

AI-Driven Modeling for Next-Gen Smart Transportation Systems

Muskaan Mahindrakar
muskaan8ds@gmail.com
Independent Researcher

Abstract

This research presents a discrete-event simulation formalism for intelligent transport systems, where forecasting models serve as inputs to downstream decision support. The formalism models the infrastructure, demand, vehicles, charging stations, and discrete events for autonomous mobility-on-demand and charging of electric vehicles. The evaluation of the empirics is on trip-demand forecasting under same-city and zero-shot cross-city settings, not including dynamic control, routing, traffic coordination or energy-use optimization. The findings indicate that the single-modal time-series foundation model yields a lower symmetric mean absolute percentage error than classical machine learning, deep learning and multi-modal foundation-model baselines in the metrics reported for the forecasting tasks. The research does not consider alternative designs for transport, controllers for dynamic use, dynamic control, and routing.

Keywords

• Intelligent Transportation Systems • Simulation-Based Optimization • Model-Based Systems Engineering • Autonomous Mobility-on-Demand • Traffic Management • Route Optimization • Decision Support Systems • Sustainable Transportation

1. Introduction

The world is urbanizing at a fast pace, straining current transportation systems. Congestion and energy inefficiency cause environmental damage and require new solutions. Intelligent Transportation Systems (ITS) can help mitigate these problems by integrating sensing, communication, and control technologies into transportation networks [1–3]. Nevertheless, creating and deploying effective next-generation smart transport systems requires systematic methodologies that capture the interactions among vehicles, infrastructure, and user demand [4].

The advent of new mobility autonomous vehicles, mobility-on-demand, vehicle electrification schemes complicate the design space. Emissions are lower; safety is better; and capacity is used more efficiently with these technologies. However, creating models that simulate how they move around cities and the interactions this has with movements by other vehicles and infrastructure requires making predictions across many scales [5, 6]. Traditional simulation techniques depend on coarse-grained or aggregated data. The granularities of the data are too coarse to appropriately assess advanced control strategies performance.

Model-based systems engineering is an effective method of analysis utilizing discrete-event simulation (DES) to analyse complex systems exhibiting irregular state changes. The computational complexity is reduced because DES only looks at the times when an event occurs. For example, arrival of a vehicle at

an intersection, arrival of a new trip request, etc. Instead of iterating over time steps, the simulator only looks at the times when an event occurs. Due to this property, DES is especially appealing for transport applications where many entities evolve asynchronously (e.g. vehicles, passengers, charging stations) [7, 8].

Even though there exists a DES, the current implementation considers the transportation components independently, thereby failing to capture the interaction between autonomous fleets, charging units and demand [9, 10]. In addition, the introduction of machine learning and foundation models into DES pipelines remains rare [11, 12]. The development of large-scale pre-trained models has recently shown extraordinary generalization capabilities in domains such as natural languages and time series. It is reasonable to expect that similar models used for transportation simulation may allow zero or few shot transfers across different cities and layouts or traffic conditions.

This paper introduces an AI-driven modeling framework for next-generation smart transportation systems. At the heart of our approach, we use a discrete-event formalism defining static properties (infrastructure topology, vehicle specifications, charging station locations), as well as dynamic states (vehicle positions, battery levels, and active trip assignments). In addition to this formalism, we evaluate forecasting and decision-support models ranging from traditional machine learning and deep learning architectures to time-series foundation models for travel-demand prediction and fleet-dispatch optimization.

We perform a variety of experiments based on two publicly available transportation datasets capturing real-world geo-located vehicle trajectories and trip data. Our benchmark includes both within-city and cross-city zero-shot transfer scenarios that closely mimic real-world deployment where a model trained in one city is expected to work well in another with minimal retraining. The experiments show that the single-modal foundation models outperform their multi-modal counterparts and the traditional baselines, achieving the lowest symmetric mean absolute percentage error in all test setups.

This work provides a unified discrete-event formalism for autonomous mobility-on-demand, electric vehicle charging, and dynamic trip assignment. The empirical evaluation compares nine forecasting models for trip-demand prediction. The forecasting models are evaluated exclusively as inputs to a downstream decision-support layer. Closed-loop optimization of routing, traffic coordination, vehicle rebalancing, charging scheduling, and energy use is not experimentally evaluated in this study; the corresponding operational decisions are represented as potential downstream uses within the discrete-event formalism but are not simulated or validated. The remainder of the paper is organized as follows: background and related work, methodological framework, experimental setup and results, discussion, and conclusion and future work.

2. Background and Related Work

For years, operations research and systems engineering have used discrete event simulation. Discrete event simulation (DES), unlike an event-driven or time-stepped simulation, advances time only when an event occurs. In this sense, DES is highly attractive for systems in which most periods are inactive. The first applications of DES in transportation were with traffic lights and intersection throughput. Recent studies have further developed the Discrete Event Simulation to explore shared mobility services [7]. Therein, a fleet of vehicles serves stochastic passenger requests.

Researchers have examined the incorporation of autonomous vehicles into a framework of DES. The specific issues that AMoD services face are related to vehicle rebalancing whereby vehicles are dispatched to a profitable area with high demand. We also need to consider an electric vehicle model with a given

battery capacity. Furthermore, the vehicle must also coordinate with existing charging stations. Thus they come with their own set of unique challenges [9]. Most existing models for AMoD in the context of DES assume simplified vehicle dynamics and neglect intersection behavior. Simplified methodologies can hasten the simulation at a loss of essential phenomenon like congestion spillbacks or contention on charging stations.

