saman Muthukumarana

Author

W.M.P.G Dhanushka Manjula Roshan

Multi-Objective Dynamic Data-Driven Electric Vehicle Route Planning

Abstract

This thesis proposes a multi-objective, data-driven methodology for EV route planning under dynamic real-world conditions in urban transportation networks. The proposed methodology integrates four stages: data collection, road travel time forecasting, charging station availability prediction, and route optimization. The designed algorithms were validated and evaluated using City of Winnipeg transportation data. In forecasting road segment travel time, six traffic volume prediction methods were evaluated using five-fold cross-validation across 367 monitored road segments. The K-Nearest Neighbors model combined with network centrality features achieved the best performance, obtaining an R^2 of 0.534, a mean absolute error (MAE) of 61.7 vehicles per hour, and a root mean square error (RMSE) of 97.0 vehicles per hour. The results demonstrate that network topology features can improve prediction accuracy when used with suitable models. This model was applied to

estimate traffic volumes on unmonitored road segments, which were then converted to travel times using the Bureau of Public Roads (BPR) function. Monte Carlo simulation was subsequently applied to these travel time estimates to forecast future travel times.

In forecasting EV charging station availability, occupancy data from 86 Winnipeg charging stations were analyzed over a 12-hour prediction horizon. A sequential Long Short-Term Memory model produced the highest accuracy with an R^2 of 0.9504, while a parallel LSTM achieved an R^2 of 0.9344 with substantially lower training time through regional clustering. The SARIMA statistical method showed significantly lower performance. The multi-objective route optimization applies a bio-inspired heuristic to minimize travel time and charging cost while satisfying battery and charging constraints. The heuristic outperformed the A* baseline algorithm in finding routes under static conditions. A dynamic re-planning strategy further improved performance by adapting routes to real-time conditions during travel.

Contents

Abstract	ii
Table of Contents	vii
List of Figures	viii
List of Tables	x
Acknowledgments	xii
1 Introduction	1
1.1 Gaps in the Literature	5
1.2 Goal of the Thesis	5
1.3 Contributions	6
1.4 Scope and Limitations	6
1.5 Thesis Structure	7
2 Literature Review	9
3 Real-Time MOP EV Routing: Proposed Methodology	13
3.1 STAGE 1: Data Collection	15
3.1.1 Road Network Data	15
3.1.2 Historical Traffic Data	16
3.1.3 EV Charging Stations Data	17
3.1.4 EV Charging Stations Availability Data	18
3.2 STAGE 2: Forecasting	19
3.2.1 Road Segment Travel Time Forecasting	19
3.2.2 Forecasting EV Charging Station Availability	20
3.3 STAGE 4 & STAGE 5: Finding Initial Optimal Route & Re-evaluate Optimal Routes	25

4	Finding Initial Optimal Route- Bi-objective Optimization Algorithm	27
4.1	NSGA II Heuristic	33
4.1.1	The Chromosome	33
4.1.2	NSGA-II Step-by-Step Execution	36
4.2	STAGE: 5 - Re-evaluate Optimal Route	48
4.2.1	Incident Model	50
4.2.2	Re-plan Cost	50
4.3	Summary of the Methodology	50
5	Implementation	52
5.1	Overview	54
5.2	Experimental Setup	54
5.3	Data Collection	55
5.3.1	Road Network Data	55
5.3.2	Historical Traffic Data	55
5.3.3	EV Charging Stations Data	56
5.3.4	EV Charging Availability Data	57
5.4	Road Segment Travel Time Forecasting	58
5.4.1	Predict Missing Vehicle Counts for Each Road Segment	58
	Implementation of Volume Prediction Models	59
	Network Feature Set	62
5.4.2	Travel Time Calculation for Each Road Segment	63
5.4.3	Monte Carlo Simulation for Future Travel Time	64
5.5	Forecasting EV Charging Station Availability	65
5.5.1	Implementation of SARIMA	65
5.5.2	Implementation of the LSTM Model and its Paralleization	65
5.6	Find the Initial Route	70
5.7	Re-evaluate Optimal Route	71
5.8	Summary of Implementation	72
6	Results	73
6.1	Road Segment Travel Time Forecasting	73
6.1.1	Methods for Estimating Traffic Volumes Comparison	74
	Impact of Network Features	74
	IDW and Kriging Performance	76
6.1.2	BPR Function Results	77
6.1.3	Future Travel Time: Monte Carlo Results	78
6.1.4	Discussion	79

6.2	Forecasting EV Charging Stations Availability	79
6.2.1	SARIMA Performance	81
6.2.2	Sequential LSTM Performance	82
6.2.3	Parallel LSTM Performance	83
6.2.4	Comparative Analysis	84
6.2.5	Limitations	85
6.3	Route Optimization Results	86
6.3.1	Experimental Routes	86
6.3.2	Initial Optimal Route Results	86
	The Charging Threshold and the Degenerate Fee Objective . .	88
	Convergence Behaviour	89
6.3.3	Re-evaluating Optimal Routing Results	91
6.3.4	Initial vs. Re-Evaluate Optimization	94
6.3.5	Optimal Route Maps and Charging Stations	94
6.3.6	Re-plan Trigger Breakdown	98
7	Discussion & Conclusion	100
7.1	Future Work	102
A	Methods for Estimating Traffic Volumes at Unmonitored Locations	103
A.1	Inverse Distance Weighting (IDW)	103
A.2	Kriging	104
A.3	Gradient Boosting	105
A.4	K-Nearest Neighbors(KNN)	106
B	Description of Road Capacity Standards and the BPR Function	108
B.1	Road Capacity Standards	108
B.2	Bureau of Public Roads (BPR) Function	109
B.3	Level of Service (LOS)	111
C	Description of Forecasting Models for EV Charging Station Avail-	
	ability	112
C.1	SARIMA (Seasonal ARIMA)	112
C.2	LSTM Model	113
C.3	Evaluation Metrics	114
D	Description of the Multi-Objective Optimization Method	116
D.1	Genetic Algorithms	116
D.2	Non-dominated Sorting Genetic Algorithm II (NSGA-II)	117

D.2.1 Non-dominated Sorting and Population Update	119
D.3 Knee-Point Selection	120
Bibliography	130

List of Figures

1.1	Map from decision space to Objective Space	3
1.2	Framework	4
3.1	Methodology	14
3.2	Sample road network as a directed graph	16
3.3	Calculate virtual path for charging stations	17
4.1	Methodology	28
4.2	A candidate route on the road network	35
4.3	A route as a chromosome	35
4.4	The NSGA-II generational loop as implemented for the EV routing problem.	36
4.5	Heuristic route generation	38
4.6	Illustration of the crossover operator. The two parents share a common node d . The offspring inherits the first parent's path up to d (blue) and the second parent's path from d onward (red), producing a single connected route.	44
5.1	System architecture for EV route planning. Gray boxes are external actors (data sources, the vehicle). Teal boxes are system components.	53
5.2	City of Winnipeg road map	55
5.3	City of Winnipeg EV charging stations	57
5.4	Road travel time prediction methodology	58
6.1	The top two methods.	75
6.2	Average predicted charging station occupancy by hour of day and day of week across all Winnipeg stations.	80

6.3	Per-route Pareto fronts (blue), and the A* baseline (red star). Routes 0-1 show a genuine trade-off.	88
6.4	Per-route Pareto fronts (blue), and the A* baseline (red star). Routes 2-4 reduce to a single non-dominated point because every feasible path has zero charging fee.	89
6.5	SoC profiles for the best-time solution. Left: route 0 needs one charging stop (green band). The trace touches the 25% trigger (red dashed line). Right: route 2 completes on the initial charge and never triggers, which is why its fee objective is degenerate.	90
6.6	Convergence of best time, best fee, and Pareto-front size over 50 generations for route 0 (top) and route 1 (bottom).	91
6.7	Replan-trigger composition by route. Intersection triggers (green) dominate over scheduled triggers, while post-charging triggers add a small, constant contribution.	93
6.8	Initial optimal routes vs. re-evaluated optimal routes	93
6.9	Optimal route map for Route 0	95
6.10	Optimal route map for Route 1	96
6.11	Optimal route map for Route 2	97
6.12	Optimal route map for Route 3	97
6.13	Optimal route map for Route 4. As with the other short routes, no charging is required.	98
B.1	BPR function with Level-of-Service bands.	110
D.1	Non-dominated sorting layers on a sample combined population. Each curve is one Pareto front. Deeper fronts are shown in grey.	120

List of Tables

3.1	Example of charging station availability raw data in long format . . .	22
3.2	Example of data after pivoting into wide format	22
4.1	Re-plan triggers	49
5.1	Vehicle-count data sources.	56
5.2	Hyperparameters used in the model.	61
5.3	Feature set for estimating the vehicle count on a road segment.	63
5.4	Sequential LSTM configuration.	66
5.5	Station cluster sizes from K-means clustering on usage patterns ($K = 5$).	68
5.6	Parallel LSTM configuration (applied to each of the five cluster models).	69
5.7	Default parameter values used in initial route experiments.	70
5.8	Default parameter values used in route re-evaluation experiments.	71
6.1	Method comparison: five-fold cross-validation.	74
6.2	Effect of adding network features on cross-validated R^2	75
6.3	Network-wide BPR results: six-year mean (08:00-08:30), 2020-2025	77
6.4	LOS distribution(2020-2025 mean).	78
6.5	SARIMA performance.	81
6.6	Sequential LSTM performance.	82
6.7	Parallel LSTM aggregate performance.	83
6.8	Comparison of all models for connector count prediction (12-hour horizon, 86 stations).	84
6.9	The five evaluated routes	86
6.10	Initial optimal routes results	87
6.11	Re-evaluation execution per route.	92

6.12	Re-plan trigger distribution across all experimental runs. The incident row reflects zero because most experiments ran without incidents enabled.	99
B.1	Per-lane capacities and free-flow speeds used in this study	109

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisors, Prof. Parimala Thulasiram and Prof. Saman Muthukumarana, for their invaluable advice, continuous support, and patience during my Master's study. It was a privilege to work under their supervision.

I would also like to thank my advisory committee members, Prof. Noman Mohammed and Prof. Shaun Lui, for allocating their time to review my research and for their suggestions and comments to improve it.

My gratitude extends to the staff, support staff, and my colleagues in the Department of Computer Science, University of Manitoba.

Last but not least, I would like to thank my family and all the people who have helped me.

Chapter 1

Introduction

Electric vehicles (EVs) have moved from niche technology to a practical transport option for millions of drivers. Global EV stock surpassed 30 million units in 2022, with projections suggesting this figure could exceed 240 million by 2030 [1]. Behind that growth is a harder problem for route-planning systems.

Most vehicle navigation systems are built on shortest-path algorithms such as Dijkstra’s algorithm or A*, which route a vehicle by minimizing distance or travel time over a road network [2; 3]. This is efficient for petrol vehicles because a petrol car’s range anxiety is alleviated by filling up in five minutes at any of the thousands of stations. An EV’s operational constraints, on the other hand, differ from those of a petrol vehicle. The amount of usable energy currently remaining in a battery relative to its maximum capacity (also called state of charge) degrades non-linearly with speed, ambient temperature, and road gradient [4]. For example, on highways,

more power is required to overcome air resistance (aerodynamic drag). EVs would need to consume more battery power to overcome this drag. Furthermore, charging stations vary by connector type, charging rate, and availability. A 40-minute charge becomes a 90-minute wait if the station is occupied. None of this is captured by an algorithm whose only notion of cost is distance, such as Dijkstra’s algorithm, which is an efficient algorithm for single-objective optimization problems.

An optimization problem can be defined mathematically as a maximization or minimization problem with one or more objective functions subject to possible constraints. A single-objective optimization considers only one objective. In Dijkstra’s shortest path algorithm, the objective function is the distance, and the algorithm finds the ”best” minimum path among the many feasible solutions [5]. In the EV routing problem, multiple objectives must be considered, including battery power, charging station costs, and more. Therefore, the routing problem must be formulated as a multi-objective optimization problem (MOP) [6].

The EV or green vehicle routing problem [7] is a subset of the general, well-known vehicle routing problem (VRP). In the literature, the green VRP is considered in the static domain. That is, road conditions, energy consumption, and station availability are fixed at the time of planning. In this thesis, we consider *dynamic* domains that reflect real-time unpredictable traffic, weather, battery power loss during travel prompting drivers to detour in search of charging stations, and other variables. We therefore formulate the problem as a dynamic multi-objective optimization problem

(MOP).

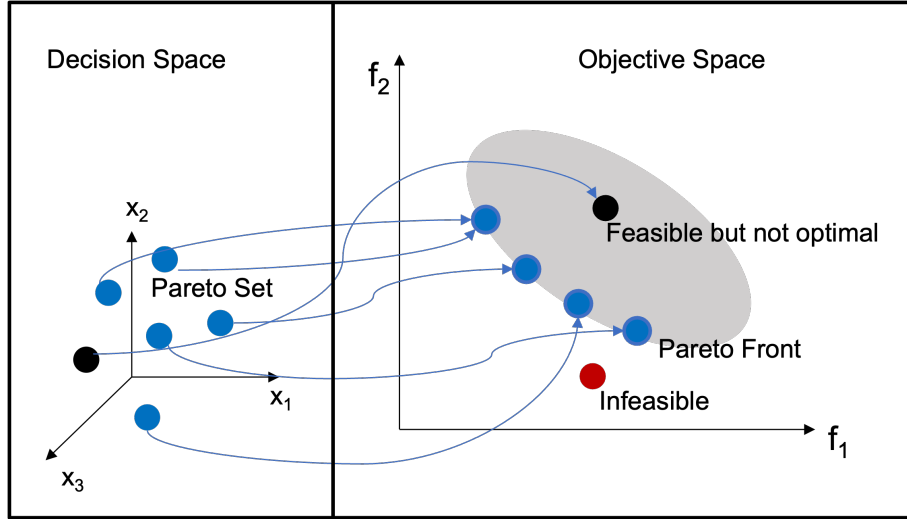


Figure 1.1: Map from decision space to Objective Space

The MOP is challenging because multiple conflicting objectives are present simultaneously. Improving the solutions for one objective may degrade the solutions for other objectives. The output of an MOP algorithm is a set of solutions, called the *Pareto Set*. Figure 1.1 [8] illustrates that every solution in the Pareto set is projected onto a point in the objective space representing a trade-off. Multiple solutions may map to the same point in the objective space, yielding a feasible, infeasible, or optimal solution. The set of optimal or near-optimal solutions represents the *Pareto front*. There is no single optimum to converge to, only a front of equally valid trade-offs, and choosing which point on that front to act on is itself an open problem [9]. Therefore, finding Pareto-front solutions in an MOP is an NP-hard problem [10]. The search space is also larger, since the algorithm has to maintain and compare

many routes rather than refine a single best-guess route. The problem's dynamism under consideration in this thesis also makes it even more challenging. Deterministic algorithms do not suffice. In this thesis, we study heuristics (or approximation algorithms), such as evolutionary algorithms, for solving the MOP.

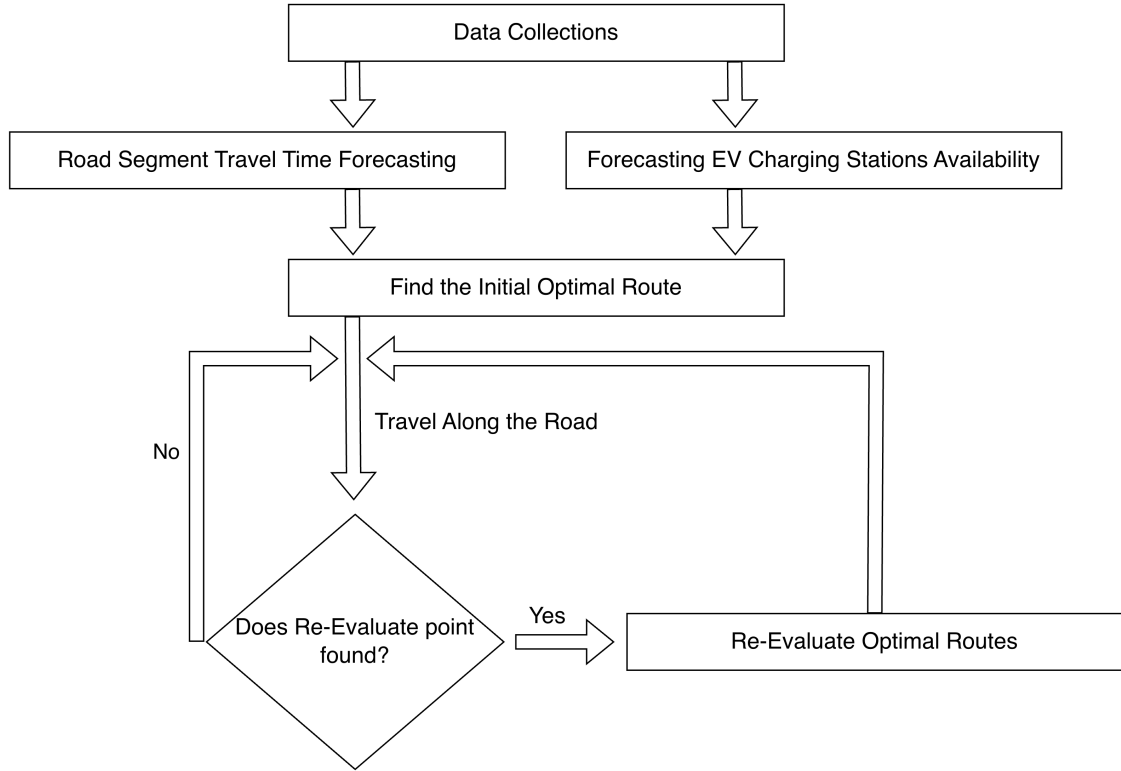


Figure 1.2: Framework

1.1 Gaps in the Literature

There are two gaps in the EV literature that we address in this thesis. First, most existing formulations assume static traffic conditions and fixed charging station availability. These assumptions are rarely valid in practice. Traffic shifts throughout the day, and station occupancy varies continuously. Second, few studies integrate data-driven prediction with multi-objective optimization. Most rely on fixed parameters. This thesis addresses both gaps by developing a framework that combines traffic and occupancy forecasting with Pareto-based evolutionary optimization, producing route recommendations that adapt to changing conditions rather than assuming constant conditions.

Figure 1.2 provides our proposed framework, which will be discussed in detail in future chapters. The thesis follows a data-driven approach. Using real-time data, the proposed algorithm forecasts traffic congestion and EV charging station availability with machine learning techniques to solve the MOP routing problem. The algorithm proposes re-optimizing the routing algorithm on the fly.

1.2 Goal of the Thesis

To study dynamic route planning for electric vehicles with multiple objectives and constraints.

1.3 Contributions

1. Proposed a mathematical formulation of the multi-objective problem with constraints that include battery SoC dynamics, charging-station behavior, and traffic variability.
2. Proposed predictive models to forecast travel time and charging station availability using real-world road network and infrastructure datasets.
3. A parallel-designed technique is proposed to forecast charging station availability.
4. Using the proposed mathematical multi-objective formulation, an approximation-based multi-objective optimization algorithm was designed for route planning under dynamic conditions.
5. Simulated the proposed algorithm on real-world datasets.
6. Compared and evaluated the proposed algorithm against baseline routing approaches using standard multi-objective performance indicators and domain-specific metrics.

1.4 Scope and Limitations

This work focuses on single-vehicle EV routing. Fleet routing and vehicle-to-grid (V2G) interactions are outside the scope; simulations are limited to natural

extensions of the methodology. The methodology is evaluated in simulations without physical vehicles. Road network data are sourced from OpenStreetMap [11], and charging infrastructure data are obtained from publicly available APIs. Simulation introduces a known validity gap. Real-world battery degradation, driver behavior, and hardware-level charging variability are difficult to reproduce fully in software.

Energy consumption modeling draws on established physics-based and data-driven approaches [4; 12]. The model is calibrated on existing datasets and does not account for individual vehicle wear or tire pressure variation. Two further limitations apply. The research does not address privacy concerns arising from continuous location tracking, and it does not consider accessibility requirements for drivers with disabilities.

1.5 Thesis Structure

The remainder of this thesis is organized as follows. Chapter 2 reviews related work on EV route optimization. It summarizes existing results and the optimization algorithms used. Chapter 3 describes the proposed methodology for finding the best routes for EVs by minimizing travel time and charging fees, while accounting for congestion time and charging station availability. Chapter 4 describes the proposed optimization methodology and problem formulation. Chapter 5 presents the implementation of the proposed methodology using real data and techniques used to improve model performance. Chapter 6 presents the implementation results. Finally,

Chapter 7 discusses these results and concludes the thesis.

Chapter 2

Literature Review

The foundation of electric vehicle (EV) routing is commonly traced to the Green Vehicle Routing Problem (G-VRP) introduced by Erdoğan and Miller-Hooks [7]. Their formulation does not require vehicles to recharge en route. Charging stops are inserted only when the battery range would otherwise be exceeded. The model assumes each customer can be served from the depot with at most one station visit, at which point the vehicle charges to full.

Later studies introduced more operationally realistic constraints. The authors in [13] and [14] extended the routing problem to accommodate multiple charging technologies and partial recharging strategies. Early models typically assumed energy consumption scaled linearly with distance, a simplification that [15] later challenged. Their work built on earlier models with non-linear charging and showed that battery drain is highly sensitive to vehicle load. In reality, speed and terrain both affect

power consumption, but their routing model held those constant to isolate the load effect cleanly.

