Delivery Optimization with Decision Transformers

Dillon Wilson

Abstract

With the prevalence of companies like Amazon, Walmart, and other retailers that allow customers across the country to purchase from a seemingly endless catalog of items, a sophisticated approach to fulfill those orders is a necessity. Due to the advancement of computational power, the feasibility of planning a national logistic network that consists of millions of unique items (2.2 million in wishlists from a study conducted in 2017 (Li et al. 2017)) has become a reality and any company wishing to enter the market requires a modern approach if they wish to compete with the industry giants. This project will explore that concept of planning a much-reduced network that consists of customers that order a specific item, distribution centers that facilitate deliveries and shipment of items across the country, and an agent that is tasked with efficiently handling it all.

Introduction

This project will explore the concept of creating an agent capable of managing a large scale logistics network in a manner similar to that of Amazon or Walmart. In order to achieve this, multiple challenges will need to be overcome. First, an environment capable of scaling will need to be created and tested for the agent to perform it's experiments. This environment will need to be able to dynamically change the number of customers, the number of and the location of the distribution centers, as well as the number of unique items for each episode. Thankfully there will be a constant for the environment, that being the paths that the agent will be able to ship items on which will be modeled based on the existing US Interstate System. Agent Design will be the next large challenge, in particular, how rewards will be given to the agent and how the state-space will be handled. These topics will be discussed much further in the Methodology section. This project will also use a fairly modern implementation of Reinforcement Learning known as a Decision Transformer. This recent modification to the standard transformer model normally used for language models like GPT can be adapted to the problem of Reinforcement Learning while retaining the ability to handle sequential data. This topic will be expanded on greatly in the Background section. The final, and likely largest, challenge of this project will be that of scale. The first working version of the model will be greatly reduced from the final goals. Hopefully, by the end of this project the size of the model will have been scaled up to three times the initial numbers presented. To be more specific, roughly 150-200 distribution centers, 10,000 unique items, and 30,000 customer orders. This is a far cry from the scale that a company like Amazon operates at, operating roughly 138 million square feet of warehouse alone (Hahn, Kim, and Youn), but serves as a large enough size to conduct tests.

Background

A cornerstone of this project is the usage of a very new type of model in the field of Reinforcement Learning known as the Decision Transformer. This model is a modification of the same transformer model that has been made famous due to it's application in the field of Natural Language Processing. It can be adapted with minimal changes for the goal of training a Reinforcement Learning agent. As with language transformers, Decision Transformers treat their input data sequentially meaning the agent will consider previous actions based on an attention mechanism presented in research that first introduced the concept (Vaswani et al. 2023). At each time step, a tuple of reward, action, and state is fed into an embedding layer that creates a representation of that data. Next, a positional encoder marks at what time step that tuple was processed by the model. The embedding is then fed into the transformer model, producing an output that is taken into a decoding layer. At the end of this process, an action is predicted that should have the highest likelihood of the best possible future reward. An illustration of this is provided at the end of this section in Figure 1.

A secondary option for model design will be presented for the case that the Decision Transformer proves to be a poor fit for this project. If it is determined not to be feasible a more simple approach will be taken in terms of model implementation. For a Model-Free implementation a Q-learning model or one of it's derivatives would make implementation far simpler, but forgo the ability of the agent to handle actions in a sequential matter. Alternatively, there is a great deal of research on the topic of LSTM models for Reinforcement Learning. This approach would likely not be as powerful as a decision transformer, but some research has shown that the two architectures are comparable in performance (Siebenborn et al. 2022) so it could serve as a possible secondary option if need be.

