Multi-Objective GPS Optimization

CS4991 Capstone Report, 2023

Pawan Jayakumar
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
Pj8wfq@virginia.edu

ABSTRACT
Modern mobile electronic devices such as smartphones or drones come with many different sensors to augment their capabilities. Since these devices have limited energy, engineering sensors to use as little energy as possible is an important goal. I addressed the challenge of optimizing GPS sensor usage in smartphones through the application of multi-objective reinforcement learning (MORL). I first simulated the performance of my algorithm in an ideal simulation using python and gradually introduced complexity. This technique reduced the number of required readings by 7-16%. Further empirical testing and evaluation on real data are ongoing to validate these findings and uncover potential challenges.

1. INTRODUCTION

In the age of the internet, mobile devices are everywhere. Consider that over 1.3 billion smartphones were sold every year since 2015 (Laricchia, 2023). Over time, smartphone technology has evolved to enable them for more uses. Taxi cab drivers in Manhattan prided themselves on having memorized all the streets and shops in the city, but now anyone can navigate the dozens of twists and turns with just a smartphone connected to the internet. This is because these phones come with an array of GPS sensors.

As useful as phones have become, they are limited by their battery life. GPS sensors, in particular, are notorious for draining phone batteries. One study showed that GPS sensors alone consume upwards of 50% of the energy a phone uses for all of its sensors (Khan, et al., 2016). These findings motivate the need to use them as sparingly as possible. A phone without energy amounts to nothing but a paperweight.

2. BACKGROUND

Reinforcement learning (RL) involves agents making sequential decisions in an environment to maximize cumulative rewards. In the case of MORL, the reward is a vector with each component representing the reward for a particular objective.

The simplest method to solve multi-objective problems with RL is to create a scalarization function, which converts the reward vector into a scalar by taking the dot product with a weight vector. This allows the designer to specify the relative importance of different objectives. At this point, a plethora of standard RL techniques can be used.

While scalarization can work, it is not very adaptable to changing conditions. For example, when a user wants to prioritize battery life, the GPS sensor should reduce the quality or frequency of readings.

3. RELATED WORKS

Several past experiments and algorithms have been researched to improve sensor usage efficiency. Yan, et al. (2012) approached this problem by dividing sensor readings into correlated segments. A group of algorithms

were then applied to optimally sample a subset of the sensor readings. This approach does not work in real time, as the segments are calculated post-sensing and so it does not actually save any energy.

Moamen, et al. (2015) created a service which acts as an intermediary between the application and the sensor. This service determines the minimum sensing rate that would satisfy the application and has filters to send out only the needed data to each application. While finding the minimum rate can save energy, there are many situations where this rate should be increased or decreased to provide better coverage/efficiency.

Cai, et al. (2019) successfully applied reinforcement learning to accelerometers. They model a group of sensor readings as the state in a Markov chain. This state is passed to the agent which then decides whether to turn the sensor on or off for the next few time steps. The agent is then updated using Q-learning. Importantly, they did not model sensor efficiency as a multi-objective problem. This means their algorithm is unable to adapt to changing user preferences. For example, sometimes the user wants to conserve their battery as much as possible, and other times requires high precision GPS readings and does not care about how much power is consumed.

4. PROJECT DESIGN

This project design simulates a real-world scenario where there is a user who moves about in their day but spends a significant amount of time stationary in a few locations such as the home, work, or gym. They have a navigation application on their phone which takes the movement trajectory of the user to make sure they are on the right path towards their destination. The goal of this project is to enable this application to use an optimal number of GPS readings based on the budget of energy it is given.

4.1 Multi-Objective Problem Formulation

Formally, a trajectory is a list of coordinates of the user over a period of time. The trajectory coverage is defined as the proportion of the coordinates that were sensed by activating the GPS. Sensor efficiency is defined as the number of times the GPS sensor was used over the time period of the trajectory. Increasing coverage comes at the cost of decreasing sensor efficiency.

4.2 Environment Modeling

A 5x5 grid world was set up, where the middle 9 squares are blocked off, effectively creating a circle. I modeled the end users of our GPS application as an agent which moved clockwise around the grid. The corners of the grid represented points of interest, and the agent stopped at each of these points for some amount of time. I first started with a fixed amount of time. Later I increased the difficulty by having the user stop for an amount of time sampled from a geometric distribution. One episode is defined as a full circle from the top left corner of the grid.