Early EVRP research was mostly about commercial delivery with homogeneous fleets, central depots, and fixed schedules. The authors in [16] extended this to multi-depot setups with time windows. Still, the core assumptions stayed the same, including uniform charging speeds, strict delivery windows, and distance as the variable to minimize. *This is efficient for logistical applications such as supply and demand.* However, for personal EV routing, where vehicles charge at different rates, electricity prices shift throughout the day, and drivers have their own preferences about acceptable detours.

A persistent issue in EV routing is the trade-off between travel time and charging cost. Fast charging reduces journey time but is more expensive. Slower charging is cheaper but extends total duration. In [17], the authors examined this trade-off in hybrid charging systems and found that reducing charging time can substantially increase overall cost. This result complicates any single-objective formulation.

In [17], the authors approached the problem with a weighted objective function that trades off time against cost based on user preferences. To handle real-world uncertainty, such as shifting traffic and fluctuating electricity prices, they added Model Predictive Control, which updates routing decisions on the fly rather than locking in a fixed plan. In [18], the authors proposed a multi-objective MINLP model for personal EV routing that trades off trip time against charging costs. The model

considers the fact that neither variable is fixed. Local electricity rates determine the actual cost of a given route, and charger type matters because L1, L2, and L3 stations recharge batteries at very different rates. Using the ϵ -constraint method alongside metaheuristic techniques, the model generates a Pareto front rather than a single answer, allowing drivers to choose whichever solution fits their tolerance for spending more time or money.

Traffic congestion and station availability add further uncertainty. In [19], the authors introduced a routing system that predicts wait times at charging stations by having vehicles share their planned arrival times, rather than relying on historical occupancy data. Battery constraints are treated as non-negotiable, and any route that would drain the charge to zero is ruled out entirely.

Two charging approaches appear in the literature. Wired stations, where vehicles stop and plug in, and wireless charging roads, which transfer energy while the vehicle is moving. Wireless charging cuts travel time but costs more to build and run. This thesis uses only wired stations, which reflects what is currently available in practice.

The EV routing problem is NP-hard, which limits the applicability of exact methods to small problem instances. The authors in [17] formulated their problem as a Mixed Integer Linear Program (MILP) and applied linearization techniques to manage non-linear constraints. In [20], the authors studied a variant of the traveling salesperson problem where EVs recharge while driving on electrified roads. Exact methods are inefficient on larger networks, often failing to find any feasible solution

within reasonable time. Their fix was a network-reduction heuristic that compresses the road network into a smaller multigraph and checks only a subset of paths between customers, thereby significantly reducing computation without straying far from the optimal solution.

In this thesis, we study evolutionary heuristics. Evolutionary algorithms are particularly well-suited to multi-objective EV routing because they maintain a population of candidate solutions and can approximate the full Pareto front in a single run.

In [21], the authors showed that co-evolutionary techniques, in which multiple sub-populations evolve simultaneously and interact, yield greater solution diversity and faster convergence than standard evolutionary approaches for problems with conflicting objectives and complex search spaces.

There are two gaps that we address in this thesis. First, most existing formulations assume static traffic conditions and fixed charging station availability. These assumptions are rarely valid in practice. Traffic shifts throughout the day, and station occupancy varies continuously. Second, few studies integrate data-driven prediction with multi-objective optimization. Most rely on fixed parameters. This thesis addresses both gaps by developing a methodology that combines traffic and occupancy forecasting with Pareto-based evolutionary optimization to produce route recommendations that adapt to changing conditions.

Chapter 3

Real-Time MOP EV Routing: Proposed Methodology

EV routing with multiple objectives is challenging. In this work, we consider two objectives: travel time and charging costs. Neither objective can be minimized in isolation. The shortest route may carry the highest charging cost, and the cheapest charging path may add considerable time. We propose a data-driven approach that uses real-time data to find a route that optimizes the objectives, producing a set of Pareto-optimal routes for the driver to choose from.

There are four stages to our methodology: data collection, road congestion forecasting, charging station availability prediction, and bi-objective route optimization. Figure 4.1 illustrates how these stages connect.

The methodology begins with data collection from public and private sources.

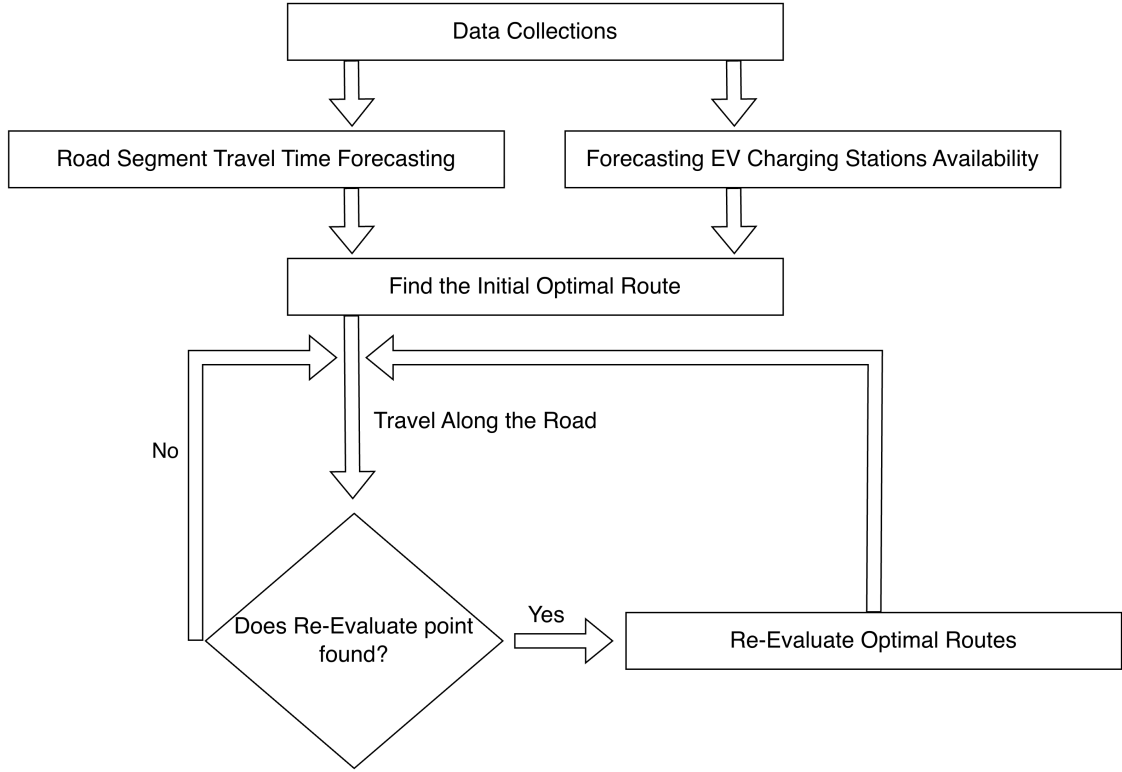


Figure 3.1: Methodology

That data feeds two forecasting stages: one forecasts travel times across the road network, and the other forecasts EV charging station availability across a city. The prediction techniques used for forecasting will be explained in later sections. Note that in the experiments, we use data obtained from the City of Winnipeg, but our methodology applies to data from any city.

The next stage of optimization, the bi-objective optimization algorithm, uses the forecast travel times and charging station availability predicted in the previous stages to find the initial optimal route. The vehicle travels this route, passing several re-

evaluation points (as defined in Table 4.1 and elaborated in later sections). At each one of these points, the algorithm re-evaluates the optimal route against the latest forecasting data. The vehicle switches to a new route only if the re-evaluation finds a better route. Otherwise, the vehicle travels on the existing route.

3.1 STAGE 1: Data Collection

The first stage is data collection. Four types of data were collected for this research. These are:

1. Road network data for the selected geographical area
2. Historical traffic data
3. EV charging stations data
4. EV charging stations availability data

3.1.1 Road Network Data

This research uses datasets from public and private sources. Road network data was downloaded from OpenStreetMap (OSM) [11], an open-access map database that covers road geometry, lane configurations, speed limits, and network connectivity for the selected geographical area.

After collecting data from the road network, the road system is modeled as a directed graph as shown in Figure 3.2, where road segments are edges and road

connections (intersections) are nodes. The edge direction is derived from the travel direction in each segment, allowing the graph to represent both one-way and two-way traffic flows. Each edge carries four attributes describing the corresponding road segment. These are the *maximum speed*, *number of lanes*, *road type*, and *segment length*.

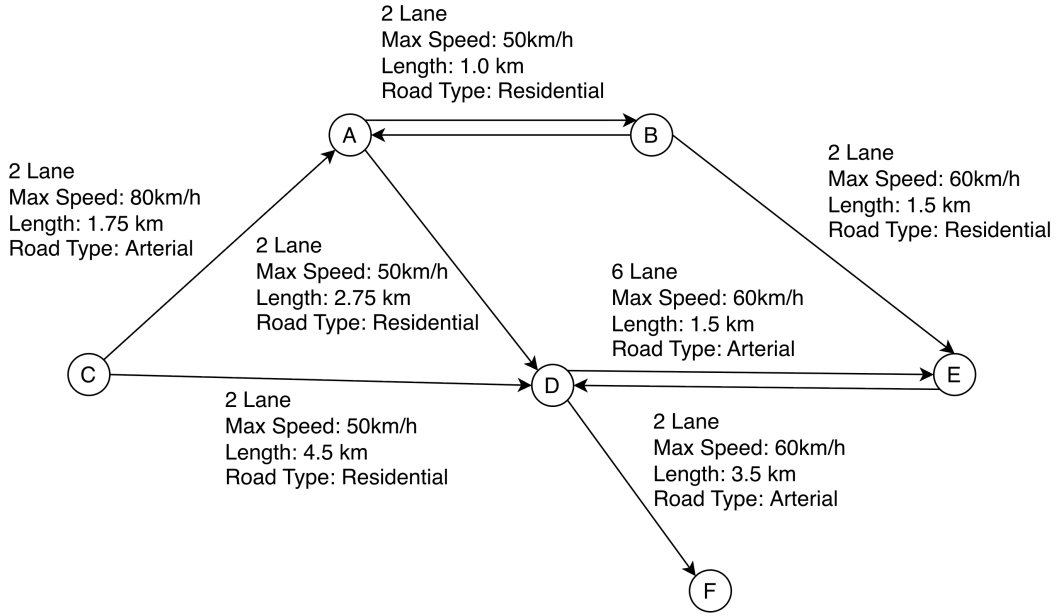


Figure 3.2: Sample road network as a directed graph

3.1.2 Historical Traffic Data

Historical traffic data was collected from public and private data sources. The traffic data contains travel times for all road segments within the selected time frame and geographic area.

3.1.3 EV Charging Stations Data

EV charging infrastructure data were extracted from OpenChargeMap [22] an open global EV charging database. In this thesis, EV charging station fees were extracted from the ChargePoint [23] website, which is a large electric vehicle (EV) charging network and hardware company that provides EV charging station locations online.

Charging stations are incorporated into the road network as *virtual* nodes using perpendicular projection onto the nearest road segment as shown in Figure 3.3. For each station, the projection point P is computed as the orthogonal foot from the station onto the segment defined by nodes A and B. Two virtual edges are then added connecting P to A and P to B, with distances derived from the projection geometry.

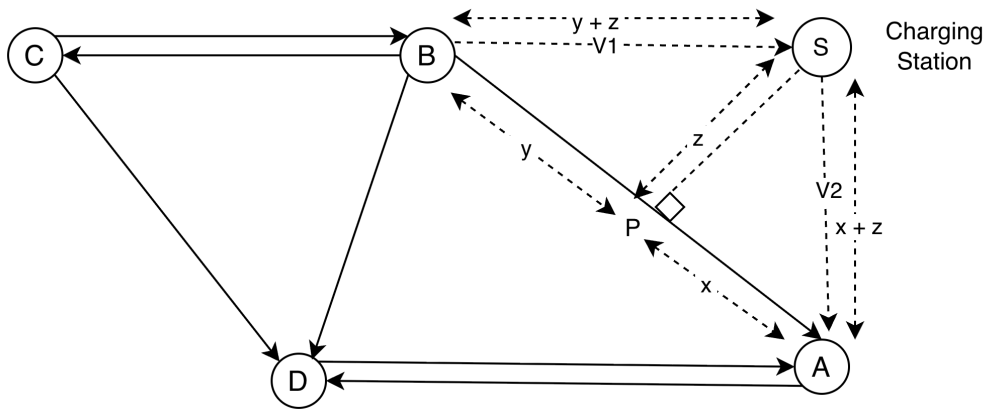


Figure 3.3: Calculate virtual path for charging stations

In Figure 3.3, V_1 and V_2 are virtual paths to the charging station. $y + z$ is the distance from node B to the charging station, and $x + z$ is the distance from node A to the charging station.

3.1.4 EV Charging Stations Availability Data

Real-time availability data for EV charging stations were extracted using HERE Technologies [24], an online platform that provides live maps for drivers, and Charge-Point [23]. For this thesis, six fields were extracted from the collected data:

1. Location ID
2. Timestamp
3. Number of available connectors
4. Number of installed connectors
5. Number of connectors planned for future installation
6. Number of offline connectors

3.2 STAGE 2: Forecasting

3.2.1 Road Segment Travel Time Forecasting

In this thesis, the Monte Carlo simulation technique[25] is used to forecast future travel times for each road segment. The travel time variability is modeled using a Poisson distribution (Equation 3.1):

$$n_i \sim \text{Poisson}(\lambda_i), \quad \lambda_i = \mu_{\text{now},i}, \quad i = 1, \dots, N, \quad (3.1)$$

where n_i is the simulated travel time of segment i , N is the number of segments, and λ_i is the Poisson rate parameter, set equal to the current mean travel time $\mu_{\text{now},i}$. The rate is estimated by maximum likelihood as

$$\hat{\lambda}_i = \frac{1}{M_i} \sum_{j=1}^{M_i} x_{i,j}, \quad (3.2)$$

where $x_{i,j}$ denotes the j -th historical travel time observation recorded for segment i , and M_i is the number of such observations available for that segment within the corresponding time-of-day interval. Algorithm 1 presents the steps of the Monte Carlo simulation, where K denotes the number of replications.

Algorithm 1 Monte Carlo forecasting of travel times for each road segment

```

1: for  $i = 1$  to  $N$  do
2:    $\hat{\lambda}_i \leftarrow \frac{1}{M_i} \sum_{j=1}^{M_i} x_{i,j}$ 
3: end for
4: for  $k = 1$  to  $K$  do
5:   for  $i = 1$  to  $N$  do
6:     draw  $n_i^{(k)} \sim \text{Poisson}(\hat{\lambda}_i)$ 
7:   end for
8:    $T^{(k)} \leftarrow \sum_{i \in \mathcal{P}} n_i^{(k)}$ 
9: end for

```

There are two reasons for using the Monte Carlo technique. First, the technique produces a distribution of possible travel times instead of a single predicted value. This is important because charging station availability depends on how much travel time actually varies, not just its average. Second, every output can be traced back to a sampling step and a distributional assumption, which makes it easier to analyze. For each segment, the mean travel time is extracted and used as the representative travel time in the optimization stage.

3.2.2 Forecasting EV Charging Station Availability

A charging station that appears on the map may be occupied, offline, or at capacity by the time a vehicle arrives, turning a valid route into a stranded one.

Predicting station availability in advance is therefore important.

The objective is to forecast the exact number of available EV charging connectors at each station. This enables route optimization, avoiding sending vehicles to occupied chargers.

I. Preprocessing Data: First, the data is pre-processed. The steps are as follows.

1. **Filtering:** In this step, records with zero total connectors are removed. These represent decommissioned or placeholder stations that carry no availability data.
2. **Removing always-zero stations:** Stations where the available connector count is zero at every timestamp are dropped, since a constant series provides no signal for a predictive model.
3. **Pivoting:** The long-format records are pivoted into a wide matrix with timestamps as rows and station identifiers as columns. Each raw record contains six fields: location ID, timestamp, and the number of available, installed, planned, and offline connectors. Since the goal is to predict charging station availability, the available connector count is selected as the primary field during pivoting. Table 3.1 shows an excerpt of the raw long-format data, and Table 3.2 shows the corresponding wide-format matrix, where each timestamp’s available connector counts across all stations are combined into a single row. This restructuring is necessary because sequence models operate on one feature vector

per timestamp, capturing the state of all stations simultaneously.

Table 3.1: Example of charging station availability raw data in long format

Station ID	Timestamp	Available	Installed	Planned	Offline
S_001	2025-05-01 00:00	3	5	0	1
S_002	2025-05-01 00:00	0	2	0	0
S_001	2025-05-01 00:15	2	5	0	2
S_002	2025-05-01 00:15	1	2	0	0

Table 3.2: Example of data after pivoting into wide format

Timestamp	S_001	S_002
2025-05-01 00:00	3	0
2025-05-01 00:15	2	1

4. **Feature engineering:** Three temporal features are extracted from each timestamp: hour of day (0-23), day of week (0 = Monday, 6 = Sunday), and month. The resulting feature matrix has dimensionality $T \times (S + 3)$, where T is the number of timestamps and S is the number of charging stations.
5. **Scaling:** All features are normalized to $[0, 1]$ using a Min-Max scalar fitted on the training partition only. Fitting the scalar on the full dataset before

splitting would allow test-set statistics to influence training, which constitutes data leakage.

6. **Sliding-window sequences.** Input-output pairs are constructed using a sliding window. Each input is a sequence of $L = 96$ consecutive time steps (24 hours at 15-minute resolution), and the corresponding target is the availability values for the following $H = 48$ time steps (12 hours). Both model variants use the same input length and forecast horizon.

The dataset is then split chronologically into three non-overlapping partitions (training set, validation set, and test set).

- **Training set (70%):** Used for gradient updates during model training.
- **Validation set (15%):** Used for early stopping. Training halts when validation loss does not improve for a specified number of consecutive epochs.
- **Test set (15%):** Held out entirely and used only for final evaluation. The model sees no test data during training or hyperparameter selection.

In time-series forecasting with sliding-window sequences, a naive chronological split can introduce data leakage. The targets used to train the model may overlap with the input sequences drawn from the validation or test period. To prevent this, a temporal gap equal to the forecast horizon H is inserted between each pair of partitions, ensuring that no training target coincides with a validation or test input.

$$\text{gap} = L + H - 1 = 96 + 48 - 1 = 143 \text{ sequences} \quad (3.3)$$

No row used as a training target appears in any validation input, training target, or test partition, and the same separation is enforced between the validation and test partitions. The Min-Max scalar is fitted on the training rows only, so the scaling parameters are determined without access to any future observations.

The SARIMA(Appendix C.1) and LSTM (Appendix C.2) machine learning models are used for training.

II. Machine Learning Models for Forecasting A separate SARIMA model is fitted for each station, since SARIMA does not support multivariate inputs. The forecast horizon is set to 12 hours. Parameter search bounds are selected to limit the search space while covering the orders most commonly identified from occupancy data.

The LSTM model is trained on all stations simultaneously. Shared hidden states allow the model to learn correlations between stations, but the same shared representation must also produce a high-dimensional output mapping across all stations and 48 forecast steps.

Training Procedure The LSTM models are trained with the Adam optimizer [26] at a learning rate of 0.001, using Mean Squared Error as the loss function:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N \cdot H \cdot S} \sum_{n=1}^N \sum_{h=1}^H \sum_{s=1}^S (y_{n,h,s} - \hat{y}_{n,h,s})^2 \quad (3.4)$$

where N is the batch size, $y_{n,h,s}$ is the true available connector count, and $\hat{y}_{n,h,s}$ is the predicted count for station s at future time step h .

Early stopping is applied with a patience of 5 epochs, monitored on validation loss. Training halts when validation loss fails to improve by at least $\delta = 10^{-5}$ for five consecutive epochs. The test set is held out entirely and plays no role during training or model selection.

After prediction, raw model outputs are inverse-scaled using the Min-Max scaler fitted on the training partition, rounded to the nearest integer, and clipped to $[0, C_s]$, where C_s is the total number of installed connectors at station s . This ensures predictions are integer-valued and fall within the physically valid range for each station. Model performance is evaluated on the held-out test set using five metrics (Appendix C.3). All metrics are computed on predictions of the number of available connectors after inverse scaling, rounding, and clipping.

3.3 STAGE 4 & STAGE 5: Finding Initial Optimal Route & Re-evaluate Optimal Routes

After forecasting the charging station availability and traffic in the road network, we pass this data to the optimization stage to find the initial route. The optimization

heuristic used in finding the optimal route is described in Chapter 4. This is followed by the description of the reevaluation step in Chapter 4, Section 4.2.

Chapter 4

Finding Initial Optimal Route- Bi-objective Optimization Algorithm

This chapter discusses the bi-objective optimization algorithm for finding optimal routes in Figure 4.1.

In this thesis, we consider two objectives for route planning. These objectives conflict with each other. Minimizing travel time favors faster roads and higher-power chargers, while minimizing charging cost favors slower, cheaper alternatives. No single route dominates on both simultaneously. A single-objective algorithm returns one route, optimized for either time or cost but not both, and gives the driver no basis for trading one against the other.