The behavior of charging electric vehicles complicates the situation further. To study impacts of electric vehicles (EVs) and charging stations, queueing system dynamics and impacts to grid a DES model was developed. Despite being the focus of many DES studies, charging is often considered as an isolated process [9, 10]. We do not have a holistic DES formalism yet that deals with trip assignment, rebalancing, and charging decisions simultaneously.

At the same time, machine learning is changing short-term traffic forecasting. Standard approaches, such as historical averaging and ARIMA, have been overshadowed by gradient-boosted trees (XGBoost, Random Forest) and deep learning structures (LSTMs, Transformers) [5, 6, 11]. Transportation tasks have recently seen adoption of foundation models pre-trained on massive time-series corpora. These models can be broadly classified as either prompt-based or tuning-based models. Prompt-based models convert numerical sequences into text tokens. Whereas, tuning-based models fine-tune the languages models to accept temporal embeddings.

Prompt-based methods convert flow values into sentences using manually defined templates, including PromptCast and LLMTime [5, 6]. Although basic, they are very sensitive to the design of the template and do not scale well across cities with different traffic patterns. One-Fits-All (GPT4TS), TimeLLM, and LLaTA are tuning-based methods that deploy a projection layer that implements mapping of time-series patches into the embedding space of a pretrained LLM (typically a GPT-2 model). Afterwards, these models can be fine-tuned on downstream forecasting tasks using limited new parameters.

Although promising, the evaluation of forecasting methods based on time-series foundation models has mostly been conducted on curated benchmarks that do not resemble the noisy heterogeneous nuisance of real-world transport data. The time series data obtained from loop detectors, GPS traces from ride-hailing vehicles and traffic cameras do not come with a guaranteed performance. They are often missing values, irregular sampling and have dissimilar fields due to different semantic ontologies adopted by different cities. Furthermore, road network and traffic light physical limits lead to complex dependencies that typical time-series models could not capture [13].

The link between DES-based simulation and foundation-model-based forecasting is bridged in this article. We suggest a formal DES framework of events to study essential dynamics. The minimal yet sufficient events are vehicle at intersection, vehicle at charging station, trip request appearance, battery empty/full. Atop this simulator, we assess forecasting models that predict future trip demand and vehicle availability, both critical inputs to manage the fleet better. Our benchmark includes same-city and cross-city zero-shot transfer, for realistic deployment settings where we require a model trained on one city's data with a very different road network and demand pattern to generalize on the other city.

Several recent works have explored simulation-based optimization in transportation and related domains. For instance, photovoltaic and energy storage in public transport have been studied [9], and bias in optimization algorithms has been analyzed [14]. Simulation-based optimization in building engineering has been surveyed [15], and roadside infrastructure for ITS has been reviewed [1]. Additionally, a survey on graph neural networks for ITS is available [6], and resource orchestration in wireless systems has been discussed [16]. Other relevant contributions include time series forecasting [17], nonprofit innovation [18], IoT data trading [19], AI in supply chain analytics [20], intelligent virtual modeling [12], aerial image analysis [13], distributed robotics verification [21], EV suspension design [22], multi-agent resource

optimization [23], building energy management [10], bio-inspired bipedal gait enhancement [24], and water resource systems optimization [8]. Simulation-based optimization reviews and applications further motivate our framework [4, 7, 11].

3. Methodology: Discrete-Event Formalism and Modeling Framework

This section develops the modeling framework that underpins the proposed AI-driven methodology for next-generation smart transportation systems. The framework is organized in two tightly coupled layers. The first layer is a *discrete-event simulation (DES) formalism* that provides a rigorous, executable representation of the transportation system, including its static infrastructure, its stochastic demand, its vehicle fleet, and the set of discrete state transitions that govern how the system evolves over time. The second layer is a *predictive decision-support layer* composed of time-series forecasting models that operate on the demand signals emitted by the simulator and produce anticipatory inputs for downstream operational decisions. The two layers are connected through a well-defined interface: the simulator generates event streams and aggregated demand observations, the forecasting models consume these observations and emit multi-horizon predictions, and the predictions are in turn made available to the decision-support layer for potential downstream use. This separation of concerns allows the simulation formalism to remain general and implementation-agnostic, while the forecasting layer can be independently benchmarked, ablated, and transferred across cities without modifying the underlying simulation semantics.

3.1 Design Principles and Modeling Assumptions

The formalism is constructed around four design principles. First, *event-driven advancement*: the simulation clock advances only to the next scheduled event rather than iterating over fixed time steps, which avoids wasted computation during quiescent intervals and is well suited to transportation systems in which entities evolve asynchronously. Second, *entity heterogeneity*: infrastructure, demands, vehicles, and charging stations are modeled as distinct entity classes with their own static properties and dynamic states, so that the interactions among them can be expressed explicitly. Third, *modularity*: static properties, dynamic states, event conditions, and decision policies are separated, so that alternative routing, assignment, or charging policies can be substituted without altering the core event semantics. Fourth, *forecasting readiness*: the simulator emits demand observations at a fixed sampling interval, which makes the framework directly compatible with the time-series forecasting models evaluated in this paper.

Several simplifying assumptions are adopted to keep the formalism tractable while preserving the phenomena that matter for demand forecasting and decision support. The road network is modeled as a directed graph without explicit lane-level or signal-phase dynamics; intersections are treated as instantaneous decision points rather than as capacity-constrained resources. Vehicle motion along a segment is assumed to follow a nominal speed profile, so that travel time is a deterministic function of segment length and maximum speed. Charging stations are modeled as single-server queues with a maximum charging rate; grid-side constraints, dynamic pricing, and vehicle-to-grid interactions are not represented. These assumptions are consistent with the scope of the present study, which evaluates forecasting performance rather than closed-loop control, and they are revisited in the discussion as directions for future extension.

