Rethinking Multi-Modal Tokenization for Reinforcement Learning with Transformers in Mobility-on-Demand Tasks

Gabriel Schwartz
Institute of Mathematics and Statistics
University of São Paulo
São Paulo, Brazil
gschwartz@ime.usp.br

Raphael Y. de Camargo
Federal University of ABC
Santo Andre, Brazil
raphael.camargo@ufabc.edu.br

Abstract

Deep Reinforcement Learning (Deep RL) has been applied to Autonomous Mobility on Demand (AMoD) systems to optimize vehicle assignment, fleet distribution, and passenger wait times. Recently, transformer-based models, such as the Decision Transformer and Trajectory Transformer, have emerged as promising alternatives by framing policy learning as sequence modeling, improving sample efficiency and capturing long-range dependencies. These models, however, face challenges when handling the high-dimensional, multi-modal state spaces typical of AMoD environments. In this work, we propose researching new tokenization and embedding techniques that extend transformer architectures for these complex tasks by expanding input modalities and improving token representations. Our approach will initially be evaluated on the vehicle assignment task, with the goal of future use in implementing micro-transit solutions to address the last-mile problem in urban transportation.

1 Preliminary Research Introduction

Deep Reinforcement Learning (Deep RL) models have become increasingly prominent in tackling complex control problems in Autonomous Mobility on Demand (AMoD) systems [1]. By integrating deep neural networks with reinforcement learning algorithms, these models can learn sophisticated policies that map high-dimensional inputs—such as real-time vehicle positions, passenger requests, and traffic conditions—to optimal actions [2, 3, 4, 5]. Some approaches use a single agent to decide for all vehicles [6] or model each vehicle using a separate agent [7, 5]. In MoD contexts, Deep RL agents aim to optimize long-term objectives like minimizing passenger waiting times, balancing fleet distribution, and maximizing overall system efficiency. Also, these models scale well for larger fleets since they can make decisions via inference in DNNs instead of solving ILPs, which are NP problems. They can also integrate vehicle assignment with strategic repositioning in a unified framework, allowing both to be balanced and optimized collectively. Finally, they can learn from experience, enabling their training using simulations and historical data.

Deep learning in general has, however, been the subject of a foundational shift in the past few years with the advent of data-driven large language models (LLMs) for natural language tasks, the main drivers of which are the introduction and development of the transformer architecture in Vaswani [8] and further works and the rise in availability of massive amounts of both data and computing power. These advances have naturally spread beyond natural language processing tasks, with fields like computer vision already heavily adopting these transformer-based models as state of the art and many other fields, including reinforcement learning, exploring their possible contributions. The Decision Transformer, proposed by Chen et al. [9], uses return-to-go conditioning to predict optimal actions from a transformer decoder-only sequence model trained on a dataset of trajectories for offline RL. In a similar vein, the Trajectory Transformer proposed by Janner et al. [10] also attempts to solve offline RL tasks by treating them like sequence modeling tasks and utilizing beam search to plan out trajectories. More recently, works such as Wu et al. [11]'s Masked Trajectory Models have expanded upon the concept by adopting a more general training approach through masked autoencoding, a simple self-supervised learning task which resulted in a pre-trained model applicable to different tasks.

These transformer-based models differ fundamentally from traditional Deep RL approaches by treating policy learning as a supervised learning problem on offline datasets of trajectories rather than relying on online interaction with the environment. This approach can lead to improved sample efficiency since the models can be trained on existing datasets without the need for extensive simulation [12, 11, 13]. Furthermore, transformers' ability to model long-range dependencies and handle variable-length sequences makes them well-suited for capturing the complex temporal patterns inherent in MoD systems, as seen in recent advances with world model transformers like Dreamer v3 [14]. For instance, they can potentially better predict passenger demand and optimize vehicle routing over extended time horizons, potentially improving over Deep RL methods in terms of both performance and generalization to new settings. One major limitation with transformer-based trajectory models, however, is their quadratic scaling with context length, which limits the length of the trajectory that can be fed into the transformer. This is especially an issue in models such as the Trajectory Transformer, in which each dimension in the state and action spaces has to be individually discretized into a separate token for transformer input, although it has also been an impediment to other transformer models like Gato [15], which cite the context length as a limiting factor.

Modeling MoD systems as reinforcement learning (RL) problems leads to high-dimensional multi-modal state spaces. Even for relatively simple tasks like vehicle assignment, where a control system must either decline or assign a vehicle to a trip request, the state space can include numerous variables from a large array of sources: vehicles, requests, traffic, street networks and more. This complexity grows in real-world scenarios because unlike in toy enviroments or standarized isolated agents, MoD systems present varying state sizes over time, such as a higher volume of requests during peak hours or an arbitrary amount of vehicle data due to dynamic fleet size changes. Traditional RL models require excessive padding to handle these variations, making state encoding difficult and inefficient. Creating token embeddings for each dimension, as is done in Trajectory Transformers [10] quickly becomes impractical due to the limitations in context length. Models like the Decision Transformer [9] and Masked Trajectory Transformer [11] simplify this by using modality-specific tokens (for "state," "action," and "rewards"). However, these models rely on relatively simple encoders, which work for RL benchmarks like Atari [16, 17, 18, 19] or D4RL [20] which have well defined shapes and data types. The Atari environment, for example, has a high-dimensional state space due to its observations being images, but it lacks the multi-modality of having multiple types of data and thus the entire space can be directly encoded by a convolutional network. These simple encoders may not suit more complex environments like vehicle assignment, where it is impractical to simply feed the entire state through a linear layer.