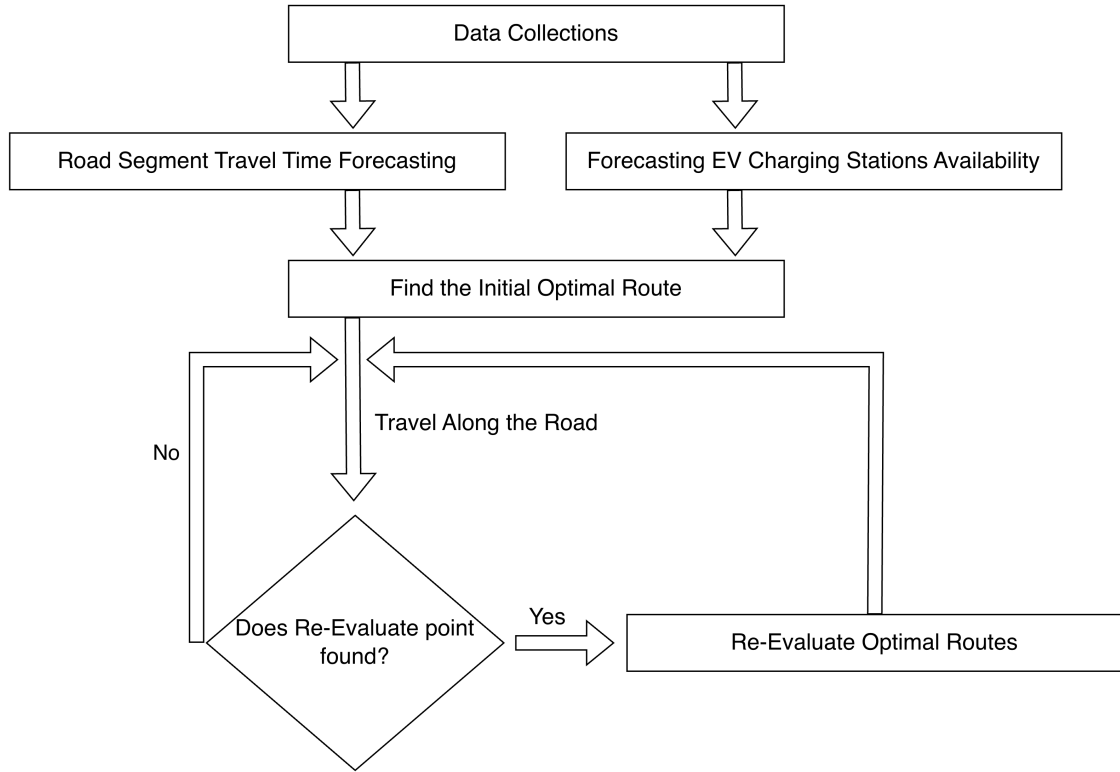


Figure 4.1: Methodology

The bi-objective optimization problem is represented as a minimization problem as follows. Let a candidate EV route be denoted r . The optimization problem is then:

$$\min_{\mathbf{x}} \quad (T_{\text{total}}(r), C_{\text{charge}}(r)) \quad (4.1)$$

$$\text{s.t.} \quad \text{Traffic and routing constraints} \quad (4.2)$$

$$\text{Battery energy constraints} \quad (4.3)$$

$$\text{Charging station constraints} \quad (4.4)$$

where $T_{\text{total}}(r)$ is the total travel time of route r estimated from the Monte Carlo technique [25], and $C_{\text{charge}}(r)$ is the total charging fee incurred along the route.

The two objective functions are the *total travel time* and the *total charging fee*.

Objective Functions:

- The total travel time comprises time spent on road segments and time spent charging. The total travel time is defined as [17]:

$$T_{\text{total}} = \sum_{(i,j) \in A} t_{ij}^r + \sum_{i \in N} t_i^z \quad (4.5)$$

where A is the set of directed road segments and N is the set of nodes. Road travel time t_{ij}^r consists of free-flow time and congestion delay:

$$t_{ij}^r = t_{ij}^0 + t_{ij}^w, \quad t_{ij}^0 = \frac{l_{ij}}{v_{ij}} \quad (4.6)$$

where l_{ij} is the length of segment (i, j) , v_{ij} is the average travel speed, and t_{ij}^w is the congestion time. Charging time at station i is:

$$t_i^z = \frac{c_i C}{L_i} \quad (4.7)$$

where c_i is the fraction of the battery charged at station i , C is the total battery capacity, and L_i is the charging power of the station.

- The total charging cost is the sum of fees charged at each station visited along the route [17]. This is defined as:

$$C_{\text{charge}} = \sum_{i \in N} \lambda_i P_i \eta_i u_i \quad (4.8)$$

where λ_i is the charging duration at station i , P_i is the charging power, η_i is the charging efficiency, and u_i is the electricity price per unit energy. Price and power vary across stations and connector types, so the formulation captures the cost difference between fast charging and slower alternatives rather than treating all stations as equivalent.

Constraints:

- Traffic and Routing Constraints:

A binary decision variable x_{ij} indicates whether segment (i, j) is included in the route:

$$x_{ij} = \begin{cases} 1 & \text{if road segment } (i, j) \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \quad \forall (i, j) \in A \quad (4.9)$$

Flow conservation is imposed at each node to ensure continuity of the selected route from the origin to the destination[17]:

$$\sum_{j|(i,j) \in A} x_{ij} - \sum_{j|(j,i) \in A} x_{ji} = \begin{cases} 1 & \text{if } i \text{ is the origin} \\ -1 & \text{if } i \text{ is the destination} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in N \quad (4.10)$$

At the origin, one unit of flow leaves the node, while at the destination, one unit of flow enters the node. At every intermediate node, incoming flow equals outgoing flow, maintaining continuity along the selected route. Thus, this constraint ensures that the selected segments provide a feasible flow from the origin to the destination.

- Battery Energy Constraints:

The battery state of charge (SoC) at node j is updated from the SoC at node i by subtracting energy consumed on the connecting segment and adding any energy gained through charging at node i [17]:

$$c_j = c_i - x_{ij} l_{ij} K + \lambda_i z_i P_i \eta_i \quad \forall (i, j) \in A \quad (4.11)$$

where K is the energy consumption rate per unit distance, l_{ij} is the segment length, λ_i is the charging duration at node i , z_i is a binary variable indicating whether charging occurs at node i , and the product $\lambda_i z_i P_i \eta_i$ is the net energy added. The SoC is constrained within defined bounds at every node:

$$c^{\min} \leq c_i \leq c^{\max} \quad (4.12)$$

The lower bound $c^{\min}$ prevents the battery from reaching full depletion, which would leave the vehicle immobilized. The upper bound $c^{\max}$ prevents over-charging.

- Charging Station Constraints:

The number of vehicles charging simultaneously at station i cannot exceed its port capacity:

$$\sum_{v=1}^V x_{i,v}(t) \leq s_i \quad \forall i, t \quad (4.13)$$

where $x_{i,v}(t)$ is a binary variable indicating whether vehicle v is charging at station i at time t , and s_i is the number of available charging ports at station i . The constraint prevents more vehicles from charging simultaneously than the station has ports to accommodate.

The bi-objective optimization problem for finding routes is solved using a heuristic. Although any heuristic can be used, we use Non-dominated Sorting Genetic

Algorithm II (NSGA-II) (Appendix D.2) [27], as it has proven efficient for solving multi-objective routing problems [21].

4.1 NSGA II Heuristic

NSGA-II is the most widely used algorithm for multi-objective optimization and is well suited to this application for two main reasons. First, it does not require the objectives to be combined into a single weighted score. Instead, it produces a set of Pareto-optimal solutions representing different trade-offs between conflicting objectives. Second, its fast non-dominated sorting and crowding distance mechanisms maintain a diverse spread of solutions across the Pareto front.

NSGA-II is a genetic algorithm. It maintains a population of candidate routes. It improves this population over successive generations by following the principles of natural selection. Routes with higher fitness are more likely to reproduce. Their features are recombined and then mutated to produce new routes. The fittest routes survive into the next generation. The rest of this section steps through each mechanism in sequence. Selection comes first, crossover follows, and mutation ends the sequence, each one specified for the routing problem addressed here.

4.1.1 The Chromosome

In a genetic algorithm (Appendix D.1), a chromosome is the data structure that encodes a single candidate solution. The first design decision concerns how to rep-

resent a candidate route as a chromosome, a data structure the algorithm can copy, cut, and recombine. In this research, we use a chromosome as the ordered list of road-network nodes the vehicle visits, from the start node to the destination.

Formally, a chromosome is defined as a sequence:

$$\pi = (v_0, v_1, v_2, \dots, v_k), \quad v_0 = \text{start}, \quad v_k = \text{destination} \quad (4.14)$$

where each v_i is the identifier of a road node, and every consecutive pair (v_i, v_{i+1}) is a real road segment that exists in the network. This representation has two features.

- **Variable length.** Different routes visit different numbers of nodes. Therefore, a chromosome is not a fixed-length vector of numbers.
- **Charging stops are not stored in the chromosome.** The chromosome records only the road nodes driven through. Whether, where, and for how long the vehicle charges is decided later, when the route is evaluated (Step3). As a result, the search stays focused on the geography of the route, while the same route remains free to adjust its charging behavior whenever conditions such as time of day, prices, or station queues shift.

Figure 4.2 illustrates a single route over a small road network. Figure 4.3 shows the same route as a chromosome, with one *gene* per visited node, listed in the order the vehicle travels them. The charging station appears as a virtual node attached to both ends of its nearest road segment.

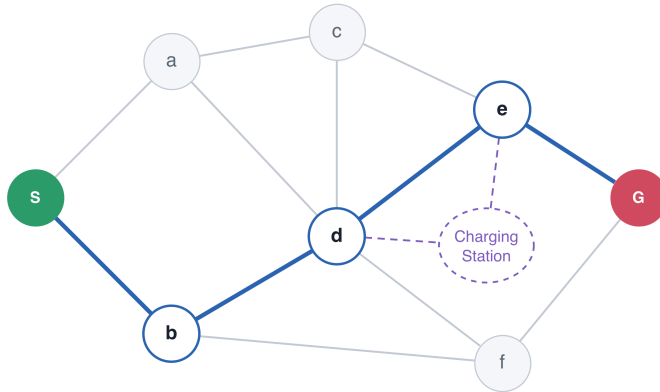


Figure 4.2: A candidate route on the road network



Figure 4.3: A route as a chromosome

In Figure 4.3, the numbers within the chromosome are the identifiers of road junctions. The road network is represented as a graph, in which each junction is a node with a unique identifying number, and each road connecting two junctions is an edge between the corresponding nodes. Since a chromosome is defined as the ordered list of junctions a route passes through, each gene stores the identifier of a single junction, and the road segment traveled between two consecutive genes corresponds to the edge joining that pair of nodes. These identifiers are used because they provide each location in the city with a single, unambiguous label that the algorithm can copy and recombine.

4.1.2 NSGA-II Step-by-Step Execution

The optimizer repeats a fixed cycle. It creates a population, scores it, selects parents, breeds offspring, then keeps the best individuals. Figure 4.4 shows the full loop. The subsections that follow address each stage as it applies to the routing problem, while the generic version of each step is described in Appendix D.1.

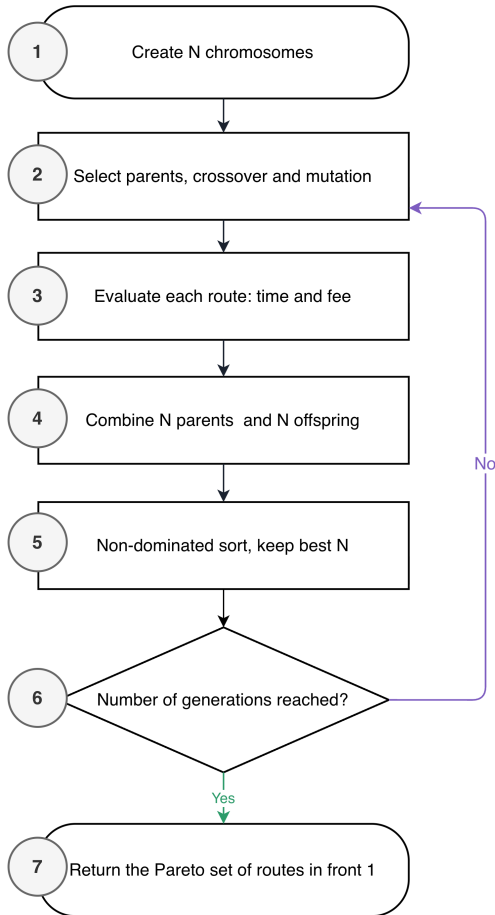


Figure 4.4: The NSGA-II generational loop as implemented for the EV routing problem.

1. Create N chromosomes (Figure 4.4)

The algorithm begins by generating an initial population of N candidate routes. Each candidate route is encoded as a chromosome, which is an ordered list of the road nodes the route traverses, from the start to the destination. If the initial population is too similar to one another, the algorithm cannot explore a sufficiently diverse range of possibilities. Therefore, to create the initial population, two types of methods are used.

The first is to create 40% of paths randomly from the start to the destination without considering travel time or charging stations. This method verifies that the path is connected between the start and the destination and that there is no loop in the path. The second method generates routes heuristically, producing 60% of candidate routes that account for EV station prices, distance to the destination, and travel time. This heuristic generation procedure is described in the section below.

Heuristic Route generation

Route generation follows a heuristic path-building procedure, illustrated in Figure 4.5. The figure illustrates how one candidate route is built from scratch. The overall procedure is described below.

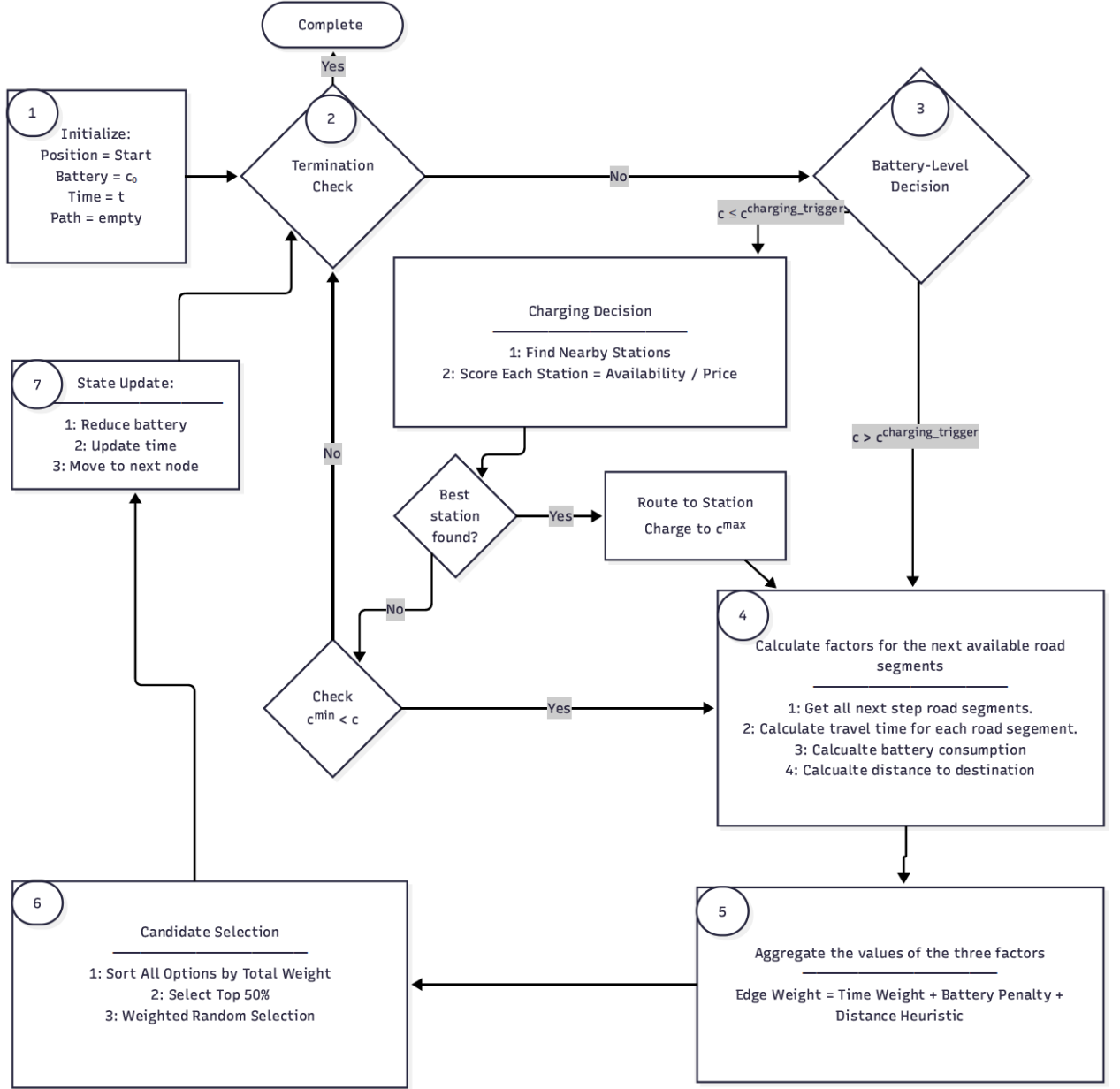


Figure 4.5: Heuristic route generation

(a) Step 1- Initialize: The vehicle's position is set to the origin, the battery

charge is set to its initial value c_0 , the time is set to t , and the traversed path is set to the empty set.

- (b) Step 2- Termination Check: *If* the destination has been reached, the traversed path is returned as the completed route. Go to Step 1. *Else*, the algorithm assesses the vehicle's current battery level, which subsequently determines the operating mode for the current iteration. Go to Step 3.
- (c) Step 3 - Battery-Level Decision: If the battery level exceeds the charging trigger threshold ($c^{\text{charging-trigger}}$), proceed to Step 4. Otherwise, identify candidate charging stations within the surrounding area of the vehicle's current position. The charging trigger threshold is a configuration parameter that determines when the search for a charging station is initiated. It must be greater than the minimum battery level $c^{\min}$.

Each candidate station is evaluated according to a scoring function defined as the ratio of its reported availability to its price, as shown in Equation 4.15:

$$\text{score} = \frac{\text{availability}}{\text{price}} \quad (4.15)$$

This formulation prioritizes stations that offer a higher likelihood of connector availability and a lower charging cost. If a suitable station is identified, the vehicle is routed to that station and charged to the maximum battery capacity, $c^{\max}$, and goes to Step 4. If a suitable station cannot be

found within the search radius and the current battery capacity is greater than the minimum battery capacity $c_{\min}$, the procedure reverts to the normal navigation routine (Step 4). This allows the vehicle to continue toward the destination despite the absence of an available charging option. If a suitable station cannot be found and the current battery capacity is less than the minimum battery capacity, the process proceeds to Step 2 and the current path is terminated.

(d) Step 4 - Calculate factors for the next available road segments:

Determine the subsequent road segment to traverse through a multi-factor scoring procedure applied to the set of outgoing edges from the current node. This procedure considers three factors: traffic conditions, battery consumption, and distance towards the destination. After computing these factors, go to Step 5.

- Factor 1: Traffic

For each candidate edge, the current travel time is calculated and normalized to produce a time-based weighting term, shown as Equation 4.16:

$$\text{Time weight} = \frac{\text{Travel time}}{60} \quad (4.16)$$

As a result, edges with longer travel times receive a higher penalty, consistent with current traffic conditions.

- Factor 2: Battery Impact

The energy required to traverse a given edge is calculated as the product of the edge's distance and the vehicle's consumption rate. Based on this, the projected state of charge (SoC) is determined, and a corresponding penalty is applied according to the following rule, shown as Equation 4.17:

$$\text{Battery penalty} = \begin{cases} 1000 & \text{if the resulting SoC is critical} \\ 100 & \text{if the resulting SoC is low} \\ 0 & \text{otherwise} \end{cases} \quad (4.17)$$

As a result, edges leading to a critically low state of charge are avoided, maintaining that routing decisions remain battery-feasible.

- Factor 3: Distance-to-Destination

The remaining distance to the destination is combined with the candidate edge's length to generate a destination-oriented heuristic term, computed as follows, shown as Equation 4.18:

$$\text{Distance heuristic} = \frac{\text{Edge length}}{\text{Distance to goal}} \quad (4.18)$$

As a result, edges that reduce the remaining distance to the destination are prioritized.

- (e) Step 5- Aggregate the values of the three factors The three factors described above are aggregated into a single overall weight for each candidate edge.

This formulation 4.19 enables the algorithm to account for traffic conditions, battery feasibility, and goal-directed progress within a unified score. shown as Equation 4.19:

$$\text{Edge weight} = \text{Time weight} + \text{Battery penalty} + \text{Distance heuristic} \quad (4.19)$$

- (f) Step 6 - Candidate Selection: In this step, the next road segment is selected as follows. Following the computation of overall edge weights, all candidate edges are ranked in ascending order of weight, and the top 50% of ranked candidates are retained for further consideration. Rather than strictly selecting the single lowest-weighted edge, the algorithm performs a weighted random selection among the top candidates, such that edges with stronger scores are assigned a proportionally higher probability of selection.
- (g) Step 7- State Update: Upon selection of an edge, the vehicle's state is updated accordingly. The battery level is decremented in proportion to the estimated energy consumption, the total time is incremented, and the vehicle's position is advanced to the corresponding next node. The methodology is then returned to the main loop, where the destination and battery-level conditions are re-evaluated for the subsequent iteration.