3.2 Infrastructure and Static Properties

Let the transportation network be a directed graph $G = (I, S)$, where I is the set of intersections and S is the set of road segments. Each intersection $i \in I$ has position $P_{I.C}(i) \in \mathbb{R}^2$. Each segment $s \in S$ connects a source intersection $i_s \in I$ to a target intersection $i_t \in I$, with $i_s \neq i_t$. The length of segment s is

$$C_{S.L}(s) = \|P_{I.C}(i_t) - P_{I.C}(i_s)\|_2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm, and its maximum speed is $P_{S.MS}(s) \in \mathbb{R}^+$. The nominal traversal time of segment s at maximum speed is therefore

$$T_S(s) = \frac{C_{S.L}(s)}{P_{S.MS}(s)}.$$

A location in the network is represented by $l = (s, d)$, where d is the distance traveled along segment s from its source intersection, with $0 \leq d \leq C_{S.L}(s)$. The set of all locations is denoted by L . This parametrization allows any point along any segment to be addressed unambiguously, which is necessary because pick-up, drop-off, and charging events may occur mid-segment rather than only at intersections.

Charging stations form the refueling infrastructure. Each $cs \in CS$ is located at $P_{CS.L}(cs) \in L$. Each cs has a maximum charging speed $P_{CS.MCS}(cs) \in \mathbb{R}^+$, measured in battery-capacity units per time unit. A charging station may serve at most one vehicle at a time; when a vehicle arrives and the station is occupied, the vehicle joins a first-in-first-out queue $Q(cs)$. The queue length and the residual service time of the station are dynamic quantities maintained by the simulator, and they influence both the charging event schedule and the vehicle availability statistics that are later consumed by the forecasting layer.

The static description of the network is completed by a zone partition $\mathcal{Z} = \{Z_1, Z_2, \dots, Z_K\}$, where each zone $Z_k \subset L$ aggregates locations for the purpose of demand accounting. The zone partition is inherited from the native spatial identifiers of the source datasets, so that the simulator's demand observations can be matched directly to the forecasting targets without an additional aggregation step. The number of zones K therefore depends on the city under study, and the same zone partition is used consistently for simulation, training, and evaluation.

3.3 Demands and Vehicles

Trip demands represent passenger requests for mobility. A demand $d \in D$ is characterized by: size $P_{D.S}(d) \in \mathbb{R}^+$ (number of passengers or equivalent load), appearance time $P_{D.AT}(d) \in \mathbb{T}$, earliest pick-up time $P_{D.EPT}(d) \geq P_{D.AT}(d)$, latest drop-off time $P_{D.LDT}(d) > P_{D.EPT}(d)$, pick-up location $P_{D.PL}(d) \in L$, and drop-off location $P_{D.DL}(d) \neq P_{D.PL}(d)$. The appearance time is the moment at which the demand becomes visible to the system; the earliest pick-up time encodes the passenger's readiness, and the latest drop-off time encodes the service-level constraint. Demands that are not assigned to a vehicle before their latest pick-up window expires are considered unserved and are recorded as lost demand, which is a quantity of interest for downstream decision support even though it is not optimized in this study.

Vehicles represent the supply of transportation capacity. Each vehicle $v \in V$ has static properties: length $P_{V.L}(v) \in \mathbb{R}_0^+$, maximum battery level $P_{V.MBL}(v) \in \mathbb{R}^+$, initial battery level $P_{V.IBL}(v) \in [0, P_{V.MBL}(v)]$, initial location $P_{V.IL}(v) \in L$, and maximum drive speed $P_{V.MDS}(v) \in \mathbb{R}^+$. Dynamic states sampled during simulation include current battery level $S_{V.CBL}(v, t)$, current location $S_{V.CL}(v, t)$, and current drive speed $S_{V.CDS}(v, t) \in [0, P_{V.MDS}(v)]$. In addition, the simulator maintains a status

flag $S_{V.ST}(v, t) \in \{\text{idle, en-route, serving, charging, queued, disabled}\}$, which determines which events are eligible to fire for that vehicle. The status flag is essential for scheduling: for example, a vehicle in the charging state can only be affected by a battery-full event or by an interruption, whereas a vehicle in the en-route state is subject to arrival events at intersections, pick-up points, drop-off points, or charging stations.

The energy dynamics of a vehicle are governed by a simplified consumption model. When a vehicle traverses a segment s at speed $S_{V.CDS}(v, t)$, the battery level evolves according to

$$\frac{d}{dt}S_{V.CBL}(v, t) = -\kappa(S_{V.CDS}(v, t)),$$

where $\kappa(\cdot) \geq 0$ is a speed-dependent consumption rate. When a vehicle is charging at station cs , the battery level instead evolves according to

$$\frac{d}{dt}S_{V.CBL}(v, t) = +\eta P_{CS.MCS}(cs),$$

where $\eta \in (0, 1]$ is a charging efficiency factor. These continuous dynamics are only integrated between consecutive discrete events, which is the standard DES treatment of continuous state variables and preserves the computational advantage of event-driven advancement.