Existing research on high-dimensional multi-modal state spaces in Deep RL typically focuses on standard inputs like images and text, using well-established encoders (e.g., ResNets [21] or pre-trained models). However, these approaches struggle with the arbitrary and variable data shapes and types in MoD scenarios. Transformers, known for learning complex representations through data-driven training and modeling long-range dependencies, face limitations here due to context length constraints. There's a trade-off: encoding more observations into fewer tokens allows longer trajectory modeling but limits the transformer's ability to develop its own and possibly more meaningful representations from raw data. Models like the Decision Transformer and Trajectory Transformer represent two extremes—one compresses the entire state into a single token, sacrificing data richness for longer trajectories, while the other encodes every state dimension, offering more detail but limiting sequence length. Gato [15] takes a slightly more balanced approach, using ResNets to encode images in the inputs on one side, but still encoding each individual continuous observation as its own token, for example. This approach, although more balanced, is focused on multi-task performance and training a large foundational model, and because of that we believe that there is room for optimization in the case of MoD, where most of the inputs are not modalities like images or text.

To deal with these MoD tasks, we aim to develop trajectory tokenization and embedding techniques for these high-dimensional multi-modal scenarios by studying the features required for a sequence modelling transformer to achieve high performance in these tasks. We consider an approach that expands on Masked Trajectory Modelling by increasing the number of modalities beyond state, action and reward and possibly subdividing each into task-specific submodalities based on data structure. We also look to study and compare the ways in which embeddings for different modalities can be combined into individual tokens without impairing representation in order to further increase the efficiency of transformer-based models. We aim to use these tools to implement a sequence-modelling approach to the vehicle assignment problem and compare its performance with existing, non-RL algorithms and with other benchmark Deep RL approaches. Finally, we aim to advance beyond the vehicle assignment task to implement these techniques in possible micro-transit solutions for the last-mile problem in public transportation.

2 Work developed so far

To implement the vehicle assignment task for MoD, we developed an easily adaptable toolkit for generating candidate trajectory datasets based on FleetPy [22] simulations. FleetPy is an open-source simulation framework designed for fleet management and MoD services. It provides tools to model, simulate, and optimize the operations of vehicle fleets, such as ride-hailing services, car-sharing systems, and autonomous vehicle fleets. The framework supports dynamic fleet control and decision-making through modular real-time optimization techniques, considering various factors like vehicle availability, passenger requests, and routing. We execute simulations using the Ride Pooling Batch Offer Simulation module, which means that customers continuously make requests in real time to a single operator which periodically batches them together and simultaneously decides which to accept and which vehicles to assign to which request. Being a ride pooling simulation, vehicles can transport multiple passengers at once. FleetPy includes built-in algorithms for vehicle assignment, including amongst others the algorithm proposed by Alonso-Mora et al. [23], which is the default option, and a simple batch heuristic insertion alternative. Finally, FleetPy also contains a repositioning module, which tells idle vehicles to move to a new location in order to be closer to probable future demands. There are also built-in algorithms for this, and although we aim to eventually include repositioning as part of our proposed transformer controller, for now we have excluded that task and are only collecting data relating to request vehicle assignment.

FleetPy simulations take as input a set of parameters including fleet composition, vehicle types and optimization parameters, as well as a demand file describing the customer requests to simulate. We are initially using the TLC Trip Record Data archive of yellow taxi trips in Manhattan as a source of trip request demand. Since TLC's records are anonymous and thus have NYC zones as origins and destinations instead of precise geolocations for each trip, we randomly sample starting coordinates within the origin zone and ending coordinates within the destination zone so that their distance to each other is close to the reported trip's length in order to generate realistic trips from the record data. We then transform the generated trips into the appropriate input demand file format for FleetPy. We use OpenStreetMaps [24] to download the graph representation of Manhattan's street network which we use as input to FleetPy's routing engine and for sampling nodes for generated trips start and end points. Each generated file consists of a single day's worth of trips, totalling an average of over 80000 trips per day when using the data from the month of October, 2023, which was our original testing subset.