2. Select parents, crossover, and mutation (Figure 4.4)

This step generates N new candidate routes from the routes the algorithm already maintains. In the terminology of genetic algorithms, the existing routes used to produce new ones are called parents, and each new route produced from them is called an offspring (or child). At the start of every generation, the current population of N routes serves as the set of parents, and this block produces N offspring from them. Generating new routes is essential because they are the only way the search reaches parts of the map the current population does not cover. Without them, the population would stay frozen from one generation to the next.

This step operates in three stages. The first stage is selection, which decides which parents get to reproduce. Two routes are picked from the population at random. If one of them dominates the other, that route is chosen. Dominating means being at least as good on both objectives, time and fee, and strictly better on at least one, as formalized in step 5 (Figure 4.4). If neither route dominates the other, a random route is selected.

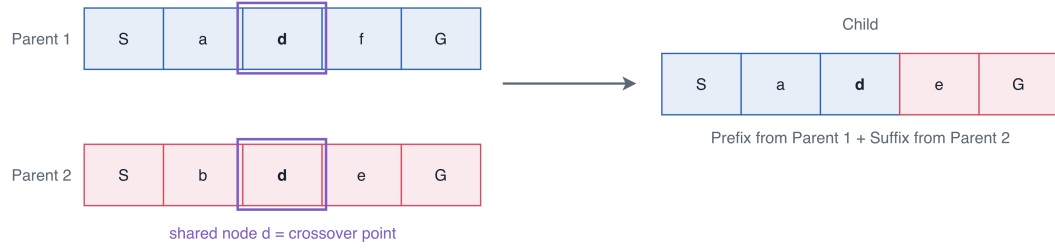


Figure 4.6: Illustration of the crossover operator. The two parents share a common node d . The offspring inherits the first parent's path up to d (blue) and the second parent's path from d onward (red), producing a single connected route.

The second stage is crossover, which combines two selected parents. If both parents pass through a common node, the offspring is formed by following the first parent's path up to that node, then the second parent's path from that node onward, producing a single connected route that inherits part of each parent (Figure 4.6). The third stage is mutation, a small random modification applied to an offspring with a given probability, in which a randomly chosen intermediate segment of the route is replaced by the shortest path across that segment. This allows the search to discover road segments that neither parent used. After these three stages, only valid, start-to-destination routes are selected.

3. Evaluate each route: time and fee (Figure 4.4)

Once new routes have been generated, the algorithm evaluates how good each one is. Every route is assigned two scores: the total trip time in minutes,

including both driving and charging, and the total charging cost in dollars. To compute these scores, the algorithm traverses each route node by node, accumulating distance, time, cost, and remaining battery level. Driving times are not fixed values. They are taken from the travel-time forecasting model in which each road segment has a travel time that varies with time of day. This stage also determines charging behavior. Whenever the battery level falls below a specified threshold, the algorithm inserts a stop at the nearest suitable charging station, recharges the battery, and adds the resulting time and cost to the two scores. Therefore, charging decisions never need to be encoded in the chromosome. *Note that a route that proves infeasible is not discarded immediately, and it is assigned a large penalty cost.* This gives the route a brief chance to survive, in case part of it is reusable. In contrast, the penalty guarantees it is eventually eliminated.

4. **Combine N parents and N offspring**(Figure 4.4)

Having scored the N offspring, the algorithm places them together with the N parents into a single combined pool of up to $2N$ routes. The algorithm requires parents and offspring to compete on equal terms. The advantage of this approach is that a genuinely strong parent is never discarded simply because it belongs to an earlier generation. It remains eligible for selection and survives unless a better route actually outperforms it. After combining both lists into a single list, it then removes any exact duplicates so that identical routes cannot

occupy multiple positions and reduce the diversity of the population.

5. Non-dominated sort, keep best N routes (Figure 4.4)

The combined route list holds up to $2N$ routes, but the population size is fixed at N . Therefore, this stage determines which N routes survive to become the parents of the next generation. Since no single route can be considered best when two objectives conflict, this comparison relies on the criterion of dominance. A route r_1 is said to dominate a route r_2 if r_1 is at least as good as r_2 on both objectives and strictly better on at least one, that is, if r_1 represents a clear improvement over r_2 . If instead one route takes less time while the other is less costly, neither dominates the other, and the two are said to be incomparable.

Using this comparison, the stage performs a non-dominated sort, which partitions the pool into layers called fronts. The first front contains every route that is not dominated by any other route. These represent the current best trade-offs. Removing this front uncovers a second front, containing routes dominated only by members of the first, then a third front, and so on, with each subsequent layer one tier lower in quality. The next generation is filled one front at a time, beginning with the best, until N routes have been selected. If the pool reaches a front that would push the total beyond N , only part of that front can be retained. In this case, the routes within that front are ranked by crowding distance, a measure of how isolated a route is from its neighbors

in objective space, computed from the distance to its nearest neighbors along each objective. Routes in less crowded regions represent more unique trade-offs and are given priority, so the remaining positions are filled with the routes of largest crowding distance.

6. Number of generations reached (Figure 4.4)

After filtering the best N routes, the algorithm checks how many generations have been completed. If fewer than the given generation count have been completed, it returns to stage 2, and the entire cycle repeats, filtering the best N routes that have just survived as the new parents. If the given number of generations has been completed, it proceeds to the next stage.

7. Return the Pareto set of routes in front 1 (Figure 4.4)

This step delivers the first front routes as the output results. Because travel time and charging cost conflict with each other, there is no single best route to return. To produce this result, the population is ranked, and the first front is selected, the set of routes that no other route dominates. Each route in this set represents a different balance between travel time and cost. This collection of best trade-offs is known as the Pareto set.

4.2 STAGE: 5 - Re-evaluate Optimal Route

The initial route solves the routing problem once at departure time t_0 and returns a Pareto front. The re-evaluation in Figure 4.1 step is described as follows.

In the real world, drivers can select their preferred path from the set of optimal paths. In this thesis, we use the knee point section (Appendix D.3) to select the next route from the Pareto front, then advance along that plan one edge at a time. After each edge, five triggers are evaluated. If any fires, NSGA-II is re-run with the current position as the new origin and the current SoC as the new initial charge, and the resulting plan replaces the current one. The five triggers are listed in Table 4.1. Two are time-based, two are state-based, and one is event-based.

Trigger	Type	Condition
Scheduled	time	At least the minimum re-plan interval has elapsed since the last scheduled replan, the vehicle has traveled at least the minimum re-plan distance, it is not on the same road as before, and the remaining distance to the destination meets the minimum threshold.
Intersection	event	The vehicle has entered an edge on a different road from the previous edge, and the remaining distance to the destination meets the minimum threshold.
SoC Threshold	state	If the current battery level (c) drops below $c^{\text{charging_trigger}}$ and no station is available, the optimizer re-plans in an attempt to locate a station within the revised search area.
Post-charging	state	If the current battery level (c) drops below $c^{\text{charging_trigger}}$, a station is found and used, and the vehicle departs with $c = c^{\text{max}}$. The plan is rebuilt from the new position.
Incident	event	An active road-closure incident blocks the next edge in the current plan at the current time slot.

Table 4.1: Re-plan triggers

4.2.1 Incident Model

Three types of incidents are modeled when re-evaluating routes in this thesis. Road closures on a specific edge, congestion multipliers on a specific edge, and station failures by station identifier. Each incident has a start slot and an end slot and is active only within that window.

Re-plans triggered by incidents use a temporary edge-inflation approach. The affected edges have their length attribute set to 10^9 for the duration of optimization, making them effectively unreachable, then restored to their original values afterward.

4.2.2 Re-plan Cost

A full re-plan runs NSGA-II with a population of N and G generations. An intersection re-plan uses a population of n and g generations. The two configurations reflect different cost-quality trade-offs. Full re-plans are triggered by high-impact events such as low SoC, road incidents, and scheduled intervals, where a more thorough search is justified. Intersection re-plans occur frequently on long journeys and must be computationally inexpensive.

4.3 Summary of the Methodology

In summary, Chapters 3 and 4 presented the methodology developed for dynamic EV route planning under real-world traffic and charging conditions.

The proposed framework integrates data collection and preprocessing, road segment travel time forecasting, charging station availability prediction, and multi-objective route optimization within a unified workflow. Appropriate machine learning and optimization techniques are selected and evaluated at each stage to improve prediction accuracy and routing performance. The methodology provides a systematic approach for generating efficient and reliable EV routes. The next chapters present the experimental setup and results obtained using the Winnipeg road network.

Chapter 5

Implementation

The system architecture for EV route planning, based on the overview provided, is shown in Figure 5.1. The green-colored boxes are system components (or software programs) designed in our methodology, and the teal-colored boxes correspond to information obtained from external actors such as the vehicle.

A forecasting service is executed continuously in real time and independently of the vehicle, obtaining public and private data to predict road segment travel times and charging station availability. This is then stored in a shared forecast server. The optimization server is triggered when a vehicle requests a route based on the predicted forecasts. The request includes the vehicle's current location, battery level, and the model of the car. The optimization server program then reads from the forecast server data and executes the route bi-objective optimization heuristic. The calculated optimal route is then sends to the vehicle. Note that the exchange of

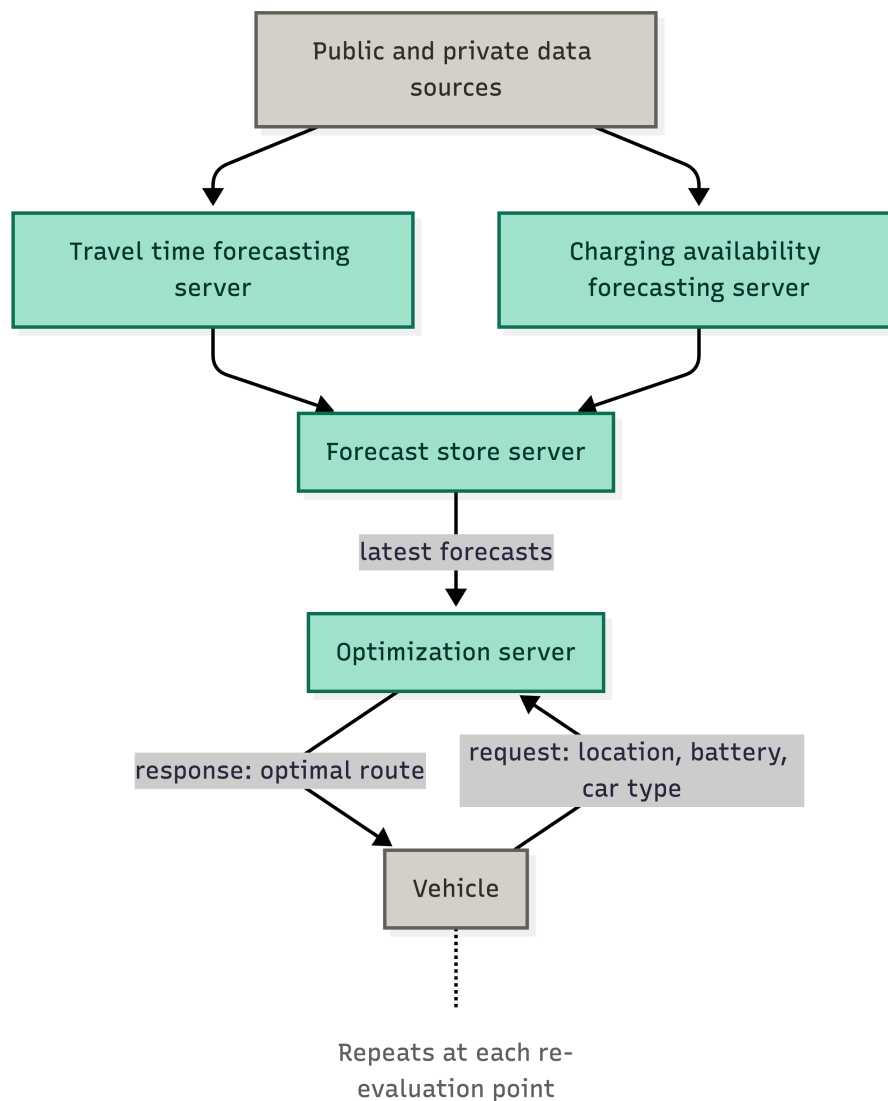


Figure 5.1: System architecture for EV route planning. Gray boxes are external actors (data sources, the vehicle). Teal boxes are system components.

information between the vehicle and the server happens at re-evaluation points, not on every timestep.

5.1 Overview

To test our system, we use Winnipeg as the City to implement and evaluate the proposed methodology. This section describes how each stage of the pipeline was implemented. The road network and traffic data were prepared, the charging station availability forecasting models were trained, Monte Carlo simulation was applied to capture travel time uncertainty, and NSGA-II with knee-point selection was used to determine charging station selection and routing. The output of each stage was evaluated against the criteria defined in the methodology in Chapter 3.

5.2 Experimental Setup

All experiments are implemented in Python 3.14. The road network is constructed and manipulated using OSMnx 2.1.0 [28] and NetworkX 3.6.1 [29]. Traffic interpolation models are implemented in scikit-learn 1.9 [30], and the Gradient Boosting model uses the XGBoost 3.3.0 library [31]. The LSTM models are built and trained in TensorFlow 2.21.0 [32].

Sequential experiments are run on a local machine with an 8-core CPU and 16 GB of RAM. Parallel LSTM training is carried out on the Digital Research Alliance of Canada cluster [33], with one CPU core allocated per cluster model.

5.3 Data Collection

5.3.1 Road Network Data

The road network dataset for the City of Winnipeg in Manitoba, Canada comprises 42,928 road segments and 15,233 road connections (Figure 5.2).



Figure 5.2: City of Winnipeg road map

5.3.2 Historical Traffic Data

The historical traffic data for the city of Winnipeg is incomplete. Public sources only provide vehicle counts for some road segments in the city. Therefore, this

research uses a methodology to estimate traffic on the road segments from those counts, which is explained in the travel time forecasting Section 3.2.1.

Two public datasets give vehicle counts of the City of Winnipeg [34] (Table 5.1). Permanent stations are fixed at specific locations along the road network and record vehicle volumes continuously at those points. Mid-block counts, on the other hand, are collected at different locations across Winnipeg on different days rather than at one fixed point. Hence, they cover more of the city but do not monitor any single location continuously.

Table 5.1: Vehicle-count data sources.

Source	Granularity	Coverage	Records
Permanent Count Stations	Continuous, 15-min counts	8 fixed locations	1.6 M
Midblock Traffic Counts	Manual surveys, 15-min counts	~1500 locations	9.0 M

5.3.3 EV Charging Stations Data

The combined dataset covers 135 charging stations across the City of Winnipeg, including station location coordinates, address, connector types, charging price, and opening times. This data was collected from public and private data sources.

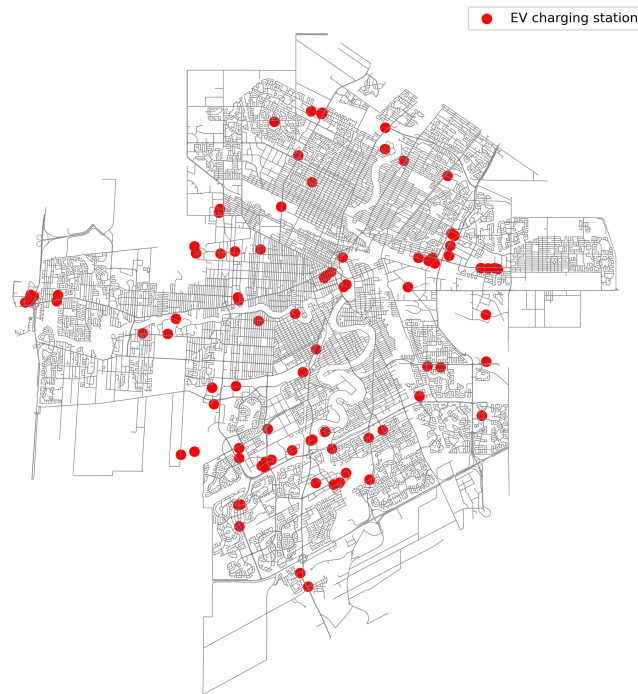


Figure 5.3: City of Winnipeg EV charging stations

5.3.4 EV Charging Availability Data

Real-time availability data for EV charging stations were collected from public charging stations across the City of Winnipeg, Manitoba, Canada. The dataset covers 30 days (20 March-19 April 2025) at a 5-minute sampling interval. It contains the following fields: Location ID, Timestamp, Number of available connectors, Number of installed connectors, Number of connectors planned for future installation, and Number of offline connectors. EV availability data were further collected from private data sources, including HERE Technologies [24] and ChargePoint [23].

5.4 Road Segment Travel Time Forecasting

Historical traffic data for the City of Winnipeg are incomplete. Vehicle counts are available for only a limited subset of road segments. Figure 5.4 outlines how travel time is estimated from vehicle count data. Since most road segments are not monitored, these missing values must be filled in before calculating the network's travel time.

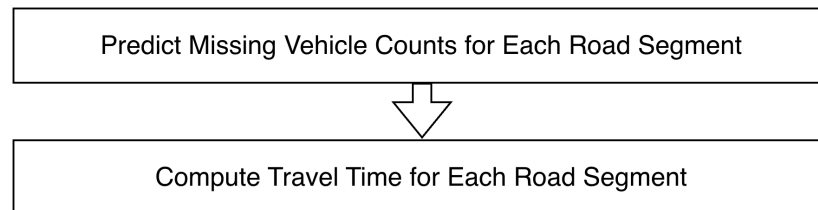


Figure 5.4: Road travel time prediction methodology

5.4.1 Predict Missing Vehicle Counts for Each Road Segment

Before predicting vehicle counts for missing road segments, the following two data preprocessing steps are applied.

1. Filter the raw count CSV files to the six-year observation window.
2. Midblock and permanent sensor locations were mapped onto the City of Winnipeg road network.

After the preprocessing step, six complementary techniques were tested, and the best-performing model was selected.

Implementation of Volume Prediction Models

Missing vehicle counts are estimated using Inverse Distance Weighting (IDW), Kriging, Gradient Boosting, Gradient Boosting + Network features, k nearest neighbor (KNN), and KNN + Network features. All use the same 5-fold cross-validation partition with a fixed random seed.

1. Inverse Distance Weighting (IDW)(Appendix A.1)

Inverse distance weighting is implemented as a distance-weighted averaging method. For each query point, either a road-segment centroid or a held-out sensor during cross-validation, the estimate is the distance-weighted mean of the n training sensors:

$$\hat{z}(x^*, y^*) = \frac{\sum_{i=1}^n w_i z_i}{\sum_{i=1}^n w_i}, \quad w_i = \frac{1}{d_i^p} \quad (5.1)$$

The power parameter p is tuned over the values $\{1.0, 1.5, 2.0, 2.5, 3.0\}$ by minimizing the pooled 5-fold cross-validation RMSE and the selected p is 2.

2. Kriging (Appendix A.2)

To predict traffic volumes at unmonitored locations, we use Ordinary Kriging, a geostatistical interpolation method that estimates each unmonitored value as a weighted combination of observed counts, $\hat{V}(x_0) = \sum_{i=1}^n \lambda_i V(x_i)$, where the

weights λ_i are derived from a variogram model $\gamma(h)$ fitted to the spatial structure of the observed data, rather than from a fixed distance rule [35; 36; 37]. Unlike distance-based interpolation, which treats all locations at a given distance as equally informative, Kriging grounds its weights in how traffic counts actually correlate across the network. The full derivation of the variogram and the Kriging weights is given in Appendix A.2.

3. Gradient Boosting (Appendix A.3)

To capture variation driven by road-level characteristics rather than spatial proximity alone, we additionally model vehicle counts using gradient boosting, which treats count estimation as a regression problem over road attributes such as road type, lane count, posted speed limit, and spatial coordinates. The model is built as an additive ensemble of shallow decision trees, $\hat{V}(x) = \sum_{m=1}^M \gamma_m h_m(x)$, where each tree h_m is fitted sequentially to the negative gradient of the loss function with respect to the current prediction [38]. Unlike the geographic-proximity assumption underlying Kriging, gradient boosting allows the model to weight road category and design attributes directly, rather than location alone. The full model specification is given in Appendix A.3. The Table 5.2 shows the hyperparameters used in the model.

The maximum tree depth of 2 restricts each tree to at most three binary splits, limiting the complexity of individual weak learners. The minimum child node weight of 3 prevents any split from being applied to groups of fewer than three

Table 5.2: Hyperparameters used in the model.

Hyperparameter	Value
Number of boosting rounds	50
Maximum tree depth	2
Minimum child node weight	3
L1 regularisation	1.0
L2 regularisation	2.0

samples, blocking noise-driven partitions on rare feature combinations [31]. All five hyperparameters are selected by five-fold cross-validation with RMSE as the criterion.

4. Gradient Boosting with Network Features

The gradient boosting configuration is identical to method 3 above. The road network structure is characterized using seven features, summarized in Table 5.3.