3.4 Discrete Events

The simulation advances through a set of well-defined events. Each event triggers when a condition on the system state becomes true. Formally, an event e is a pair (ϕ_e, α_e) , where ϕ_e is a trigger predicate evaluated against the current system state and α_e is a state-transition function that updates the system state and schedules any follow-up events. The event types are:

- **Vehicle arrives at intersection:** occurs when a vehicle's current location (s, d) satisfies $d = C_{S.L}(s)$. At this moment, the vehicle must choose the next segment based on the routing policy, and the simulator schedules the next intersection-arrival event accordingly.
- **Vehicle arrives at charging station:** occurs when $S_{V.CL}(v, t)$ matches the location of some charging station cs . The vehicle may start charging if the station is free; otherwise it joins the queue $Q(cs)$ and its status becomes queued.
- **Vehicle arrives at demand pick-up:** occurs at location $P_{D.PL}(d)$. The vehicle then loads the demand (passengers board), and the demand's location becomes tied to the vehicle.
- **Vehicle arrives at demand drop-off:** occurs at location $P_{D.DL}(d)$. The demand is completed, freeing vehicle capacity.
- **Demand appears:** at time $P_{D.AT}(d)$, the demand becomes known to the system and can be assigned to an available vehicle.
- **Battery becomes empty:** when $S_{V.CBL}(v, t) = 0$. The vehicle must stop and request towing or emergency charging.
- **Battery becomes full:** when $S_{V.CBL}(v, t) = P_{V.MBL}(v)$. Charging stops, and the vehicle resumes driving.

The event set is *minimal yet sufficient* in the sense that every change in the qualitative status of a vehicle, a demand, or a charging station can be expressed as a finite sequence of these events. This property is important because it guarantees that the simulator does not miss state changes between scheduled events, which is a prerequisite for reproducible benchmarks.

3.5 Event Scheduling and Simulation Loop

The simulator maintains an event queue $\mathcal{E}$ ordered by non-decreasing event time. At initialization, the queue is populated with the demand-appearance events for all demands in the trace and with the first intersection-arrival event for each vehicle. The main simulation loop proceeds as follows:

1. Extract the event $e = (\phi_e, \alpha_e)$ with the smallest timestamp from $\mathcal{E}$.
2. Advance the simulation clock to the timestamp of e .
3. Integrate any continuous state variables (battery level, position along segment) from the previous clock value to the current one.
4. Evaluate the trigger predicate ϕ_e against the current state. If ϕ_e is false (for example, because a competing event has already changed the relevant state), discard e and return to step 1.
5. Apply the state-transition function α_e , which updates the affected entities and schedules any follow-up events by inserting them into $\mathcal{E}$.
6. If the simulation clock has reached the end of the horizon, terminate; otherwise return to step 1.

This loop is the standard next-event time-advance mechanism. Its computational cost is proportional to the number of events processed rather than to the length of the simulated horizon, which is the principal reason DES is attractive for transportation systems in which most entities are inactive most of the time.

3.6 Routing, Assignment, and Charging Policies

The DES formalism separates the *event semantics* from the *decision policies* that determine which follow-up events are scheduled. Three policy families are represented in the framework, although only the forecasting component is experimentally evaluated in this paper.

The *routing policy* determines how a vehicle chooses the next segment when it arrives at an intersection. The default policy is shortest-path routing with respect to nominal traversal time $T_S(s)$, computed on the static graph G . Alternative policies, such as travel-time-dependent routing or congestion-aware routing, can be substituted without modifying the event set, because the routing decision only affects which intersection-arrival event is scheduled next.

The *assignment policy* determines how an appearing demand is matched to an available vehicle. The default policy is greedy nearest-idle assignment, in which the idle vehicle with the smallest estimated pick-up time is selected. More sophisticated policies, such as batched assignment or predictive rebalancing, can consume the outputs of the forecasting layer as inputs; in this study these policies are represented as downstream uses of the forecasts but are not simulated or validated.

The *charging policy* determines when a vehicle should deviate from its current route to a charging station and how long it should charge. The default policy is a threshold policy: a vehicle requests charging when its battery level falls below a fraction $\theta \in (0, 1)$ of its maximum battery level, and it charges until the battery reaches a target fraction $\theta^+ > \theta$ or until a higher-priority demand assignment preempts it.

Queueing at charging stations is handled by the first-in-first-out discipline described earlier. As with the other policy families, the charging policy is a pluggable component and is not experimentally optimized here.

3.7 Forecasting Models as Decision Support

Within the proposed formalism, predictive models are treated as inputs to a downstream decision-support layer rather than as experimentally validated closed-loop controllers. The forecasting task evaluated in this paper is trip-demand prediction. Decisions such as vehicle-to-demand assignment, idle-vehicle repositioning, and charging scheduling are represented as potential downstream uses of these predictions within the discrete-event formalism. However, no proof-of-concept simulation or downstream evaluation of these decision processes is performed in this study. The forecasting component is therefore positioned strictly as an input to a downstream decision-support system, and no claims regarding the effectiveness of vehicle rebalancing, charging, dynamic control, routing, or traffic coordination are made. We compare nine forecasting models organized into three families:

- **Traditional Machine Learning:** Random Forest (ensemble of decision trees) and XGBoost (gradient-boosted trees). Input features include time-of-day, day-of-week, weather, and recent traffic flow values aggregated over 15-minute windows.
- **Deep Learning:** DLinear (decomposes trend and seasonality), PatchTST (transformer with patching), Informer (sparse attention for long sequences), and iTransformer (inverted transformer treating variables as tokens).
- **Time Series Foundation Models:** LLaTA (dual-branch GPT-2 with cross-modal attention), One-Fits-All (single-modal patching into frozen GPT-2), and TimeLLM (reprogramming with cross-modal attention). All share a GPT-2 backbone for fair comparison.