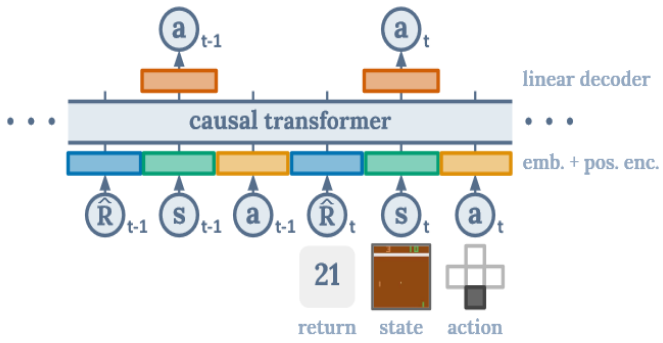

Figure 1: Decision Transformer (Chen et al. 2021)

Related Work

A crucial project that will be referenced constantly while working on this project is the research that introduced the concept of the Decision Transformer by adapting a language model to the problem of Reinforcement Learning (Chen et al. 2021). Luckily enough, there is even an experiment contained in that paper that is directly related to finding the shortest path in a graph. Due to the nature of the problem being essentially a graph where the edges are interstates and the vertices are distribution centers, research from 2023 around the application of Decision Transformers for graphs may prove useful. It is applied to a different type of problem, primarily Atari games, but will likely include useful details that will be applicable to this project (Hu et al. 2023). Another article that may prove very useful when attempting to formulate the problem correctly is known as the Meal Delivery Problem. It has it's differences, but retains the main elements of a source, a carrier that moves an item between locations, and a destination where the item needs to arrive in reasonable times. This article details an approach to solve this problem using deep Q-learning networks (Jahanshahi et al. 2022). Finally, in research done proposing delivery techniques that do not require human contact, an encoder-decoder based architecture is applied to the problem of Reinforcement Learning (Wu et al. 2023). This research may prove useful as a second source of implementation details, albeit using a different model architecture.

Environment

The first major step of the project is the creation of an environment that the agent will be able to use for it's simulations. As stated above, the 'map' is based on the real-world interstate system of the United States and distribution centers will be located along those roads to keep things simple. For the first major step in this project; there will be one distribution center per state, 1,000 unique items, and 10,000 customer orders. The customers will be randomly generated across the country as well as where those items are located in various distribution centers. Upon the start of an episode the agent will be allowed to perform two actions; move an item or items from one distribution center to another and perform a delivery from a distribution center. The agent will perform these actions until all customer orders have been fulfilled at which point the episode will end. Currently, there are two approaches that can be used to create this environment. The first and likely easiest approach will be using a pre-built environment in OpenAI's Gymnasism, a framework used to model various Reinforcement Learning problems (Brockman et al. 2016). This would also be convenient because the research presented in Chen et al. 2021 includes Python code made to work with OpenAI Gym. It is possible that an environment that fully captures the problem presented in this proposal does not exist, or does not quite meet the requirements needed. In that case, a custom-built environment will need to be created with Python.

Action-space As stated above, the design of the action-space for this project is extremely simple in terms of number of actions. There is some complexity in how the agent chooses to use those actions however. For example, if the agent chooses to move item, it needs to choose the type/types of items, the source distribution center, and the destination for that shipment. These decisions will be informed by the state-space which will encode the necessary data to help inform the agent. This can be accomplished by utilizing multiple models that take in the same state information and all make the separate decisions for an action. In essence, one model picks the action, a second model picks the source and destination pair, and a final model chooses which items should be moved. At some point in development, the agent will hopefully learn how to group orders that need to go in the same direction into singular actions which would great increase it's efficiency similar to the research presented in an article titled, 'The Order Batching Problem' (Cals 2019).

State-space The state-space is also fairly simple in design with some complexities in terms of size. The state must include current information on where customers are located and what items they desire, as well as the location of all distribution centers along with their current stock count. This introduces difficulty purely in the amount of data in the state at each time step. Each distribution center can have a random subset of the 1,000 total items with a random stock count. At the current stage, this is manageable but at the desired final state of 10,000 unique items the problem increases by an order of magnitude. One potential approach to solve this problem is to simplify what the agent is attempting to solve at each step. Rather than presenting every order to the agent at once, customers can be filtered down in a way to reduce what the agent needs to consider. For example, at a given time step the agent may be attempting to solve one order. To account for this the state-space could be drastically altered. Instead of representing the inventory of all of the distribution centers across the country, it could simply have a binary value indicating whether or not an individual distribution center would be able to fulfill the order the agent is attempting to fulfill. This a rather simplistic approach and may prove detrimental to the efficient behaviors that are desired in the agent, but could serve as a good starting point. After that system is working, the agent could increase it's capacity to consider every order in a particular state or potentially all orders of a

particular item.