5. K-Nearest Neighbors (Appendix A.4)

As a further baseline, we estimate vehicle counts using K-Nearest Neighbors (KNN), a non-parametric method that makes no assumptions about the underlying data distribution and instead relies on similarity in feature space [39]. We use a distance-weighted variant [40], in which the predicted count at an

unmonitored segment is computed as an inverse distance-weighted average of its K nearest monitored segments, so that segments more similar to the query point contribute more to the prediction than those further away. KNN differs from Kriging and gradient boosting in that it does not explicitly model spatial dependence or road-level attributes. Instead, it bases its estimate purely on the local similarity structure of the feature space. The general form of the distance-weighted estimator is given in Appendix A.4.

A distance-weighted KNN regressor with $k = 7$ is applied to the base feature set. The value $k = 7$ is selected by a grid search over $k \in \{3, 5, 7, 10, 15\}$ using five-fold cross-validation.

6. K-Nearest Neighbors with Network Features

The K-Nearest Neighbors configuration is identical to Method 5 above. The road network structure is characterized using seven features, summarized in Table 5.3.

Network Feature Set

Table 5.3 lists all features used in Methods 4 and 6. The base features (length, lanes, maxspeed) describe physical road characteristics. The network features (betweenness, closeness, degree, node density) describe each segment’s structural position in the road graph.

Table 5.3: Feature set for estimating the vehicle count on a road segment.

Feature	Description
Length	Geometric length of the road segment
Lanes	Number of lanes
Maxspeed	Posted speed limit
Betweenness	Edge betweenness centrality. Fraction of shortest paths passing through the segment [41]
Closeness	Mean node closeness centrality for the segment's endpoints [41]
Degree	Node degree of the segment's endpoints
Node Density	Local node density near the segment midpoint

After predicting the missing vehicle counts, the Bureau of Public Roads (BPR) (AppendixB.2) function is applied to each road segment to determine its travel time.

5.4.2 Travel Time Calculation for Each Road Segment

With vehicle counts estimated for all road segments, travel times can be derived across the full network using the Bureau of Public Roads (BPR), which gives the total travel time for each segment. The BPR function requires five parameters. These include the calibration parameters α and β , traffic volume, road segment capacity, and free-flow travel time. The calibration parameters are set to their standard values,

where α is 0.15 and β is 4, and the vehicle counts are used as the traffic volume.

The segment capacities are derived from the lane count and a per-lane capacity lookup by road class [42]:

$$C \text{ [vph]} = \text{lanes} \cdot c_{\text{lane}} \quad (5.2)$$

where c_{lane} is the per-lane capacity from the table in Appendix B.1. The free-flow travel time is computed from segment length and the posted speed limit:

$$t_0 \text{ [s]} = \frac{L}{v_{\text{free}}/3.6} \quad (5.3)$$

where L is the segment length in meters and v_{free} is the max speed value in km/h when present. Where max speed is absent, a road-class default from Appendix B.1 is substituted. The travel times produced here are used in the Monte Carlo forecasting stage that follows.

5.4.3 Monte Carlo Simulation for Future Travel Time

After estimating travel time, the Monte Carlo simulation described in Chapter 3 is used to forecast future travel time for each segment. Travel time variability is modeled using a Poisson distribution, consistent with the approach outlined earlier. For each segment, $K = 1,000$ volume scenarios are sampled, and the BPR function is applied to each scenario to produce K travel time estimates per segment.

5.5 Forecasting EV Charging Station Availability

With travel time forecasting complete, the next stage is forecasting charging station availability, using the three model configurations described in Chapter 3. The City of Winnipeg has 135 EV charging stations in total. After preprocessing, 86 remain, since the remaining 49 report zero available connectors across every record in the data.

5.5.1 Implementation of SARIMA

A separate SARIMA model is fitted per station, since SARIMA does not support multivariate input. The forecast horizon is 12 hours. Parameter search bounds are set to $p, q \leq 3$, $d \leq 2$, $P, Q \leq 1$, and $D \leq 1$, selected to limit the search space while covering the orders most commonly identified from occupancy data.

5.5.2 Implementation of the LSTM Model and its Parallelization

The LSTM model is implemented as described in Section 3.2.2. The model configuration is shown in the table 5.4:

Table 5.4: Sequential LSTM configuration.

Parameter	Value
Forecast horizon (H)	48 steps (12 hours at 15-minute intervals)
Hidden size (d)	200
Stacked layers	3
Batch size	1
Maximum epochs	30
Partition gap	143 sequences

To speed up the training process, we parallelized the LSTM model. This is discussed below.

Parallel LSTM: The stations are divided into groups, with a separate model trained for each group. Stations are grouped by K-means clustering on usage patterns rather than by geography. Grouping by usage pattern has two advantages. First, stations within each cluster share similar demand shapes, so each regional model learns a more uniform input-output mapping. Second, the number of models is reduced, lowering both training time and the overhead of maintaining separate models as the station network changes.

Each station is represented by a 34-dimensional feature vector, drawn from three groups of features:

- **Summary statistics (3 features):** mean availability, standard deviation, and connector count.
- **Hourly profile (24 features):** average availability for each hour of the day.
- **Daily profile (7 features):** average availability for each day of the week.

Together, these features capture each station’s overall activity level and the shape of its temporal demand across the week. Feature vectors are standardized before clustering to prevent high-variance features from dominating the distance metric.

Configuration of the Parallel LSTM model A separate model is trained for each usage-pattern cluster. Each model receives input features from all stations at every time step, but it produces predictions only for the stations in its own cluster. The full input is retained across all models because dropping it would lose the cross-station correlations from stations outside a given cluster. This reduces output dimensionality from the total number of stations, as in the sequential model, down to just the stations in each cluster. In contrast, input dimensionality stays the same across all models.

Applying usage pattern-based station clustering identifies five clusters, chosen using silhouette analysis subject to the constraint that each cluster contains enough stations to support model training. Table 5.5 shows the resulting cluster sizes.

Cluster	Stations
Cluster 0	42
Cluster 1	9
Cluster 2	2
Cluster 3	15
Cluster 4	18
Total	86

Table 5.5: Station cluster sizes from K-means clustering on usage patterns ($K = 5$).

Cluster 2 contains only two stations whose demand patterns did not fit any larger group. These are retained as a separate model rather than merged into another cluster, since forcing them into a group with dissimilar stations would reduce model accuracy for both. The remaining clusters range from 9 to 42 stations.

After identifying the clusters, the configuration of the parallel LSTM is shown in the Table 5.6:

Table 5.6: Parallel LSTM configuration (applied to each of the five cluster models).

Parameter	Value
Forecast horizon (H)	48 steps (12 hours at 15-minute intervals)
Hidden size (d)	200
Stacked layers	3
Batch size	32
Maximum epochs	30
Partition gap	143 sequences

All cluster models are trained simultaneously on the high-performance computing cluster available from Digital Research Alliance of Canada [33]. The computational environment on the HPC cluster was configured to match the specifications of the local computing environment used for training the sequence LSTM models, ensuring consistency in the model training setup and computational requirements. Each cluster is assigned to a separate processor core, allowing the models to train in parallel rather than sequentially and reducing the overall training time. Since each model runs independently on a separate core, the total training time is determined by the cluster model with the longest training time.

5.6 Find the Initial Route

To find the initial route, NSGA-II is used to minimize travel time and charging fee, following population initialization, solution encoding, genetic operators, and fitness evaluation described in the methodology section 4.1.2. The implementation and parameter configuration are as follows 5.7:

Parameter	Symbol	Value
Population size	N	100
Generations	G	50
Tournament size	k	2
Mutation probability	p_m	0.3
Battery capacity	B	60.0 kWh
Consumption rate	K	0.18 kWh/km
Initial SoC	c_0	0.40
Minimum SoC	$c^{\min}$	0.10
Charge trigger SoC	$c^{\text{charging_trigger}}$	0.25
Charge target SoC	$c^{\max}$	0.55

Table 5.7: Default parameter values used in initial route experiments.

5.7 Re-evaluate Optimal Route

The optimal routes are re-evaluated as described in Section 4.2. The configuration used for this step is shown in Table 5.8.

Parameter	Symbol	Value
Population size	N	100
Generations	G	50
Tournament size	k	2
Mutation probability	p_m	0.3
Battery capacity	B	60.0 kWh
Consumption rate	K	0.18 kWh/km
Initial SoC	c_0	0.40
Minimum SoC	$c^{\min}$	0.10
Charge trigger SoC	$c^{\text{charging_trigger}}$	0.25
Charge target SoC	$c^{\max}$	0.55
Replan interval		20 min
Minimum replan distance to destination		3,000 m
Minimum replan travel		5,000 m
Intersection NSGA-II		population 50, 20 generations

Table 5.8: Default parameter values used in route re-evaluation experiments.

5.8 Summary of Implementation

This chapter presented the implementation methodology developed for dynamic EV route planning under real-world traffic and charging conditions, based on the methods described in Chapters 3 and 4. The next chapter presents the results obtained from this implementation.

Chapter 6

Results

This chapter presents the results of the road segment travel time forecasting, EV charging station availability forecasting, and route optimization. Each stage is evaluated independently, and the predictive model stages are evaluated using standard regression and classification metrics. The optimization stage is evaluated using Pareto front quality indicators and representative route comparisons.

6.1 Road Segment Travel Time Forecasting

Since the City of Winnipeg does not have publicly available traffic data, historical travel time was estimated using the approach described in the implementation Section 5.4. Six methods are proposed for this estimation, and the following section presents their results along with the stages of each method.

6.1.1 Methods for Estimating Traffic Volumes Comparison

All six methods are evaluated on the same five-fold partition with the same random seed. Table 6.1 reports the results.

Table 6.1: Method comparison: five-fold cross-validation.

Rank	Method	R^2	MAE	RMSE	Std-RMSE
1	KNN + Network features	0.534	61.7	97.0	0.759
2	KNN	0.505	64.1	101.3	0.793
3	Gradient Boosting	0.455	70.6	102.4	0.802
4	Gradient Boosting + Network	0.413	72.4	105.8	0.828
5	Regression Kriging	0.269	74.7	109.1	0.854
6	IDW)	0.055	83.1	124.0	0.971

KNN with network features produces the strongest result across all four metrics; the R^2 is 0.534, the MAE is 61.7 vph, and the RMSE is 97.0 vph. Figure 6.1 shows the predicted versus observed scatter for the top two methods.

Impact of Network Features

The same network features have opposite effects on the two machine learning methods, as shown in Table 6.2. This is the most practically useful finding of the comparison.

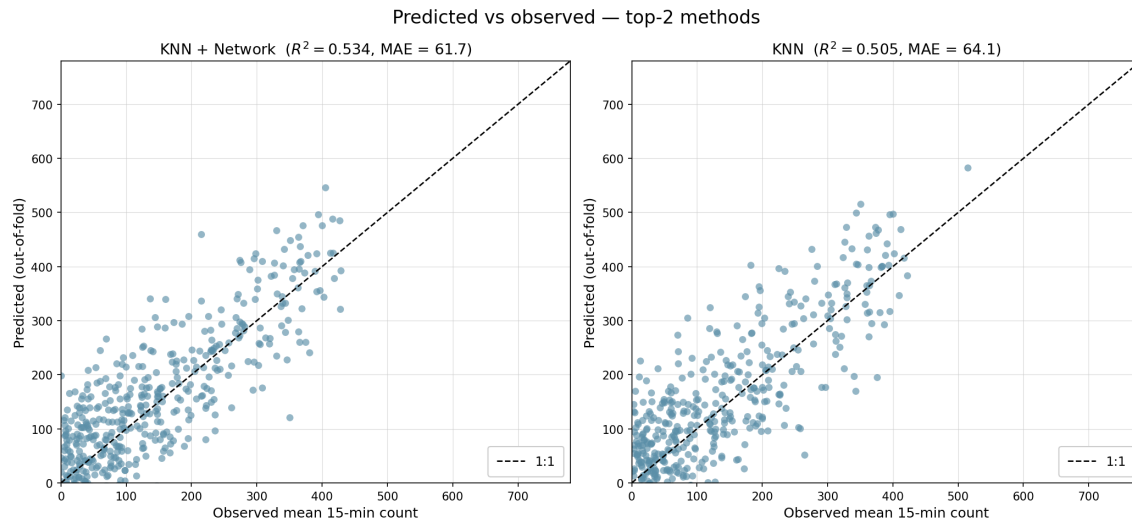


Figure 6.1: The top two methods.

Table 6.2: Effect of adding network features on cross-validated R^2 .

Method	R^2 (base)	R^2 (+ Network)	ΔR^2
KNN	0.505	0.534	+0.029
Gradient Boosting	0.455	0.413	-0.042

For KNN, the improvement is consistent with how the method works. Network features encode the structural role of a road segment independently of its physical attributes. Two data collection locations assigned to the same road can differ substantially in terms of centrality. One sits on a through-route, the other on a dead-end street. Because KNN uses every standardized feature equally in the distance metric, adding those features that genuinely separate these cases tightens the neighborhood

match and improves predictions.

For Gradient Boosting, including the same features reduces model performance. With 367 training points, the XGBoost configuration lacks the capacity to learn meaningful splits on six additional correlated continuous variables without fitting noisy quantile thresholds. The R^2 reduction of 0.042 is consistent with this interpretation. The BPR analysis uses KNN with network features as the volume model.

IDW and Kriging Performance

The R^2 for DW is 0.055, and the R^2 for Regression Kriging is 0.269. Both results are lower than those obtained by every machine learning method tested. The underlying reason is that vehicle counts at locations 50 m apart can differ by an order of magnitude when the data collected locations sit on different road classes. IDW averages across that boundary by construction, producing smoothed predictions that underestimate high-volume arterials and overestimate low-volume residential streets. Regression Kriging gets some of its accuracy from the road-type trend term. After accounting for road type, there is nothing left for kriging to learn, and spatial correlation adds almost nothing on top of the class mean.

These results indicate that IDW requires road-class correction to achieve good performance. Distance-based weighting alone does not generalize reliably across different road classes in an urban road network.

6.1.2 BPR Function Results

BPR was applied to KNN using network-feature-based volume predictions, where each volume represented the six-year mean traffic count from May 2020 to October 2025.

Table 6.3: Network-wide BPR results: six-year mean (08:00-08:30), 2020-2025

Network metric	Value
Mean V/C	0.315
Median V/C	0.207
95th-percentile V/C	1.024
Maximum V/C	2.704

According to Table 6.3, The mean V/C (volume-to-capacity ratio, the ratio of observed traffic volume to a road segment's maximum carrying capacity) of 0.315 and median of 0.207 indicate that most segments operate well below capacity during the morning peak. The 95th-percentile V/C of 1.024, however, confirms that a subset of segments already exceeds capacity, and the maximum of 2.704 points to severe localized over-saturation on freeway ramps and high-flow arterial shortcuts.

The majority of the network operates at Level of Service (LOS) A during the morning peak. LOS is a qualitative measure of traffic flow, ranging from A to F, based on the ratio of traffic volume to road capacity: LOS A corresponds to

free-flow conditions with minimal delay, while LOS F corresponds to over-saturated flow with severe congestion. A full description of the LOS scale, together with the corresponding volume-to-capacity thresholds, is provided in Appendix B.3. 5.2% of segments operate at LOS F and are concentrated in two locations. These are freeway ramps, which form the most over-saturated road class in the dataset, and high-flow streets in downtown that carry commuter traffic.

Table 6.4: LOS distribution(2020-2025 mean).

LOS	Description	Segments	%
A	Free flow	37,566	87.8
B	Reasonably free	1,144	2.7
C	Stable	829	1.9
D	Approaching unstable	634	1.5
E	At capacity	410	1.0
F	Breakdown	2,226	5.2

6.1.3 Future Travel Time: Monte Carlo Results

The Monte Carlo simulation draws 1,000 Poisson samples per segment from the BPR volume predictions. The mean V/C ratio across segments is 0.315, with 5.20% of segments exceeding $V/C > 1$ on average and 6.40% at the 95th percentile.

6.1.4 Discussion

The R^2 of 0.534 reflects a feature-information limit rather than a modeling capacity limit. Without data on commercial density, land use, or access road configuration, all six methods perform similarly, suggesting the limiting factor is data availability rather than model choice. Increasing model complexity beyond what KNN with network features provides does not close the gap. It produces overfitting, as the unconstrained Gradient Boosting result demonstrates.

The asymmetric effect of centrality features on KNN and Gradient Boosting carries a practical implication for feature engineering. The same three features improve KNN by tightening the distance metric and degrade Gradient Boosting by adding correlated inputs that a shallow tree cannot split on meaningfully. Therefore, features should be selected based on the predictive model.

6.2 Forecasting EV Charging Stations Availability

Accurate travel time estimates are necessary but not sufficient for feasible EV routing. A route that reaches a charging station on time is still infeasible if the station is occupied on arrival.

Three models are evaluated: SARIMA, a sequential LSTM, and a parallel LSTM. All three predict the exact number of available connectors at each of the 86 retained

stations over a 12-hour horizon at 15-minute intervals ($H = 48$ steps).

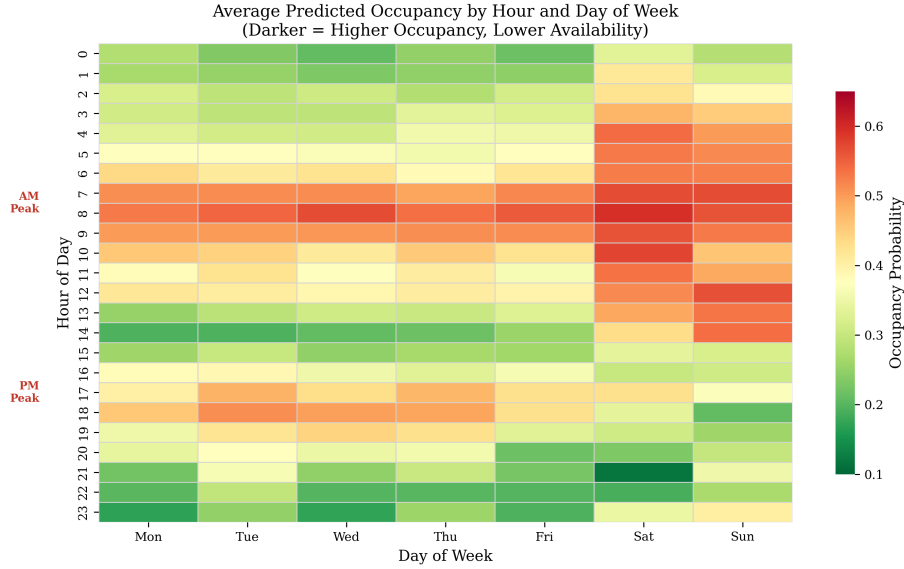


Figure 6.2: Average predicted charging station occupancy by hour of day and day of week across all Winnipeg stations.

Figure 6.2 shows average predicted availability across all stations by hour of day and day of week. EV station occupancy is highest on weekday mornings (07:00-09:00) and evenings (16:00-19:00), consistent with commute-driven charging demand. Weekend patterns are flatter. Occupancy stays moderate throughout the day, with a slight dip in the early afternoon, suggesting casual rather than commute-driven usage.

6.2.1 SARIMA Performance

A separate SARIMA model is fitted per station with seasonal period $m = 96$, corresponding to one full daily cycle at 15-minute intervals. Table 6.5 reports test-set performance.

Table 6.5: SARIMA performance.

Metric	Value
MAE (connectors)	0.4632
RMSE (connectors)	1.2143
R^2	0.6962
Training time	87.4 minutes

SARIMA accounts for approximately 70% of the variance in connector counts, which is substantially lower than the performance of the LSTM models. The gap is attributable to the model’s univariate structure. SARIMA fits each station independently and cannot draw on demand patterns at other stations or on network-wide temporal signals. In this thesis, a station is classified as dynamic if it records at least one EV visit on a given day, and as static if EV visits occur only rarely, that is, less than once per day on average. Among the 23 dynamic stations, 22 produce negative R^2 values, and the average exact match rate across dynamic stations is 45.4%. In contrast, static stations perform substantially better at 84.4% exact match, because

near-constant series are straightforward for a seasonal model to reproduce regardless of whether cross-station information is available.

SARIMA also requires the longest training time, 87.4 minutes, as each of the 86 stations requires an independent parameter search, with no parameter sharing across models.

6.2.2 Sequential LSTM Performance

The sequential LSTM is trained on all 86 stations simultaneously with a 12-hour forecast horizon. Training converged at epoch 15 of 30 through early stopping. Table 6.6 reports test-set performance.

Table 6.6: Sequential LSTM performance.

Metric	Value
MAE (connectors)	0.1873
RMSE (connectors)	0.4952
MAPE	6.62%
R^2	0.9504
Early-stop epoch	15 / 30
Training time	1,335 s

The model accounts for 95.0% of the variance in connector counts across all 86

stations and 48 future time steps. An MAE of 0.19 connectors means predictions deviate from the true count by less than one-fifth of a connector on average. However, experiments with increasing batch sizes showed a noticeable decline in prediction accuracy, suggesting that smaller batch sizes were more suitable for predicting connector availability.

6.2.3 Parallel LSTM Performance

Five cluster-level models are trained independently. Each receives the full 89-dimensional input, consisting of availability values from all 86 stations and three temporal features, but predicts connector counts only for stations in its own cluster. Table 6.7 reports the combined metrics and the per-cluster average.

Table 6.7: Parallel LSTM aggregate performance.