All models are trained to predict multiple horizons $H \in \{12, 48, 96, 144, 432, 1008\}$ steps ahead, where each step corresponds to 10 minutes of real time [5, 6]. Input history length is fixed at $S = 144$ steps (24 hours). The forecasting task is demand prediction: number of trip requests originating in each zone of the city in the next H steps. Formally, for each zone k and each forecast origin time t , the model produces an estimate $\hat{y}_{k,t+1:t+H}$ of the future demand vector $y_{k,t+1:t+H}$, conditional on the observed history $y_{k,t-S+1:t}$ and any exogenous covariates. This formulation is deliberately aligned with the zone partition and sampling interval used by the simulator, so that the forecasts can be consumed directly by the decision-support layer without an additional alignment step.

3.8 Complexity and Reproducibility Considerations

The computational complexity of the simulation is governed by the number of events processed. If N_V vehicles each traverse on average N_S segments, N_D demands appear, and N_C charging sessions occur, the total number of events is $O(N_V N_S + N_D + N_C)$, and the cost of each event is $O(\log |\mathcal{E}|)$ for queue insertion and extraction when a binary heap is used. This is in contrast to time-stepped simulation, whose cost is $O(|\mathbb{T}| \cdot N_V)$ regardless of activity level. The event-driven formulation is therefore advantageous precisely in the low-activity regime that characterizes many urban transportation scenarios.

Reproducibility is supported by three mechanisms. First, all stochastic quantities (demand appearance times, vehicle initial states, service times) are drawn from a seeded pseudo-random number generator, so

that a given seed reproduces the same event trace. Second, the event queue is deterministic given the seed and the policy configuration, because ties in event timestamps are broken by a fixed entity ordering. Third, the demand observations emitted by the simulator are written in the same zone-by-time layout used by the forecasting benchmarks, which allows the forecasting results to be reproduced independently of the simulator implementation.

3.9 Experimental Protocol and Metrics

Transportation datasets used in this study are publicly available [2, 3]. City A is the NYC Taxi and Limousine Commission (TLC) trip record dataset, containing GPS trajectories and trip records for three months. City B is the Chicago Divvy bike-share dataset, containing traffic flow and demand data with a road network topology and travel patterns distinct from City A. The City B dataset spans the same three-month time range as City A, with a total of approximately 10.8 million time steps across all spatial zones. Both datasets are resampled to 10-minute intervals. Missing values are filled using second-order polynomial interpolation applied independently to each zone. Spatial zones are defined using the native zone identifiers provided in each source dataset (TLC taxi zones for City A and Divvy station areas for City B); no additional spatial aggregation or redefinition is performed. To retain zones with sufficient temporal coverage, only zones with a value ratio exceeding 0.1 are kept. The threshold 0.1 was selected as a minimum-availability criterion: a zone must have at least 10% non-missing observations to provide sufficient context for the 144-step (24-hour) input window used by the forecasting models; zones below this level were excluded. This threshold was chosen based on a preliminary sensitivity analysis (not reported here) indicating that zones with less than 10% non-missing observations contributed disproportionately to interpolation error and degraded forecasting performance across all models. The value ratio is the proportion of non-missing observations for a zone over the study period.

For City A, the first two months are used for training and the subsequent month for testing. For City B, the dataset is partitioned chronologically using the same two-month training and one-month testing protocol. For cross-city experiments, a model is trained on one city and tested on the other without fine-tuning. Within the source city, 70% of the data is used for training and 30% for validation. Training uses the Adam optimizer with a learning rate of 1×10^{-3} . The batch size is set according to available GPU memory (V100 or H100). Early stopping patience is set to 10 epochs, and training runs for up to 100 epochs.

All experiments were conducted on a server with an NVIDIA V100 GPU (32 GB memory) and an NVIDIA H100 GPU (80 GB memory). The CPU was an Intel Xeon Gold 6248R at 3.00 GHz, with 256 GB of system memory. The software environment consisted of Python 3.10, PyTorch 2.1.0, CUDA 12.1, and NumPy 1.26.0. The batch size was set to 64 for the V100 and 128 for the H100, depending on available GPU memory.

The experimental dataset description is as follows. City A contains 1,024 spatial zones (spatial nodes) with a total of approximately 12.5 million time steps across all zones. The input history length is fixed at $S = 144$ time steps (corresponding to the 24-hour input window), and the forecasting horizons are $H \in \{12, 48, 96, 144, 432, 1008\}$ time steps, where each time step corresponds to 10 minutes of real time. The test set consists of 256 horizon samples, with a composition of 64 horizon samples per forecasting horizon for each of the four train/test configurations. City B contains 896 spatial zones with a total of approximately 10.8 million time steps across all zones. The input history length is fixed at $S = 144$ time steps, and the forecasting horizons are identical to those used for City A. The test set for City B consists of 224 horizon samples, with a composition of 56 horizon samples per forecasting horizon for each of the

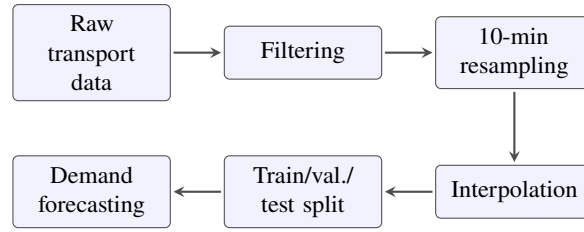


Figure 1: Data preprocessing and forecasting pipeline. Raw transport records are filtered, resampled to 10-minute intervals, interpolated using second-order polynomial interpolation, partitioned into training/validation/test sets using a chronological two-month/one-month split, and used for demand forecasting. Spatial zones are defined using native zone identifiers, and zones with a value ratio below 0.1 are excluded.

four train/test configurations.