For parameters, we use the default options in most cases with the notable exception of the fleet size, which we increased to 350 vehicles in order to properly simulate the large amount of trips in our dataset. This, however, highlighted a major issue with the default recommended vehicle assignment algorithm, as the computing time needed to solve its optimization problem scaled drastically with fleet size and number of trips, in our case not being able to run a single iteration after a few hours on an AMD EPYC 7453 CPU. Because of that, we used the simpler batch heuristic algorithm, with which we successfully simulated a day's worth of trips in under an hour. In each optimization step, which was every minute in simulation time by default, we execute the optimization algorithm for the N requests batched for this step and collect the data presented in Table 1. As a note, we believe that using just the fare as a reward is most likely insufficient for a system aiming to minimize customer waiting times and coverage, however we've deemed it an acceptable parameter for this initial stage of testing as we are still attempting to verify that the vehicle assignment problem can be modeled with Deep RL techniques in the first place. Later iterations will use more advanced reward functions for better optimization.

Table 1: Data collected at each simulation step

Data Type	Description
State	
Timestamp	Current simulation time.
Vehicle location heatmap	128x128 grid representing current vehicle locations across the service area.
List with N requests	Each request includes: - Request ID. - Origin and destination coordinates. - Maximum waiting time (in seconds). - Number of passengers. - Earliest possible pickup time and latest allowable dropoff time. - List of 20 closest viable vehicles for assignment, each with: - Vehicle ID. - Planned future stops. - Current status (e.g., idle, en route). - Position (current vehicle coordinates). - Time and distance to reach the request's origin.
Actions	N pairs of (request ID, vehicle ID), corresponding to the assignments made by the heuristic algorithm.
Rewards	
Fare	The fare received for each accepted request.
Penalty	A placeholder -1 for each declined request.

Acknowledgments and Disclosure of Funding

This study was financed, in part, by the São Paulo Research Foundation (FAPESP), Brasil. Process Number #2024/06006-5

References

[1] Zhiwei (Tony) Qin, Hongtu Zhu, and Jieping Ye. Reinforcement learning for ridesharing: An extended survey. *Transportation Research Part C: Emerging Technologies*, 144, November 2022. ISSN 0968090X. doi: 10.1016/j.trc.2022.103852. arXiv: 2105.01099 Publisher: Elsevier Ltd.

[2] Jian Wen, Jinhua Zhao, and Patrick Jaillet. Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. In *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 220–225. IEEE, 10 2017. ISBN 978-1-5386-1526-3. doi: 10.1109/ITSC.2017.8317908. URL `http://ieeexplore.ieee.org/document/8317908/`.

[3] Xiaocheng Tang, Zhiwei Qin, Fan Zhang, Zhaodong Wang, Zhe Xu, Yintai Ma, Hongtu Zhu, and Jieping Ye. A deep value-network based approach for multi-driver order dispatching. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1780–1790, 2019.

[4] John Holler, Risto Vuorio, Zhiwei Qin, Xiaocheng Tang, Yan Jiao, Tiancheng Jin, Satinder Singh, Chenxi Wang, and Jieping Ye. Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1090–1095, 2019. doi: 10.1109/ICDM.2019.00129.

[5] Ge Guo and Yangguang Xu. A Deep Reinforcement Learning Approach to Ride-Sharing Vehicle Dispatching in Autonomous Mobility-on-Demand Systems. *IEEE Intelligent Transportation Systems Magazine*, 14(1):128–140, 2022. ISSN 19411197. doi: 10.1109/MITS.2019.2962159. Publisher: Institute of Electrical and Electronics Engineers Inc.

[6] Yang Liu, Fanyou Wu, Cheng Lyu, Shen Li, Jieping Ye, and Xiaobo Qu. Deep dispatching: A deep reinforcement learning approach for vehicle dispatching on online ride-hailing platform. *Transportation Research Part E: Logistics and Transportation Review*, 161, May 2022. ISSN 13665545. doi: 10.1016/j.tre.2022.102694. Publisher: Elsevier Ltd.

[7] Ashutosh Singh, Abubakr O Al-Abbasi, and Vaneet Aggarwal. A distributed model-free algorithm for multi-hop ride-sharing using deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):8595–8605, 2021.

[8] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[9] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[10] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem, 2021. URL `https://arxiv.org/abs/2106.02039`.

[11] Philipp Wu, Arjun Majumdar, Kevin Stone, Yixin Lin, Igor Mordatch, Pieter Abbeel, and Aravind Rajeswaran. Masked trajectory models for prediction, representation, and control, 2023.

[12] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models, 2023. URL `https://arxiv.org/abs/2209.00588`.

[13] Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions, 2023. URL `https://arxiv.org/abs/2303.07109`.

[14] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL `https://arxiv.org/abs/2301.04104`.

[15] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent, 2022. URL `https://arxiv.org/abs/2205.06175`.

[16] Volodymyr Mnih. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[17] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, June 2013. ISSN 1076-9757. doi: 10.1613/jair.3912. URL `http://dx.doi.org/10.1613/jair.3912`.

[18] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[19] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models, 2022. URL `https://arxiv.org/abs/2010.02193`.

[20] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL `https://arxiv.org/abs/1512.03385`.

[22] Roman Engelhardt, Florian Dandl, Arslan Ali, Yunfei Zhang, Fabian Fehn, Fynn Wolf, and Klaus Bogenberger. Fleetpy: A modular open-source simulation tool for mobility on-demand services, 07 2022.

[23] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467, 2017.

[24] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.