Metric	Value
MAE (connectors)	0.2566
RMSE (connectors)	0.5696
MAPE	10.65%
R^2	0.9344
Avg early-stop epoch	13.8 / 30 (range: 6-21)
Training time	141 s (longest cluster)

The combined R^2 of 0.9344 was computed by aggregating the predictions from all five clusters into a single vector and evaluating this against the observed values. This is the appropriate comparison metric against the sequential model, since it covers the same 86 stations and 48 time steps. The MAE is a more informative metric for these clusters, and on both MAE and R^2 , the parallel model performs comparably to the sequential model.

6.2.4 Comparative Analysis

Table 6.8 compares all three models on the same dataset, 12-hour horizon, 70/15/15 split, and evaluation metrics.

Table 6.8: Comparison of all models for connector count prediction (12-hour horizon, 86 stations).

Metric	SARIMA	Seq. LSTM	Par. LSTM
MAE (connectors)	0.4632	0.1873	0.2566
RMSE (connectors)	1.2143	0.4952	0.5696
R^2	0.6962	0.9504	0.9344
Training time	5,254 s	1,335 s	141 s
Models trained	86	1	5

The R^2 gap between SARIMA (0.70) and LSTM (0.95) is 0.25. Two structural

limitations account for this. SARIMA fits each station in isolation without access to signals from other stations.

The parallel LSTM's R^2 of 0.9344 sits 1.6 points below the sequential model, but requires only 141 s for training. Since only the affected cluster requires retraining after station infrastructure changes, this architecture is more practical for live deployment in an evolving station network.

All models achieve MAE below 0.26 connectors. After rounding to integers, the predicted count matches the true value in most cases, which is sufficient accuracy for the route optimization to make reliable charging station selections.

6.2.5 Limitations

Two limitations apply to the availability prediction results. The first is the short observation window. The dataset covers only 30 days, from 20 March to 19 April 2025, and never sees a full seasonal cycle, so accuracy could suffer once conditions drift toward the city of Winnipeg winter the model has never encountered. The second is the absence of external covariates. Weather, electricity prices, local events, and traffic volume are all excluded from the input features, despite plausibly driving demand at the stations. Adding them is likely the most direct way to improve accuracy on that harder subset.

6.3 Route Optimization Results

NSGA-II is applied to generate Pareto-optimal route sets from the traffic and availability predictions produced in the preceding stages. The resulting Pareto front is well-distributed across the objective space, with solutions spanning from minimal travel time at higher charging cost to minimal charging cost at longer travel time.

6.3.1 Experimental Routes

The five routes are listed in Table 6.9. All share the default optimization parameters.

Route Id	Description	Approx. km	Start node	End node
0	West to East	33.5	288981953	2566870611
1	Southwest to Northeast	31.5	367131619	1688849660
2	West to East	30.2	9666839223	1688848613
3	East to West	29.8	8151998098	4410522479
4	Southwest to Northeast	29.7	1152186186	1688786938

Table 6.9: The five evaluated routes

6.3.2 Initial Optimal Route Results

Table 6.10 summarizes the initial route optimization for every route. To account for variability in the optimization process, we evaluated each route over multiple

independent runs. The best time and best fee columns are the two extremes of the NSGA-II Pareto front. A* time and fee are the shortest-length baseline evaluated under the same congestion and charging model.

Route	A* baseline		NSGA-II Initial Optimal Routes			
	Time	Fee	Front	Best time	Best fee	Runtime
	(min)	(\$)	size	(min)	(\$)	(s)
0	41.01	1.77	9	39.37	0.688	45.1
1	41.79	0.76	10	40.82	0.682	41.0
2	38.08	0.00	1	35.12	0.000	39.6
3	40.93	0.00	1	35.98	0.000	48.3
4	38.87	0.00	1	38.43	0.000	38.8

Table 6.10: Initial optimal routes results

NSGA-II beats A* on travel time on every route, by 1.0-5.0 minutes. Averaged across the five routes, the best-time front point is 37.9 min against A*'s 40.1 min, a 5.5% drop. The biggest gains are on routes 2 and 3 (roughly 3 and 5 minutes, 7-12%), where A* commits to a congested road that the optimization routes around. On the two routes that require charging (0 and 1), NSGA-II also cuts the fee hard. A* pays \$1.77 on route 0 because it charges at an expensive ($\approx$ \$0.18/kWh) station along its straight-line corridor, while the cheapest NSGA-II solution detours to a $\approx$ \$0.07/kWh station and pays only \$0.69, a 61% savings.

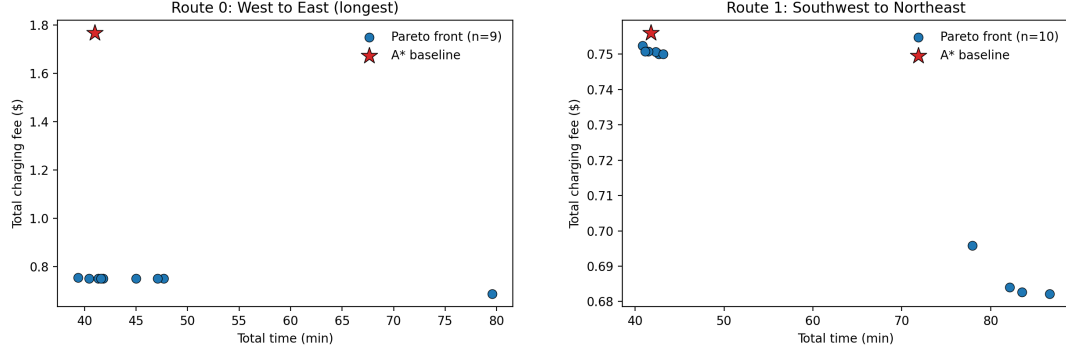


Figure 6.3: Per-route Pareto fronts (blue), and the A* baseline (red star). Routes 0-1 show a genuine trade-off.

Figure 6.3 and Figure 6.4 show them individually with the A* point marked.

The Charging Threshold and the Degenerate Fee Objective

The fee objective does not always exist. With an initial SoC, any route whose driven distance stays below that range never crosses the 25% charge trigger and never inserts a charging stop. Routes 2-4 drive 29.6-30.2 km and finish on the initial charge. Their fee is zero for every feasible path. There is no trade-off to find, and NSGA-II correctly returns a single time-minimizing solution (front size 1 in Table 6.10).

This pattern is directly reflected in the SoC traces shown in Figure 6.5. Route 0 drops below the 25% trigger threshold near the midpoint, prompting a recharge to the 55% target level. Route 2 descends monotonically and terminates above the trigger threshold, at no point reaching the charging station.

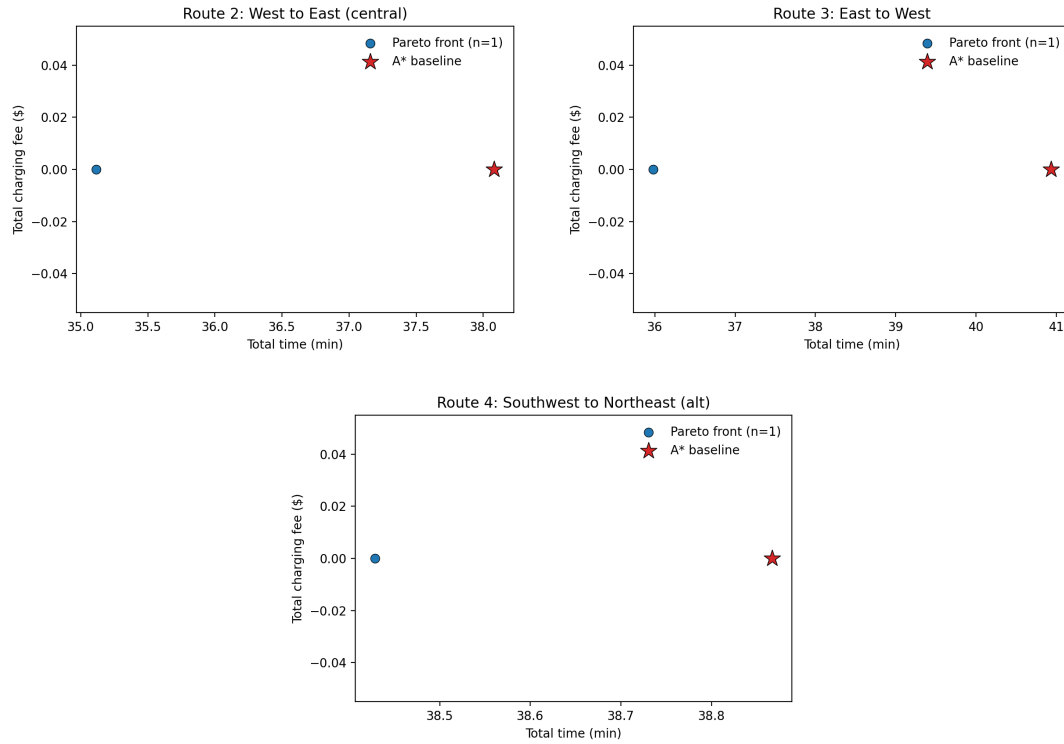


Figure 6.4: Per-route Pareto fronts (blue), and the A* baseline (red star). Routes 2-4 reduce to a single non-dominated point because every feasible path has zero charging fee.

Convergence Behaviour

Figure 6.6 presents the NSGA-II convergence metrics (best time, best fee, and Pareto-front size) over 50 generations for routes 0 and 1, the two that require charging. The best time exhibits stepwise improvement, converging well before generation 50. For Route 0, the final improvement occurs near generation 30, beyond which the value remains unchanged. The best fee remains essentially constant from the outset.

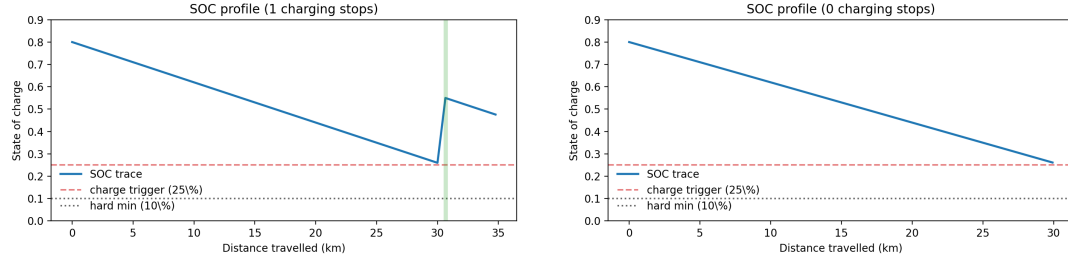


Figure 6.5: SoC profiles for the best-time solution. Left: route 0 needs one charging stop (green band). The trace touches the 25% trigger (red dashed line). Right: route 2 completes on the initial charge and never triggers, which is why its fee objective is degenerate.

The cheap-station detour is already present in the seed population, and the genetic operators therefore preserve this solution rather than discover it.

The quality of the objectives is largely determined by the population generated through heuristic and random route generation. The genetic operators contribute little beyond preserving the solutions already identified in the seed population, which also explains why 50 generations are more than sufficient. Both objectives converge by generation 30 across all routes.

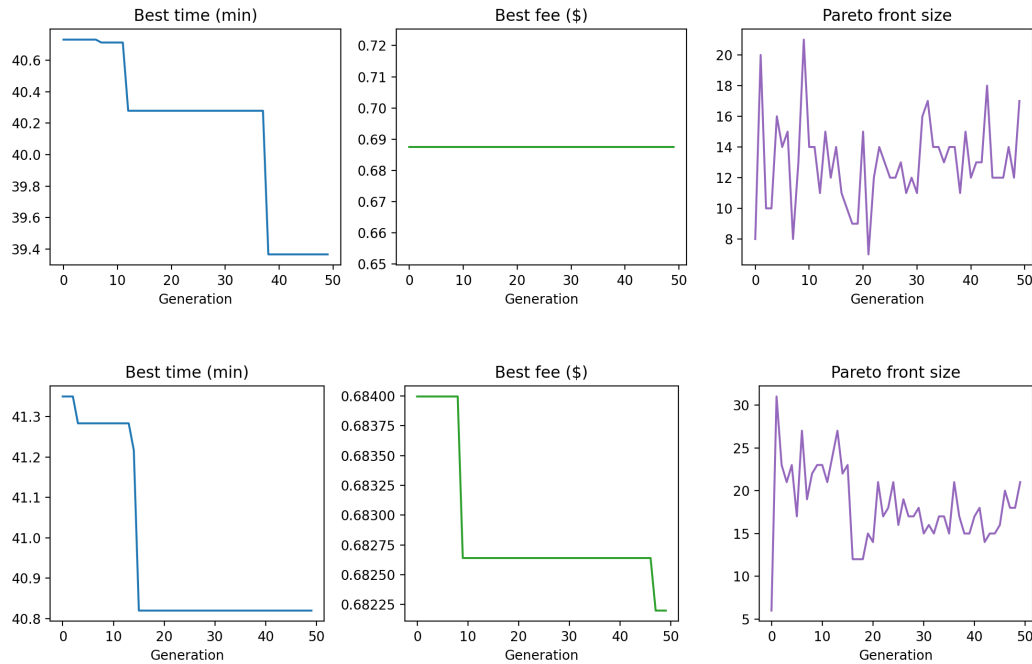


Figure 6.6: Convergence of best time, best fee, and Pareto-front size over 50 generations for route 0 (top) and route 1 (bottom).

6.3.3 Re-evaluating Optimal Routing Results

Table 6.11 shows how often each route was re-evaluated mid-journey, and what triggered each replan. All five routes successfully reached the destination after re-evaluation. The optimization selected the cheap station ($\approx \$0.07/\text{kWh}$) on Route 1, incurring a cost of \$0.75, but reverted to the expensive station ($\approx \$0.18/\text{kWh}$) on Route 0, incurring a cost of \$1.50, closer to A^* 's \$1.77 than to the \$0.69 cheap end of the static front.

Route	Executed journey				Replan triggers			
	Time	Fee	Dist.	Stops	Inter.	Sched.	Post-chg	Runtime
	(min)	(\$)	(km)					(s)
0	42.30	1.50	34.70	1	34	4	1	95.2
1	40.77	0.75	32.27	1	28	5	1	91.7
2	37.98	0.00	29.64	0	24	4	0	78.3
3	36.01	0.00	30.20	0	27	5	0	104.8
4	38.05	0.00	29.76	0	28	5	0	83.3

Table 6.11: Re-evaluation execution per route.

The replan workload remains stable across all routes and is dominated by the intersection trigger, with 24-34 intersection replans per trip compared to only 4-5 scheduled replans and at most one post-charging replan. Re-evaluation occurs at nearly every turn but rarely alters the route, consistent with the previously measured 93.9% intersection no-op rate. The runtimes (78-105 s) are approximately twice those of the initial runtimes (39-48 s).

Figure 6.7 shows the trigger composition across routes, and Figure 6.8 compares the initial route against the re-evaluate execution objective by objective.

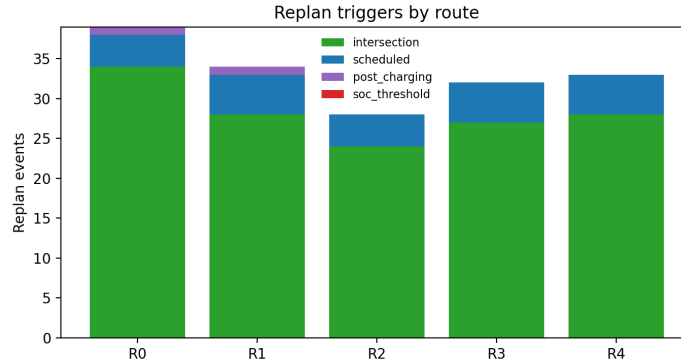


Figure 6.7: Replan-trigger composition by route. Intersection triggers (green) dominate over scheduled triggers, while post-charging triggers add a small, constant contribution.

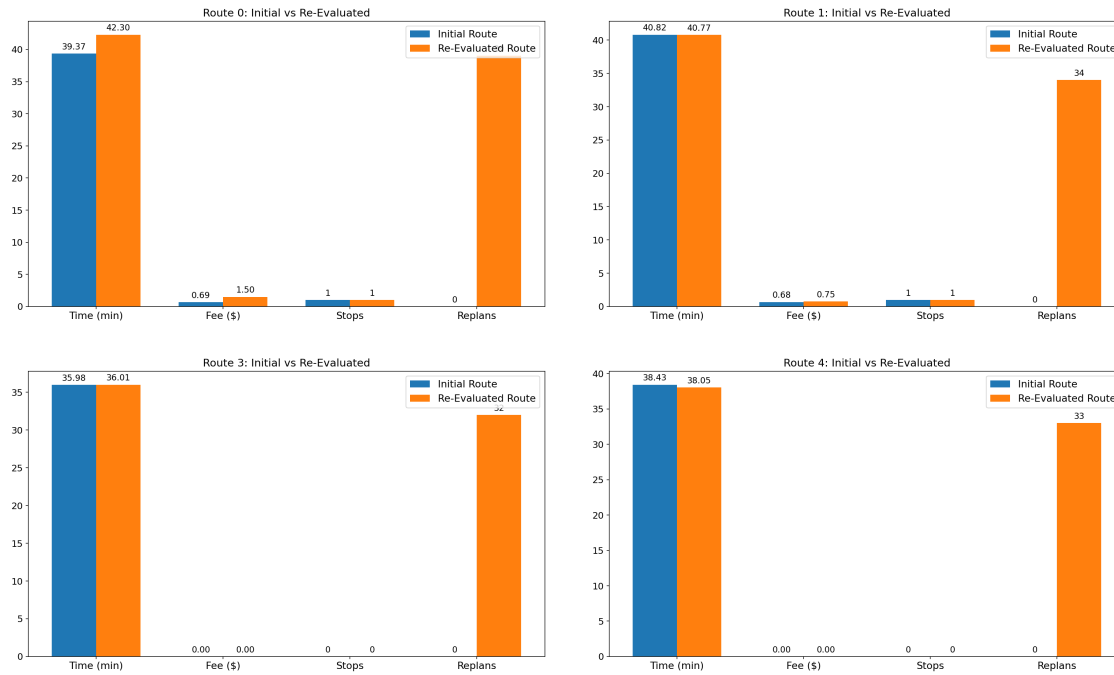


Figure 6.8: Initial optimal routes vs. re-evaluated optimal routes

6.3.4 Initial vs. Re-Evaluate Optimization

A comparison of the best-time initial route with the re-evaluated route execution (Tables 6.10 and 6.11) shows that the re-evaluated routes achieve travel times within approximately ± 3 minutes of the initial estimate on every route. It is slower on the two routes that require a charging stop (Route 0: 42.3 min compared to a 39.4 min initial best route; Route 2: 38.0 min compared to 35.1 min). On the remaining routes, the performance is essentially unchanged or marginally improved (Route 1: 40.8 vs. 40.8 min; Route 3: 36.0 vs. 36.0 min; Route 4: 38.0 vs. 38.4 min), because mid-journey congestion updates occasionally identify a slightly better route segment.

With no incidents to react to, the initial optimal route is as good as, or slightly better than, the re-evaluated route. The route re-evaluation approach yields its greatest benefit when road closures, station failures, or congestion spikes occur that the initial optimal route cannot anticipate.

6.3.5 Optimal Route Maps and Charging Stations

Figures 6.9 through 6.13 draw the optimal routes for each case on the city of Winnipeg road network, overlaying four plans. The A* shortest-length baseline appears in red, the static NSGA-II best-time solution in blue, the Initial NSGA-II optimal route's best-time solution in green, and the re-evaluated route optimization in orange. Charging stops are drawn as stars coloured by the route that uses them, with the station name annotated. The green start marker and red end marker fix

the orientation.

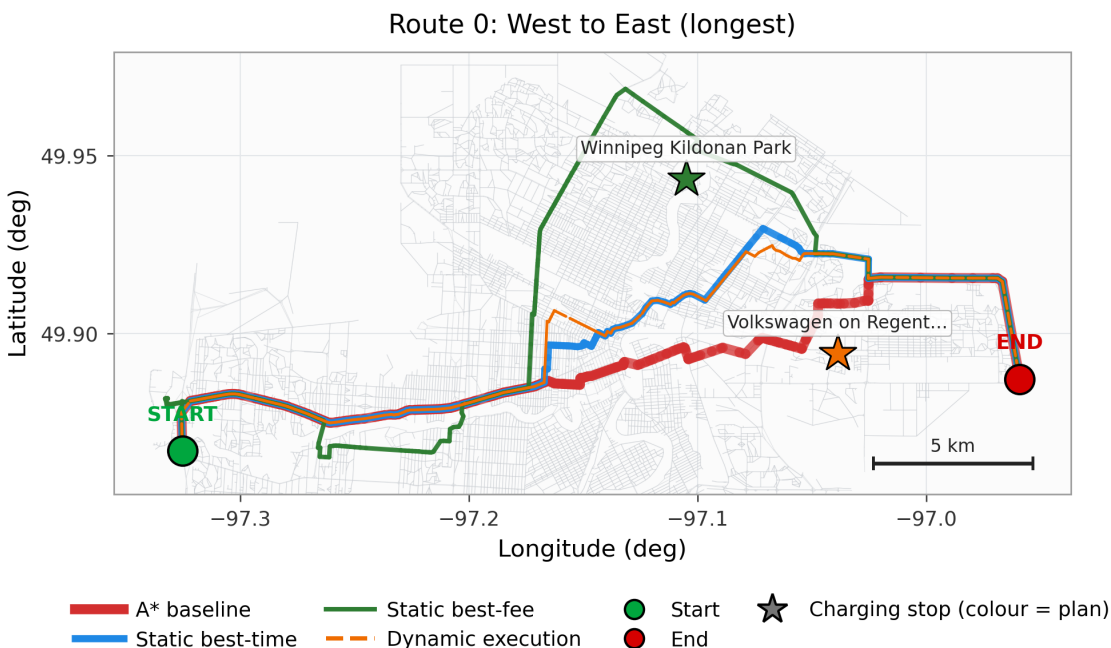


Figure 6.9: Optimal route map for Route 0

On the two charging routes 0 and 1 in Figure 6.9 the best-fee plan visibly detours north to reach a cheap ($\approx \$0.07/\text{kWh}$) station. On route 0 it swings up to the Kildonan Park area, whereas the best-time and A* plans charge at a more conveniently located but more expensive ($\approx \$0.18/\text{kWh}$) station near the direct corridor. The detour distance the green plan accepts is the price of its 60%+ fee saving. The re-evaluated route execution follows the best-time corridor closely and charges along it. On route 1, the station nearest that corridor happens to be the cheap one, so the re-evaluated route pays \$0.75. On route 0, the optimization selects a separate

station in the northeast and pays \$1.50.