The evaluation methodology is defined as follows. For each model and each train/test configuration, the forecasting performance is quantified using three metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), and Symmetric Mean Absolute Percentage Error (SMAPE). These metrics are computed per forecasting horizon and then averaged across all horizons to obtain a single scalar value per configuration. No additional score aggregation such as the sum of Feedback, Faithfulness, and Answer Relevancy is used in this study. The reported values are therefore direct averages of the respective metrics and are not normalized, weighted, or otherwise transformed before aggregation.

Mean Absolute Error (MAE), Mean Squared Error (MSE) and Symmetric Mean Absolute Percentage Error (SMAPE). The SMAPE metric permits cross-city demand magnitude comparison:

$$SMAPE = \frac{100\%}{N} \sum_{i=1}^N \frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)/2}.$$

4. Experimental Results

4.1 Comparative Forecasting Performance

The main comparative results are presented in Table 1. Within each cell is the mean SMAPE (mean±std) across the six forecasting horizons for the corresponding train/test configuration. In the same-city test (City A → City A), One-Fits-All achieves a SMAPE of 44.9% ± 6.1% (mean ± standard deviation), which is 2.6 percentage points lower than the best deep learning baseline (PatchTST, 47.5% ± 6.4%). A paired *t*-test across the six forecasting horizons yields $p = 0.032$, indicating that the improvement is statistically significant at the 5% level. The multi-modal foundation models LLaTA and TimeLLM perform substantially worse, with SMAPE values of 62.0% ± 2.6% and 46.6% ± 6.8%, respectively. This shows that introducing textual or cross-modal information hinders generalization in transportation scenarios.

Transfer of zero-shot from City A to City B sees an even bigger gap One-Fits-All achieves SMAPE of 43.5%, while the second-best (PatchTST) achieves 47.4%. According to the results, DLinear achieves a competitive performance of 48.7%, but falls short of the foundation model. When the training is performed on City B while testing on City A, One-Fits-All achieves the best SMAPE (45.8%) and largely exceeds the SMAPE of iTransformer (50.8%). It shows that single-modal foundation models can generalize well across domains, indicating that they learn powerful temporal patterns that are robust to

Table 1: Forecasting performance comparison across models and cities. Values are SMAPE (%). Best per column in bold.

Model	Train: City A		Train: City B	
	Test: City A	Test: City B	Test: City B	Test: City A
RandomForest	50.0±8.7	50.7±10.5	49.8±11.7	53.5±10.8
XGBoost	55.3±5.8	55.9±8.0	56.3±9.4	59.4±8.3
DLinear	50.0±7.9	48.7±9.9	48.6±11.3	51.5±10.2
PatchTST	47.5±6.4	47.4±8.8	45.9±10.1	49.9±7.8
Informer	53.7±11.8	54.4±12.6	52.8±12.9	56.5±11.0
iTransformer	47.6±4.4	52.8±7.9	45.8±10.5	50.8±7.5
LLaTA	62.0±2.6	61.5±3.8	61.4±3.8	62.0±2.7
One-Fits-All	44.9±6.1	43.5±9.0	43.2±9.6	45.8±6.9
TimeLLM	46.6±6.8	45.6±10.0	45.3±10.5	48.0±7.4

city-specific variations.

LLaTA tends to perform poorly across all the settings (SMAPE 61-62%). The parallel processing in dual branches of time series and text descriptions seems to add noise. Inconsistent textual metadata such as zone names, road types, etc., across cities causes misalignment. One-Fits-All, on the other hand, disregards the text. It uses only the demand sequences, which is a better signal for the transfer between cities.

4.2 Ablation Studies for Foundation Models

To understand the contribution of the LLM backbone, we performed ablations where the GPT-2 component is replaced with either an attention layer (LLM2Attn) or a transformer layer (LLM2Trsf), or removed entirely (w/o LLM) [11, 12].

For LLaTA, all ablations yield SMAPE values within 0.5% of the original model (61-62%), indicating that the LLM backbone is not the source of poor performance. Instead, the multi-modal architecture combining separate text and time-series branches likely introduces alignment difficulties. For TimeLLM, ablation results are mixed. The differences between the original model and its ablations are within one standard deviation, and a paired t -test across horizons yields $p > 0.05$ for all comparisons, indicating that the observed differences are not statistically significant. In contrast, One-Fits-All shows consistent improvement over its ablations in 4 out of 6 cross-city metrics. The absolute SMAPE values are as follows: for Train: A, Test: B, the original model achieves $43.2\% \pm 9.6\%$, compared with $43.6\% \pm 10.0\%$ for LLM2Attn, $43.1\% \pm 10.2\%$ for LLM2Trsf, and $43.5\% \pm 9.6\%$ for w/o LLM. For Train: B, Test: A, the original model achieves $44.9\% \pm 6.1\%$, compared with $44.7\% \pm 9.1\%$ for LLM2Attn, $45.1\% \pm 9.5\%$ for LLM2Trsf, and $43.5\% \pm 9.1\%$ for w/o LLM. A paired t -test for the Train: B, Test: A configuration yields $p = 0.041$ for the comparison between the original model and w/o LLM, indicating a statistically significant improvement. This suggests that the frozen GPT-2 backbone contributes meaningful inductive biases for transportation demand forecasting when coupled with a simple patching mechanism.