Reward There are multiple ideas that will be tested for the reward for the agent. Likely, the simplest approach would be to use discounted rewards to encourage the agent to efficiently deliver items. However, this approach does not account for the fact that not all deliveries should be the same length. It would not be fair to penalize an agent for taking a higher number of actions to deliver an item if it is shortest path for that delivery. In order to prevent the agent from performing useless actions moving items around aimlessly, small negative rewards could be used whenever and item is moved. However this could backfire and lead to an agent that does not want to move items because it reduces it's reward. A more complex approach may be to use a scaling positive reward based on the number of steps taken to deliver an item. So for each extra step the agent takes over the minimum steps needed it gets penalized increasingly. This does introduce some complications as the shortest paths between all nodes in the graph will need to be calculated either beforehand or during runtime. Both approaches have their own drawbacks and will be experimented with during development. The agent will also be encouraged to explore early by giving small rewards when the agent discovers a new path for the first time.

Evaluation

The agent will be evaluated with a similar approach to how a company might do so. In terms of efficiency, an agent may be scored in one of two ways, time-efficiency or cost-efficiency. A time-based approach for evaluating the agent may be something as simple as the number of actions required to fill the order since starting on that order. . Alternatively, because the simulation uses real-world interstates, the real-world mileage that the item travelled can be calculated. This would enable the second approach, cost-efficiency, which can be described simply as how much "money" was spent to deliver an item (calculated as miles travelled * cost of gas per mile) to stand in as a pseudo-realistic estimation of how well agent is doing.

Timeline

This sections outlines the major tasks of this project along with estimations of when those milestones should be reached.

Timeline	Estimated Completion
Completion of environment.	2-27-24
Finished Agent Design	3-5-24
First Working Prototype	3-12-24
Midterm Presentation	3-18-24

Conclusion

This paper outlines a project proposal centered around the idea of creating an efficient agent capable of managing a reduced logistic network in the style of a national corporation like Amazon. This project will attempt to solve this problem using an advanced type of Reinforcement Learning model known as a Decision Transformer that uses the agents previous reward, state, and action to predict what the next action should be. This agent will be trained to fulfill customer orders in both a cost-efficient and time-efficient manner. Ideally, this agent will be able to exhibit a healthy balance of these two based on cost of deliveries in dollars, as well as the number of steps needed to fulfill an order.

References

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. arXiv:1606.01540.

Cals, B. 2019. The Order Batching Problem: A deep reinforcement learning approach.

Chen, L.; Lu, K.; Rajeswaran, A.; Lee, K.; Grover, A.; Laskin, M.; Abbeel, P.; Srinivas, A.; and Mordatch, I. 2021. Decision Transformer: Reinforcement Learning via Sequence Modeling. arXiv:2106.01345.

Hahn, Y.-N.; Kim, D.-H.; and Youn, M.-K. ???? A Brief Analysis of Amazon and Distribution Strategy. *Journal of Distribution Science*, 16(4): 17–20.

Hu, S.; Shen, L.; Zhang, Y.; and Tao, D. 2023. Graph Decision Transformer. arXiv:2303.03747.

Jahanshahi, H.; Bozanta, A.; Cevik, M.; Kavuk, E. M.; Tosun, A.; Sonuc, S. B.; Kosucu, B.; and Başar, A. 2022. A deep reinforcement learning approach for the meal delivery problem. *Knowledge-Based Systems*, 243: 108489.

Li, Y.; Zheng, N.; Wang, H.; Sun, K.; and Fang, H. 2017. A measurement study on Amazon wishlist and its privacy exposure. In *2017 IEEE International Conference on Communications (ICC)*, 1–7.

Siebenborn, M.; Belousov, B.; Huang, J.; and Peters, J. 2022. How Crucial is Transformer in Decision Transformer? arXiv:2211.14655.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2023. Attention Is All You Need. arXiv:1706.03762.

Wu, G.; Fan, M.; Shi, J.; and Feng, Y. 2023. Reinforcement Learning Based Truck-and-Drone Coordinated Delivery. *IEEE Transactions on Artificial Intelligence*, 4(4): 754–763.