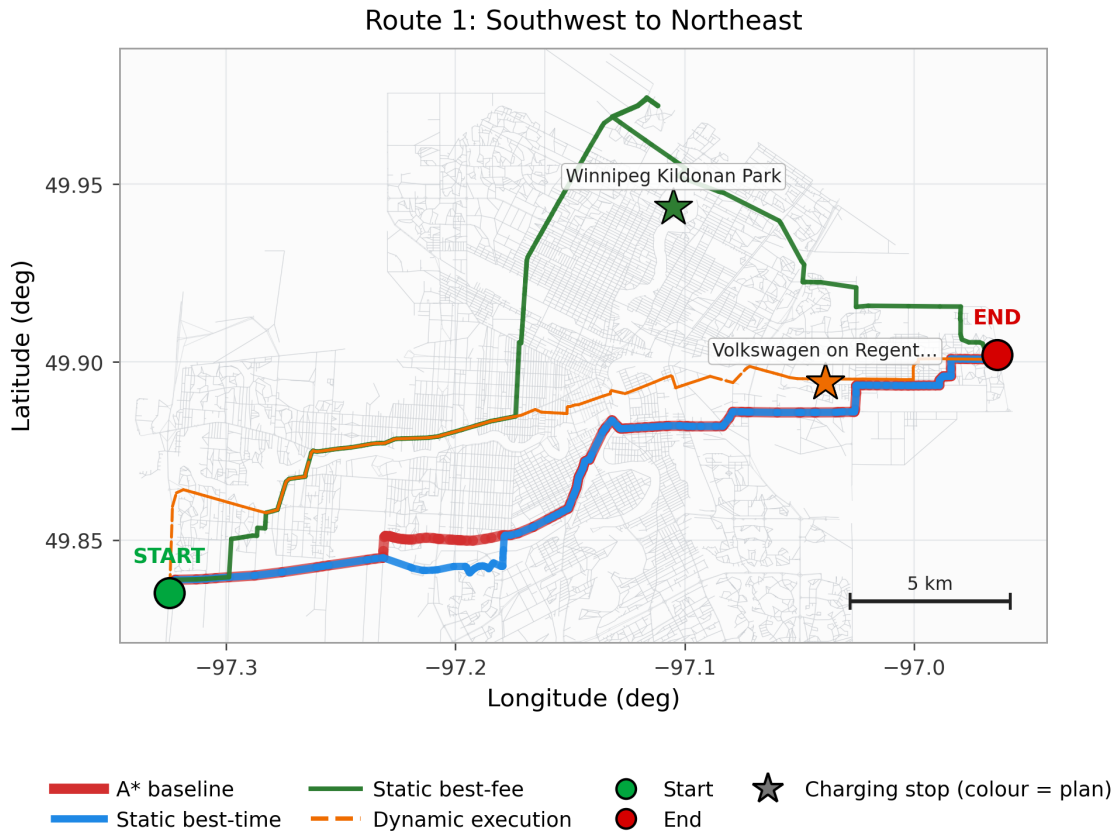


Figure 6.10: Optimal route map for Route 1

On the three non-charging routes (2-4, Figures 6.11 and 6.13), the four plans carry no charging stars at all, as the trip completes on the initial charge. The best-time, best-fee, and re-evaluated routes largely converge, since without a fee to trade against, they all reduce to the same time-minimizing corridor. The A* plan is the main outlier, committing to a slower road that both optimized routes avoid. .

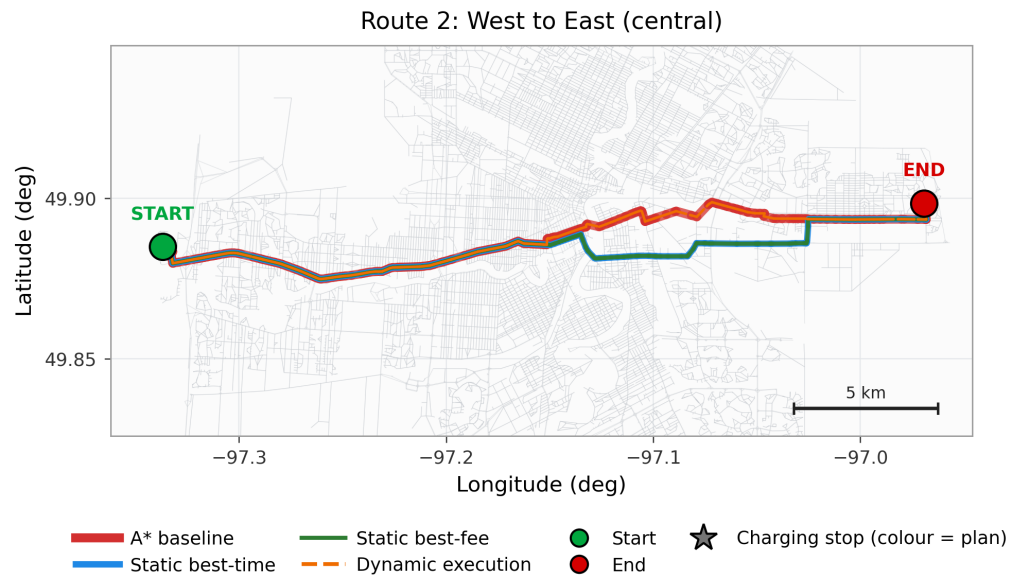


Figure 6.11: Optimal route map for Route 2

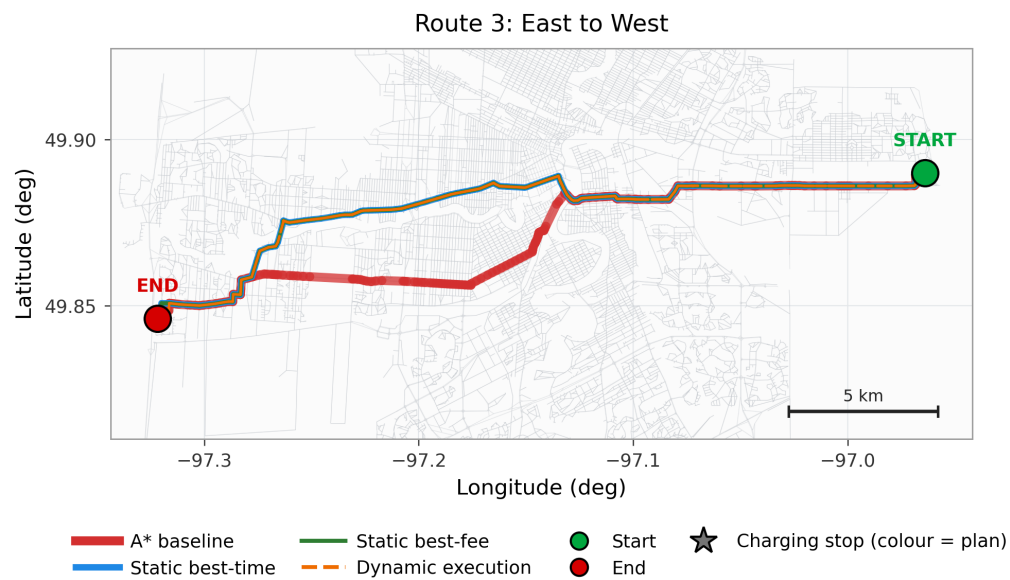


Figure 6.12: Optimal route map for Route 3

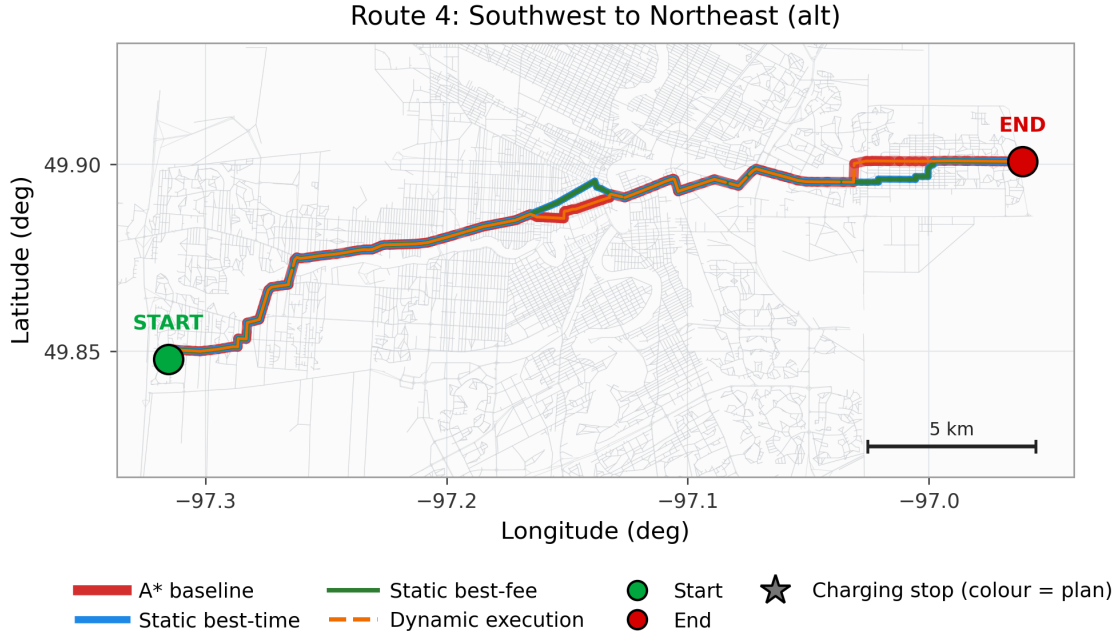


Figure 6.13: Optimal route map for Route 4. As with the other short routes, no charging is required.

6.3.6 Re-plan Trigger Breakdown

Table 6.12 reports the median trigger counts in all experimental runs.

Intersection replans account for 84% of all replan events. On this network, the road name changes between edges every few hundred meters, and the intersection trigger activates at nearly every block. Each intersection replan executes a 50-population, 20-generation NSGA-II optimization. The scheduled trigger activates every 4 to 5 events. The SoC threshold trigger rarely activates, as the static plan already identifies the correct charging stop, while the post-charging trigger activates

Trigger	Median count	Share of total
Intersection	29.5	84%
Scheduled	4.5	13%
Post charging	1.0	3%
Soc threshold	0.0	0%
Incident	0.0	0% (1.0 with incidents enabled)
Total	35.0	100%

Table 6.12: Re-plan trigger distribution across all experimental runs. The incident row reflects zero because most experiments ran without incidents enabled.

exactly once per trip, when the vehicle departs the station with a replenished SoC. Among the 33 intersection replans, 31 resulted in the same next node that the controller was already following, corresponding to a no-op rate of 93.9

Chapter 7

Discussion & Conclusion

This thesis developed a multi-objective, data-driven methodology for EV route planning under dynamic real-world conditions. The methodology addressed four key stages: data collection, travel time forecasting, charging station availability prediction, and route optimization. Each stage was evaluated against appropriate baselines using the Winnipeg road network.

This section summarizes our results and discusses key points from our experimentation. For road segment travel time forecasting, six volume prediction methods were compared using five-fold cross-validation across 367 monitored segments. KNN with network centrality features achieved the strongest performance, with an R^2 of 0.534, an MAE of 61.7 vph, and an RMSE of 97.0 vph. It outperformed both the spatial interpolation methods and the Gradient Boosting variants. The finding that centrality features improve KNN performance but degrade Gradient Boosting per-

formance confirms that feature effectiveness depends on the model using the feature rather than on the feature alone. The selected KNN model was then applied to predict traffic volumes on unmonitored road segments, extending coverage beyond the sensor network through network topology features. These predicted volumes were converted into travel time estimates using the Bureau of Public Roads (BPR) function, and Monte Carlo simulation was applied to the resulting estimates to forecast future travel times. These steps can be used to predict travel time from vehicle count on road segments.

For forecasting EV charging station availability across 86 Winnipeg stations over a 12-hour horizon, the sequential LSTM produces the strongest result, with an R^2 of 0.9504. The parallel LSTM follows with an R^2 of 0.9344, while completing training in 141 seconds across five regional clusters. SARIMA achieved the lowest performance with an R^2 of 0.6962, due to its single-station modeling scope.

For route optimization, the static plan is the initial route computed once by NSGA-II before departure, and the dynamic optimizer is that same route after it has been re-evaluated during travel. Across all five Winnipeg routes, static NSGA-II produces feasible plans that outperform A^* , while the dynamic optimizer executes these plans while re-planning against live congestion as needed. When no incident occurs along the initial route, both approaches achieve similar travel times, with the static plan maintaining a slight advantage because the initial route is already optimized and additional re-planning introduces intersection-trigger overhead without

changing the route. When an incident does occur, the dynamic optimizer performs better by adapting to conditions that are unavailable to the static plan. Charging only becomes relevant on routes long enough to drain the initial charge, so the fee objective doesn't exist on shorter routes, which reduce to single-objective time minimization. Future evaluation should stress the battery with longer routes or lower initial SoC conditions to better test the charging trade-off across the full route set rather than only the subset of routes that require charging.

7.1 Future Work

The availability prediction dataset covers a single season and excludes external covariates. Extending it to a full calendar year and incorporating weather and traffic data at the stations are the most promising approaches for improving prediction quality. On the congestion side, the R^2 of 0.534 indicates a limitation in the available feature information. Supplementary data on land use, commercial density, and additional vehicle count monitoring locations across the network would be needed to achieve further improvements.

Appendix A

Methods for Estimating Traffic Volumes at Unmonitored Locations

A.1 Inverse Distance Weighting (IDW)

Inverse Distance Weighting (IDW) is a deterministic spatial interpolation method that estimates values at unmonitored locations from nearby observations [43]. The method rests on the assumption that spatially proximate locations share more similar attribute values than distant ones, so each observation's contribution to an estimate diminishes with distance [44].

The estimated value at an unknown location x_0 is computed as a weighted average

of n surrounding observations:

$$\hat{V}(x_0) = \frac{\sum_{i=1}^n w_i(x_0) V(x_i)}{\sum_{i=1}^n w_i(x_0)} \quad (\text{A.1})$$

where $V(x_i)$ is the observed vehicle count at location x_i , and $w_i(x_0)$ is the weight assigned to that observation. Weights are defined as:

$$w_i(x_0) = \frac{1}{d(x_0, x_i)^p} \quad (\text{A.2})$$

where $d(x_0, x_i)$ is the distance between the prediction point and the observation, and p is the power parameter governing the rate at which influence decays with distance [43]. Higher values of p concentrate weight on the nearest observations and reduce the contribution of those further away.

A.2 Kriging

The kriging procedure used to estimate traffic volumes at unmonitored locations proceeds in two stages. A variogram is first estimated from the observed traffic counts, then used to compute the Ordinary Kriging weights at each unmonitored location.

The variogram $\gamma(h)$ is estimated directly from the available traffic counts:

$$\gamma(h) = \frac{1}{2} \text{Var}[V(x_i) - V(x_j)], \quad h = |x_i - x_j| \quad (\text{A.3})$$

It captures how dissimilarity between two observations grows with the distance h between them. At short distances, counts on nearby segments tend to move together, so $\gamma(h)$ stays small. A Gaussian variogram model is fitted to the data to smooth the empirical variogram and give a continuous description of spatial dependence across the network [37].

The fitted variogram is then used to solve for the Ordinary Kriging weights λ_i at each unmonitored location x_0 . The estimator takes the form [37]:

$$\hat{V}(x_0) = \sum_{i=1}^n \lambda_i V(x_i) \quad (\text{A.4})$$

where $V(x_i)$ is the observed count at monitored location x_i . The weights λ_i are found by minimizing prediction variance subject to the unbiasedness constraint $\sum_{i=1}^n \lambda_i = 1$ [37]. Since the weights come from the fitted variogram rather than a fixed distance rule, they reflect how correlated each neighboring segment is with the target location.

A.3 Gradient Boosting

Gradient Boosting treats vehicle count estimation as a regression problem, using road-level attributes as features rather than spatial position alone. The input feature vector x_i for each road segment comprises road type, lane count, posted speed limit, and spatial coordinates. The model is built as a sum of M shallow decision trees [38]:

$$\hat{V}(x) = \sum_{m=1}^M \gamma_m h_m(x) \quad (\text{A.5})$$

where $h_m(x)$ is the m -th tree and γ_m is its step size. At each iteration m , a new tree is fitted to the negative gradient of the loss function with respect to the current prediction [38]:

$$r_{im} = - \left[\frac{\partial L(V_i, \hat{V}_{m-1}(x_i))}{\partial \hat{V}_{m-1}(x_i)} \right] \quad (\text{A.6})$$

$$\hat{V}_m(x) = \hat{V}_{m-1}(x) + \gamma_m h_m(x) \quad (\text{A.7})$$

A.4 K-Nearest Neighbors(KNN)

K-Nearest Neighbors (KNN) is a non-parametric method that estimates a value at a query point from the observed values of its nearest neighbors in feature space, without assuming an underlying data distribution [39]. The model estimates continuous vehicle counts using a distance-weighted variant [40], which reduces the influence of distant outliers by computing the count at an unmonitored segment as an inverse distance-weighted average of the k most similar monitored segments:

$$\hat{z}(\mathbf{x}^*) = \frac{\sum_{i \in N_k(\mathbf{x}^*)} z_i / d_i}{\sum_{i \in N_k(\mathbf{x}^*)} 1 / d_i} \quad (\text{A.8})$$

where $\hat{z}(\mathbf{x}^*)$ is the predicted vehicle count at the unmonitored segment, $\mathbf{x}^*$ is its feature vector, and $N_k(\mathbf{x}^*)$ is the set of k nearest monitored segments in feature space. For each neighbor i , z_i is the observed vehicle count and d_i is the distance to $\mathbf{x}^*$ in feature space. Segments that are more similar to the query point contribute more to the prediction, while those further away in feature space are down-weighted accordingly.

Appendix B

Description of Road Capacity Standards and the BPR Function

B.1 Road Capacity Standards

Because the City of Winnipeg does not publish road segment capacity data at the level of detail required for this analysis, lane capacities were instead drawn from the Halton Region Transportation Master Plan [45], which documents typical per-lane capacities by functional road classification for Canadian urban roads. The values used are listed in Table B.1 and were applied uniformly across the network under normal traffic conditions.

Table B.1: Per-lane capacities and free-flow speeds used in this study

Road Classification	Capacity (vph/lane)	Free-flow Speed (km/h)
Provincial Freeway	1850	100
Regional Road	900	80
Major Arterial	850	60
Minor Arterial	750	50
Collector	450	30

B.2 Bureau of Public Roads (BPR) Function

The Bureau of Public Roads (BPR) function is a widely used volume-delay function that relates travel time on a road segment to how heavily that segment is used relative to its capacity. Current travel time for each road segment is estimated using this function [46], which models travel time as a non-linear function of the ratio of traffic volume to road capacity:

$$t = t_0 \left[1 + \alpha \left(\frac{v}{C} \right)^\beta \right] \quad (\text{B.1})$$

where:

- t is the estimated travel time on the segment,
- t_0 is the free-flow travel time under uncongested conditions,

- v is the traffic volume on the segment,
- C is the segment capacity,
- α and β are calibration parameters.

The BPR function gives the total travel time per segment, combining the free-flow travel time with an additional congestion term. At low volume-to-capacity ratios, the function returns a travel time close to t_0 . As the ratio approaches and exceeds 1, travel time increases sharply, reflecting the non-linear onset of congestion.