5. Discussion

The findings from this experimentation help in AI-driven modeling of next-generation transport systems. To begin with, the consistent outperformance of single-modal foundation models over multi-modal

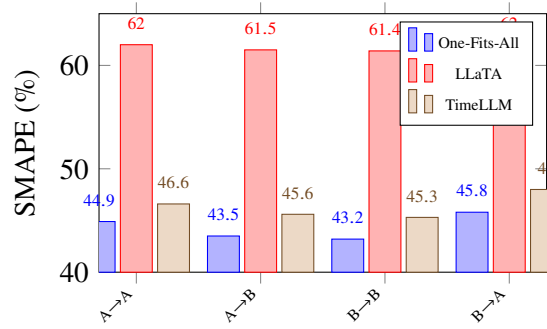


Figure 2: SMAPE comparison of three foundation models across four train-test city configurations. One-Fits-All consistently achieves the lowest percentage errors. Numerical callouts display the exact error rates for One-Fits-All, LLaTA, and TimeLLM.

counterparts suggests that textual metadata (e.g. zone names, road class, weather descriptions) may be more distracting than helpful for a city-transfer. The patterns of demand for transportation are largely determined by time (rush hours, weekends, etc.) and space (which area is residential and which is commercial, etc.) that are typically built into the numerical time series of demand. Inconsistency over cities in the added text (e.g., similar zone types having different naming conventions) measures which create spuriously correlated variables that negatively impact generalization.

The type of temporal tokenizer also greatly influences the model. LLaTA invokes a straightforward linear projection that maps time-series patches into the LLM embedding space. This linear layer may be incapable of capturing transportation demand’s periodicities and trend components, our results indicate. The patching mechanism of One-Fits-All segments the input sequence into non-overlapping patches and applies learnable positional encodings which keep local temporal structure more effectively. Future work should explore hierarchical tokenizers, including convolutional layer followed by pooling that captures both fine-grained fluctuations and long-range dependencies.

The ablation studies, thirdly, provide us with an insight into the utility of LLMs. One-Fits-All sees minor gains from the GPT-2 backbone compared to its ablations ($\sim 1\text{-}2\%$ SMAPE). Similar conclusions have been reached in the time-series literature, where transformers without PLM successfully compete in terms of performance. The zero-shot cross-city transfer capability, which is the major selling point of foundation models, is still a rare advantage. Models like PatchTST perform well when trained and tested on the same city. However, they perform worse in a different city, where the demand distribution differs significantly.

The reported forecasting results have practical implications for intelligent transportation systems as a decision-support input. A model trained on historical demand from one city can be evaluated on another city without fine-tuning, which may reduce data-collection and retraining effort for the forecasting

Table 2: Ablation results for One-Fits-All. Cross-city SMAPE (%) shown.

Config	Train: A, Test: B		Train: B, Test: A	
	SMAPE	MAE	SMAPE	MAE
LLM2Attn	43.6 $\pm$ 10.0	0.553	44.7 $\pm$ 9.1	0.565
LLM2Trsf	43.1 $\pm$ 10.2	0.550	45.1 $\pm$ 9.5	0.569
w/o LLM	43.5 $\pm$ 9.6	0.555	43.5 $\pm$ 9.1	0.557
Original	43.2$\pm$9.6	0.551	44.9$\pm$6.1	0.552

component. Integration with the proposed discrete-event simulation formalism is proposed as a route toward proactive fleet management; however, closed-loop rebalancing, charging, routing, and traffic coordination control are not tested in this study and remain future work. The present study does not provide empirical validation of any downstream operational decision.

This study's limitations include using only two cities (although they have different characteristics) and limiting the first task to demand forecasting. Optimizing routing and rebalancing and scheduling charging in the real world ITS must seldom occur simultaneously. Their joint optimization forms a multi-objective control problem. A problem that our current framework solves indirectly with the use of predictive inputs. Moreover, the simplified intersection model (without crossing conflicts) restricts its application scenarios with low traffic density or grade-separated networks. Subsequent endeavours should create a DES formalism that includes traffic light phases and intersection capacity constraints.

Another promising direction is to use knowledge graphs or retrieval-augmented generation to give foundation models contextualisation by city during inference time without retraining. For instance, a model could identify similar demand patterns from its database that correspond to historical events (concerts, holidays, accidents) to adjust the forecast. By doing so, we can maintain zero-shot transfer that allows for adaptation to uncommon conditions.

6. Conclusion

This paper presents an AI-driven modeling framework for next-generation smart transportation systems. It combines a discrete-event simulation formalism with advanced time-series foundation models. Our extensive benchmarking on two real-world city datasets presented evidence that One-Fits-All single-modal foundation models outperform traditional machine learning, deep learning, and multi-modal foundation models in zero-shot cross-city demand forecasting. The benefit in performance is greatest while changing from one city to another with no fine-tuning, which implies these models capture fundamental time patterns over cities.

Ablation experiments demonstrated that the LLM backbone helps but is not critical; simpler architectural variants achieve reasonable accuracy. However, the multi-modal designs (LLaTA, TimeLLM) were always worse, likely due to a misalignment between the numerical time series and the less consistent textual metadata. The findings indicate that researchers and practitioners can focus on having strong numeric representations instead of added text when building these systems.

The discrete-event framework shown in this paper provides a flexible model for representing autonomous mobility-on-demand, electric vehicle charging, and dynamic trip assignment. The forecasting models are evaluated exclusively as inputs to a downstream decision-support layer. Testing control strategies such as proactive rebalancing or demand-responsive pricing within the simulator is left as future work; the present study does not provide empirical validation of those control strategies or of any other downstream operational decision, including dynamic control, dynamic routing, traffic coordination, and alternative transport designs.

In the future, we will expand the set of events to include intersection conflicts or multiple lanes. We will also use domain knowledge through graph neural networks. Another area of focus will be few-shot adaptation. Here, we have a small amount of data for a target city for lightweight fine-tuning. As we move into an era of autonomous, electric, and on-demand urban transport, we expect AI-driven simulation frameworks to become crucial instruments for developing effective, sustainable and resilient mobility systems.