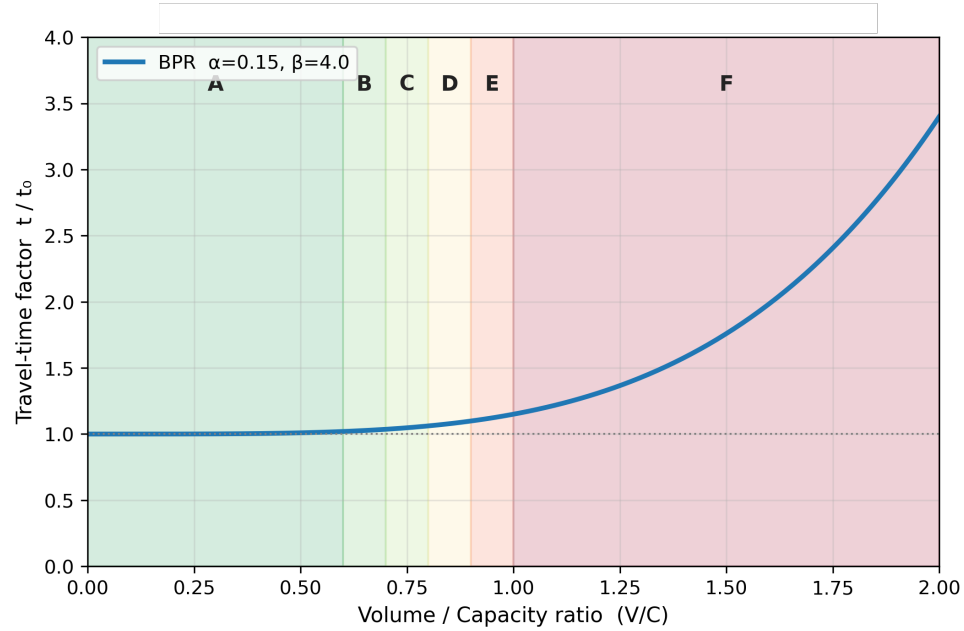


Figure B.1: BPR function with Level-of-Service bands.

Figure B.1 shows how travel time scales with the volume-to-capacity ratio under the BPR function. Travel time remains close to free-flow at low ratios and rises

sharply as the ratio approaches and exceeds 1.0.

B.3 Level of Service (LOS)

Level of Service (LOS) is a qualitative classification of traffic flow, derived from the volume-to-capacity ratio produced by the BPR function [47]. Six categories are defined, ranging from free-flow conditions at LOS A to over saturated flow at LOS F:

- LOS A: $V/C \leq 0.60$
- LOS B: $V/C \leq 0.70$
- LOS C: $V/C \leq 0.80$
- LOS D: $V/C \leq 0.90$
- LOS E: $V/C \leq 1.00$
- LOS F: $V/C > 1.00$

The bands corresponding to each LOS category are indicated in Figure B.1.

Appendix C

Description of Forecasting Models for EV Charging Station Availability

C.1 SARIMA (Seasonal ARIMA)

Seasonal Autoregressive Integrated Moving Average (SARIMA) is a classical time-series model that combines autoregressive, differencing, and moving-average components for both non-seasonal and seasonal patterns [48]. The model is denoted $\text{SARIMA}(p, d, q)(P, D, Q)_m$, where (p, d, q) are the non-seasonal orders and $(P, D, Q)_m$ are the seasonal orders with period m .

C.2 LSTM Model

Long Short-Term Memory (LSTM) networks are a class of recurrent neural network designed to model long-range temporal dependencies in sequential data, using gated units that regulate the flow of information across time steps [49]. The architecture used in this study consists of an LSTM layer followed by a fully connected output layer. The LSTM receives the input sequence $\mathbf{X} \in \mathbb{R}^{L \times F}$, where $F = S + 3$, and produces a hidden state $\mathbf{h}_t \in \mathbb{R}^d$ at each time step. The hidden state at the final time step, $\mathbf{h}_L$, is projected through a linear layer to a flattened output of length $S \times H$, which is reshaped to $\mathbb{R}^{H \times S}$. Each row of this output corresponds to one future time step, and each column corresponds to one station, giving predicted available connector counts for all S stations across the H -step forecast horizon.

The LSTM cell updates at each time step t are:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \quad (\text{C.1})$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \quad (\text{C.2})$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c) \quad (\text{C.3})$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \quad (\text{C.4})$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o) \quad (\text{C.5})$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \quad (\text{C.6})$$

where σ is the sigmoid function, $\odot$ denotes element-wise multiplication, and $[\cdot, \cdot]$

denotes concatenation.

The dense output layer is:

$$\hat{\mathbf{Y}} = \text{reshape}(\mathbf{W}_{\text{out}}\mathbf{h}_L + \mathbf{b}_{\text{out}}, [H, S]) \quad (\text{C.7})$$

C.3 Evaluation Metrics

Model performance is evaluated on the held-out test set using five metrics. All metrics are computed on predictions of the number of available connectors after inverse scaling, rounding, and clipping.

- **Mean Absolute Error (MAE):** The average absolute difference between predicted and actual counts.

$$\text{MAE} = \frac{1}{M} \sum_{i=1}^M |y_i - \hat{y}_i| \quad (\text{C.8})$$

- **Root Mean Squared Error (RMSE):** The square root of the mean squared error. RMSE penalizes large individual errors more than MAE, making it more sensitive to predictions that are far from the actual count.

$$\text{RMSE} = \sqrt{\frac{1}{M} \sum_{i=1}^M (y_i - \hat{y}_i)^2} \quad (\text{C.9})$$

- **Mean Absolute Percentage Error (MAPE):** Error expressed as a percentage of the actual value. Computed over the M' instances where $y_i \neq 0$ to

avoid division by zero.

$$\text{MAPE} = \frac{100}{M'} \sum_{\substack{i=1 \\ y_i \neq 0}}^{M'} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (\text{C.10})$$

- **Coefficient of Determination (R^2):** The proportion of variance in actual connector counts explained by the model.

$$R^2 = 1 - \frac{\sum_{i=1}^M (y_i - \hat{y}_i)^2}{\sum_{i=1}^M (y_i - \bar{y})^2} \quad (\text{C.11})$$

In all equations, M is the total number of predictions across all stations and time steps, and y_i and $\hat{y}_i$ are the actual and predicted available connector counts, respectively.

Appendix D

Description of the Multi-Objective Optimization Method

D.1 Genetic Algorithms

A Genetic algorithm (GA) is a population-based, stochastic optimization method inspired by natural selection [50; 51]. It works with a set of candidate solutions rather than a single point, and improves them over iterations called *generations*. The standard vocabulary is:

- **Individual/chromosome:** one candidate solution, encoded in a form the operators can manipulate.
- **Gene:** one element of a chromosome.

- **Population:** the set of individuals held at one time.
- **Fitness:** a measure of solution quality that drives selection.
- **Selection:** choosing which individuals reproduce, biased toward fitter ones.
- **Crossover:** combining two parents into offspring that inherit parts of each.
- **Mutation:** a small random change applied to an offspring, which maintains diversity and lets the search escape local optima.
- **Elitism:** carrying the best individuals unchanged into the next generation so good solutions are never lost.

The canonical GA loop is: initialize a population, evaluate fitness, then repeat {select parents, apply crossover and mutation, evaluate offspring, form the next generation} until a stopping criterion is met.

D.2 Non-dominated Sorting Genetic Algorithm II (NSGA-II)

NSGA-II (Non-dominated Sorting Genetic Algorithm II), introduced by [27], is a multi-objective evolutionary algorithm widely used for optimization problems with two or more conflicting objectives. Instead of reducing the objectives to a single score,

it searches for a set of Pareto-optimal solutions, each representing a different trade-off, so that no solution can be improved in one objective without worsening another. It sorts the population into fronts based on dominance, uses crowding distance to keep solutions spread out within each front, and carries the best solutions forward across generations through elitism. Algorithm 1 details this process.

Algorithm 2 NSGA-II

```

1:  $P \leftarrow \text{InitialPopulation}(start, end, t_0)$ 
2: for  $g = 1$  to  $G$  do
3:    $Q \leftarrow \emptyset$ 
4:   while  $|Q| < N$  do
5:      $p_1, p_2 \leftarrow \text{Tournament}(P), \text{Tournament}(P)$ 
6:      $c \leftarrow \text{Crossover}(p_1, p_2)$ 
7:     if  $\text{rand} < p_m$  then
8:        $c \leftarrow \text{Mutate}(c)$ 
9:     end if
10:     $c \leftarrow \text{RemoveLoops}(c)$ 
11:    if  $\text{IsValid}(c)$  then
12:       $Q \leftarrow Q \cup \{c\}$ 
13:    end if
14:  end while
15:   $S \leftarrow \text{Evaluate}(P \cup Q, t_0)$ 
16:   $P \leftarrow \text{BuildNextGen}(P \cup Q, S)$ 
17: end for
18: return Pareto front of  $P$ 

```

NSGA-II operates on a population of candidate routes. Each generation proceeds in three steps. First, solutions are ranked by fast non-dominated sorting [27]. A solution is assigned rank 1 if no other solution in the population is better on both

objectives simultaneously, rank 2 if it is dominated only by rank-1 solutions, and so on. Second, a crowding distance metric is computed within each rank to measure how isolated a solution is from its neighbors in objective space, and sparser solutions are preferred during selection to maintain spread across the Pareto front. Third, selection, crossover, and mutation operators produce the next generation of candidate routes, with selection preferring lower-rank solutions and, among equal ranks, greater crowding distance.

At convergence, the non-dominated solutions from the final generation form the Pareto front. Each point on this front represents a route that cannot be improved on one objective without worsening the other, and the driver selects from among them according to their preference for speed over cost or cost over speed.

D.2.1 Non-dominated Sorting and Population Update

After each generation, parent and offspring populations are combined and deduplicated by node sequence. The combined set is sorted into Pareto fronts using the standard $O(MN^2)$ fast non-dominated sort [27]. Fronts are added to the next generation in rank order until adding the next complete front would exceed the population cap, at which point a random subset of that front fills the remaining slots.

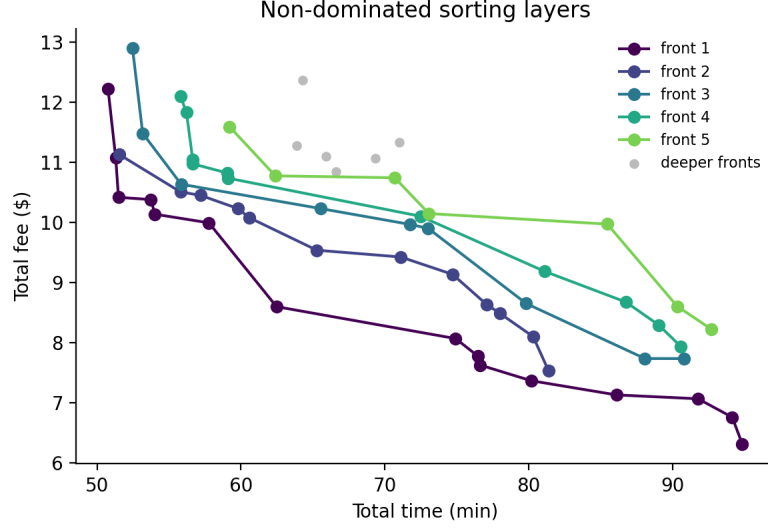


Figure D.1: Non-dominated sorting layers on a sample combined population. Each curve is one Pareto front. Deeper fronts are shown in grey.

D.3 Knee-Point Selection

A knee point is the solution on the Pareto front at which any further improvement in one objective comes only at a disproportionate cost in the other. On a two-objective front, this corresponds to the region of maximum curvature, the point farthest from the line connecting the two extreme solutions [52; 53].

Let the Pareto front contain N candidate plans, where plan i has evaluated travel time t_i (driving plus charging) and charging fee f_i . To place both objectives on a comparable scale, each is normalized to the unit interval by min–max scaling:

$$\tilde{t}_i = \frac{t_i - t_{\min}}{t_{\max} - t_{\min}}, \quad \tilde{f}_i = \frac{f_i - f_{\min}}{f_{\max} - f_{\min}}, \quad (\text{D.1})$$

where $t_{\min}, t_{\max}$ and $f_{\min}, f_{\max}$ are the extreme objective values over the current front. Let $A = (\tilde{t}_A, \tilde{f}_A)$ denote the best-time extreme (minimum $\tilde{t}$) and $B = (\tilde{t}_B, \tilde{f}_B)$ the best-fee extreme (minimum $\tilde{f}$). The perpendicular distance of each normalized plan $P_i = (\tilde{t}_i, \tilde{f}_i)$ from the chord $\overline{AB}$ is

$$d_i = \frac{|(\tilde{f}_B - \tilde{f}_A)\tilde{t}_i - (\tilde{t}_B - \tilde{t}_A)\tilde{f}_i + \tilde{t}_B\tilde{f}_A - \tilde{f}_B\tilde{t}_A|}{\sqrt{(\tilde{f}_B - \tilde{f}_A)^2 + (\tilde{t}_B - \tilde{t}_A)^2}}. \quad (\text{D.2})$$

The selected plan is the knee point, the candidate of maximum distance from the chord:

$$i^* = \arg \max_{1 \leq i \leq N} d_i. \quad (\text{D.3})$$

A knee point is well defined only when the front contains at least three distinct solutions. With two or fewer candidates, no interior trade-off exists, so the plan minimizing the sum of the normalized objectives is returned instead. The normalization in Equation D.1 is computed over the front currently under evaluation, so the geometry is relative to that front rather than to any absolute scale. As with any knee-based criterion, the selection favors balanced trade-offs. It will not return an extreme plan even when one objective is strongly preferred, which is the tradeoff of removing weights altogether.

This selection rule is applied the same way throughout the journey. Once to choose the initial plan at departure time t_0 , and again at every re-plan, where the optimization is re-run from the vehicle's current state and Equations D.1–D.3 are re-applied to the newly generated front.

Bibliography

- [1] International Energy Agency, “Global EV outlook 2023,” IEA, Paris, Tech. Rep., 2023. [Online]. Available: <https://www.iea.org/reports/global-ev-outlook-2023>
- [2] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, no. 1, p. 269–271, dec. 1959.
- [3] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.
- [4] C. Fiori, K. Ahn, and H. A. Rakha, “Power-based electric vehicle energy consumption model: Model development and validation,” *Applied Energy*, vol. 168, pp. 257–268, 2016.
- [5] E. K. Chong and S. H. Zak, *An introduction to optimization*. John Wiley & Sons, 2004.

-
- [6] K. Deb, *Multiobjective Optimization Using Evolutionary Algorithms*. Wiley, New York. Wiley, 01 2001.
- [7] S. Erdogan and E. Miller-Hooks, “A green vehicle routing problem,” *Transportation Research Part E*, vol. 48, no. 1, pp. 100–114, 2012.
- [8] Y. Y. Liu, “Multi-objective optimization on dynamic complex networks,” Doctoral Dissertation, University of Manitoba, Winnipeg, MB, Canada, 2022.
- [9] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston, MA: Springer US, jan. 1998.
- [10] M. Ehrgott, *Multicriteria optimization*. Springer Science & Business Media, 2005, vol. 491.
- [11] OpenStreetMap Contributors, “OpenStreetMap,” <https://www.openstreetmap.org>, 2025, accessed: 2025.
- [12] C. De Cauwer, J. Van Mierlo, and T. Coosemans, “Energy consumption prediction for electric vehicles based on real-world data,” *Energies*, vol. 8, no. 8, pp. 8573–8593, 2015.
- [13] Á. Felipe, M. T. Ortuño, G. Righini, and G. Tirado, “A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges,” *Transportation Research Part E: Logistics and Transportation Review*, vol. 71, pp. 111–128, nov 2014.

-
- [14] Y. Zhou, J. Huang, J. Shi, R. Wang, and K. Huang, “The electric vehicle routing problem with partial recharge and vehicle recycling,” *Complex & Intelligent Systems*, vol. 7, 02 2021.
- [15] S. Kancharla and G. Ramadurai, “Electric vehicle routing problem with non-linear charging and load-dependent discharging,” *Expert Systems with Applications*, 07 2020.
- [16] J. Paz, M. Granada-Echeverri, and J. Escobar, “The multi-depot electric vehicle location routing problem with time windows,” *International Journal of Industrial Engineering Computations*, pp. 123–136, 04 2018.
- [17] C. Li *et al.*, “An electric vehicle routing optimization model with hybrid plug-in and wireless charging systems,” *IEEE Access*, vol. 6, pp. 27 569–27 578, 2018.
- [18] K. Rajesh, E. Jain, and P. Kotecha, “A multi-objective approach to the electric vehicle routing problem,” *arXiv preprint arXiv:2208.12440*, 2022.
- [19] M. M. de Weerd, S. Stein, E. H. Gerding, V. Robu, and N. R. Jennings, “Intention-aware routing of electric vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1472–1482, 2016.
- [20] M. Akbari, “Electric vehicle traveling salesman problem with dynamic charging,” *Transportation Research Part C*, 2024.
- [21] Y. Y. Liu, P. Thulasiraman, and N. Pillay, “Bicriterion coevolution for the

- multi-objective travelling salesperson problem,” in *Proceedings of the 2022 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2022, pp. 1–8.
- [22] Open Charge Map Contributors, “Open Charge Map,” <https://openchargemap.org>, 2025, accessed: 2025.
- [23] ChargePoint, Inc., “EVSE: Electric vehicle (EV) charging stations,” 2026. [Online]. Available: <https://www.chargepoint.com/>
- [24] H. Technologies, “Here developer documentation,” 2026. [Online]. Available: <https://docs.here.com/>
- [25] N. Metropolis and S. Ulam, “The monte carlo method,” *Journal of the American Statistical Association*, vol. 44, no. 247, pp. 335–341, 1949. [Online]. Available: <http://www.jstor.org/stable/2280232>
- [26] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.
- [27] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: Nsga-ii,” *Evolutionary Computation, IEEE Transactions on*, vol. 6, pp. 182 – 197, 05 2002.
- [28] G. Boeing, “OSMnx: New methods for acquiring, constructing, analyzing, and

- visualizing complex street networks,” *Computers, Environment and Urban Systems*, vol. 65, pp. 126–139, 2017.
- [29] A. A. Hagberg, D. A. Schult, and P. J. Swart, “Exploring network structure, dynamics, and function using NetworkX,” in *Proceedings of the 7th Python in Science Conference (SciPy)*, 2008, pp. 11–15.
- [30] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [31] 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>
- [32] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2016. [Online]. Available: <https://www.tensorflow.org>
- [33] Digital Research Alliance of Canada. (2025) Digital research alliance of canada. [Online]. Available: <https://www.alliancecan.ca/en>
- [34] City of Winnipeg, “Open data portal,” City of Winnipeg, 2026, accessed: 2025. [Online]. Available: <https://data.winnipeg.ca/>
- [35] D. G. Krige, “A statistical approach to some basic mine valuation problems on

- the witwatersrand,” *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, vol. 52, no. 6, pp. 119–139, 1951.
- [36] G. Matheron, “Principles of geostatistics,” *Economic Geology*, vol. 58, no. 8, pp. 1246–1266, 1963.
- [37] H. Yang, J. Yang, L. D. Han, X. Liu, L. Pu, S.-M. Chin, and H.-L. Hwang, “A kriging based spatiotemporal approach for traffic volume data imputation,” *PLOS ONE*, vol. 13, no. 4, p. e0195957, 2018. [Online]. Available: <https://doi.org/10.1371/journal.pone.0195957>
- [38] J. Friedman, “Greedy function approximation: A gradient boosting machine,” *The Annals of Statistics*, vol. 29, 11 2000.
- [39] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.
- [40] K. Pathak, K. K, S. Madan, and J. Kapila, “Dw-knn: A transparent local classifier integrating distance consistency and neighbor reliability,” 11 2025.
- [41] Z. Shuangming, P. Zhao, and Y. Cui, “A network centrality measure framework for analyzing urban traffic flow: A case study of wuhan, china,” *Physica A: Statistical Mechanics and its Applications*, vol. 478, 03 2017.
- [42] Bureau of Public Roads, “Traffic assignment manual,” U.S. Department of Commerce, Urban Planning Division, Washington, D.C., Tech. Rep., 1964.

-
- [43] D. S. Shepard, “A two-dimensional interpolation function for irregularly-spaced data,” *Proceedings of the 1968 23rd ACM National Conference on -*, pp. 517–524, 1968.
- [44] G. Y. Lu and D. W. Wong, “An adaptive inverse-distance weighting spatial interpolation technique,” *Computers & Geosciences*, vol. 34, no. 9, p. 1044–1055, sept. 2008.
- [45] Regional Municipality of Halton. (2011) Transportation master plan to 2031: Appendix f5 – roadway planning capacity. Regional Municipality of Halton. [Online]. Available: <https://www.halton.ca/for-residents/infrastructure-and-growth/infrastructure-master-plans/transportation-master-plan-to-2031-the-road-to-c/appendix-f5-transportation-master-plan-to-2031>
- [46] AequilibraE Development Team. (2025) Volume delay functions. [Online]. Available: https://www.aequilibrae.com/latest/python/traffic_assignment/volume_delay_functions.html
- [47] Transportation Research Board, *Highway Capacity Manual*, 7th ed. Washington, D.C.: National Academies of Sciences, Engineering, and Medicine, 2022.
- [48] G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control*. San Francisco, CA: Holden-Day, 1970.

-
- [49] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: <https://doi.org/10.1162/neco.1997.9.8.1735>
- [50] J. H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.
- [51] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
- [52] J. Branke, K. Deb, H. Dierolf, and M. Osswald, *Finding Knees in Multi-objective Optimization*, 12 2004, pp. 722–731.
- [53] K. Deb and S. Gupta, “Understanding knee points in bicriteria problems and their implications as preferred solution principles,” *Evolutionary Computation, IEEE Transactions on*, vol. 43, 08 2010.