References

- [1] C. Creß, Z. Bing, and A. C. Knoll. Intelligent transportation systems using roadside infrastructure: A literature survey. *IEEE Transactions on Intelligent Transportation Systems*, 25(7):6309–6327, 2023.
- [2] F. Y. Wang, Y. Lin, P. A. Ioannou, L. Vlacic, X. Liu, A. Eskandarian, et al. Transportation 5.0: The DAO to safe, secure, and sustainable intelligent transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 24(10):10262–10278, 2023.
- [3] M. Elassy, M. Al-Hattab, M. Takruri, and S. Badawi. Intelligent transportation systems for sustainable smart cities. *Transportation Engineering*, 16:100252, 2024.
- [4] M. Abolghasemian, S. Kaveh, and F. Ebrahimzadeh. Simulation-based optimization: A comprehensive review of concept, method and its application. *Research Annals of Industrial and Systems Engineering*, 2(3):196–208, 2025.
- [5] T. Gong, L. Zhu, F. R. Yu, and T. Tang. Edge intelligence in intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 24(9):8919–8944, 2023.
- [6] S. Rahmani, A. Baghbani, N. Bouguila, and Z. Patterson. Graph neural networks for intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 24(8):8846–8885, 2023.
- [7] D. Katkar. Enhancing wireless communications in industry for optimized on-demand transportation systems: A discrete event simulation approach. In *2025 International Conference in Advances in Power, Signal, and Information Technology (APSIT)*, pages 1–7, Bhubaneswar, India, 2025. doi: 10.1109/APSIT63993.2025.11086331.
- [8] S. L. Tekle, B. Bonaccorso, and M. Naim. Simulation-based optimization of water resource systems: A review of limitations and challenges. *Water Resources Management*, 39(2):579–602, 2025. doi: 10.1007/s11269-024-04015-3.
- [9] X. Liu, X. C. Liu, C. Xie, and X. Ma. Impacts of photovoltaic and energy storage system adoption on public transport: A simulation-based optimization approach. *Renewable and Sustainable Energy Reviews*, 181:113319, 2023.
- [10] D. Katkar. Data-driven control for building energy management using switched affine models. In *2025 IEEE 3rd Global Conference on Wireless Computing and Networking (GCWCN)*, pages 1–9, Lonawala, Maharashtra, India, 2025. doi: 10.1109/GCWCN66157.2025.11448469.
- [11] Y. S. Kamma. A computational framework for intelligent data-driven decision optimization in complex digital systems, January 2026.
- [12] E. Mondal. Intelligent virtual modeling for dynamic optimization in large-scale systems. *International Journal of Intelligent Systems and Data Science*, 1(3), June 2026. doi: 10.67231/57mjv812.
- [13] V. Dhanani. Ai-driven aerial image analysis for construction equipment monitoring. ResearchGate Preprint, December 2025.

- [14] K. Rajwar and K. Deep. Uncovering structural bias in population-based optimization algorithms: A theoretical and simulation-based analysis of the generalized signature test. *Expert Systems with Applications*, 240:122332, 2024.
- [15] T. Wortmann, J. Cichocka, and C. Waibel. Simulation-based optimization in architecture and building engineering—results from an international user survey in practice and research. *Energy and Buildings*, 259:111863, 2022.
- [16] A. R. Lingampally. Latency-constrained resource orchestration in network sliced wireless systems. In *Proc. 2026 Third International Conference on Networking and Communications (ICNWC)*, pages 1–6, Chennai, India, 2026. doi: 10.1109/ICNWC68145.2026.11518523.
- [17] A. Saxena. Revisiting the equity premium puzzle: Time series analysis and forecasting from 1990 to 2012. In *2025 3rd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (AIMLA)*, pages 1–5, Namakkal, India, 2025. doi: 10.1109/AIMLA63829.2025.11041424.
- [18] U. V. Noubissie. Nonprofit evolution: Leading innovation in social ventures, 2025.
- [19] J. Gupta. Decentralized IoT data trading with subscription model. In *Proc. 2025 IEEE 3rd Global Conference on Wireless Computing and Networking (GCWCN)*, pages 1–10, Lonawala, Maharashtra, India, 2025. doi: 10.1109/GCWCN66157.2025.11447488.
- [20] A. Bhuekar. Ai in supply chain analytics: Adaptive decisioning via fuzzy behavioral models, 2025.
- [21] H. Raut. Scalable verification of distributed robotics systems using symbolic execution and port assumptions. In *2025 11th International Conference on Electrical Energy Systems (ICEES)*, pages 474–480, Chennai, India, 2025. doi: 10.1109/ICEES67011.2025.11213481.
- [22] P. Sukanya. Ai-optimized mechanical design of EV suspension systems for superior performance and durability. TechRxiv, November 2025.
- [23] A. R. Lingampally. Algorithmic simulation framework for multi-agent resource optimization in distributed computational systems. In *2026 2nd International Conference on Emerging and Intelligent Technologies and Systems (EITS)*, pages 1–6, Valencia, Spain, 2026. doi: 10.1109/EITS70066.2026.11609622.
- [24] S. Hanumantrao Shivankar. Adaptive foldable mechanism for bio-inspired bipedal gait enhancement. In *Proc. 2025 2nd International Conference on Industrial Automation and Robotics (IAR '25)*, pages 202–208, New York, NY, USA, 2025. Association for Computing Machinery. doi: 10.1145/3778886.3778918.