4.3 Agent Modeling

The agent was in charge of controlling whether or not to activate the GPS sensor. It received state in the form of a trajectory of the user in the past five time-steps. The trajectory is a list of the user's (x,y) position if it was sensed at that timestep and -1,-1 otherwise. To capture trajectory coverage, I took the ratio of the number of unique states sensed by the GPS to the number of unique states visited. To capture sensor efficiency, I applied a penalty anytime the agent activated the GPS. This created a two-dimension vector reward with corresponding weights $\boldsymbol{w_{cov}, w_{eff}}$. These weights can be dynamically changed to prioritize coverage or efficiency.

I started by fixing the weights to $\boldsymbol{w_{cov} = 1, w_{eff} = 0.3}$ It should be noted that the coverage reward component is non-Markovian

as the reward at each time step depends on the agent's previous actions. I then applied a scalarization function and then used both tabular q-learning and deep reinforcement learning with double Q- learning (van Hasselt, et al., 2015).

The agent used an epsilon greedy policy, choosing the action with the highest Q-score with probability 1-ε and a random action otherwise. The value of ε changed from 1 to 0.1 over the course of the model's train steps following an exponential decay.

4.4 Network Architecture and Training

For double Q-learning, a prioritized experience replay buffer was used (Schaul, et al., 2016). During each training step, a tuple consisting of the state, action, reward was placed into this buffer. Then, 32 of these tuples were sampled from the buffer and used to update the neural network.

The neural network architecture has three layers: an input, a hidden, and an output layer. The hidden layer has 64 neurons. Each layer is preceded by batch normalization and a RELU activation function. The network was trained using the standard gradient descent. The target neural network is synchronized with the policy network every 100 steps. The neural network is trained for 1000 episodes. Each episode varies in length based on how long the agent waited in the corners. After training, the agent's performance is tested on a different user trajectory.

5 RESULTS

The DQN agent was compared to a baseline tabular q-agent which was trained in a similar fashion except it used a table to calculate Q-values directly instead of approximating them with a neural network.

I trained both algorithms for the same number of episodes and under two different w_{eff} values: 0 and 0.3. In the first case, the agent should ideally learn to always turn on the sensor. In the second case, the agent should ideally learn to only turn on the sensor when the user is moving.

Figure 1 shows the model's performance over the duration of its training period. It shows that the DQN is able to learn how to perform better much quicker than the baseline. The base line method may need many more training steps to improve.

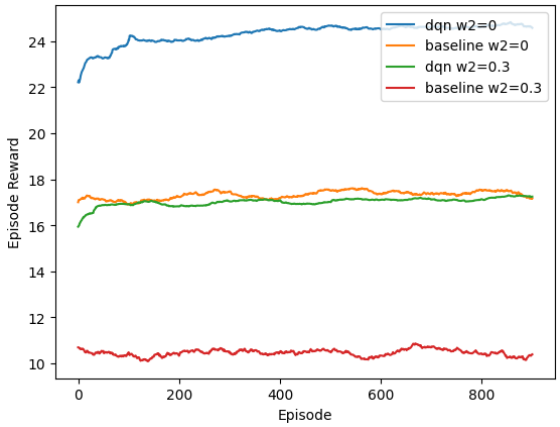

Figure 1: Episode Reward during Training

As shown in the results in Table 1, the DQN agent vastly outperformed the baseline in terms of test reward. Unlike the DQN agent, the baseline agent did not end up learning when the user was moving versus when the user was still. Given that the user is mostly moving around and rarely stays in the same position, a high sensor usage is expected, so an 8% reduction is reasonable.

Agent	w_{eff}	Test Reward	Sensor Usage
Baseline	0	15.25	0.5
Baseline	0.3	10.26	0.28
DQN	0	23.88	1.00
DQN	0.3	17.30	0.92

Table 1: Test Results

6 CONCLUSION

In summary, my paper introduces a multi-objective reinforcement learning (MORL) approach to optimize sensor usage in smartphone GPS systems. By addressing the inherent trade-off between energy efficiency and trajectory coverage, our simulation-based findings reveal a notable 7-16% reduction in the number of required readings.

My comprehensive project design incorporates environment and agent modeling, utilizing deep reinforcement learning techniques, particularly a DQN agent, which outperforms a baseline tabular Q-agent in terms of test reward.

7 FUTURE WORK

While my work set up a proof of concept in a simplified scenario, ongoing empirical testing and validation are crucial for refining my algorithm. Future work should explore its robustness across diverse user scenarios and real-world datasets, contributing to more sustainable and user-centric smartphone GPS applications.

8 ACKNOWLEDGMENTS

I want to acknowledge Dr Shangtong Zhang for creating this project idea and mentoring me through it.

REFERENCES

Cai, L., Boukhechba, M., Kaur, N., Wu, C., Barnes, L. E., & Gerber, M. S. (2019). Adaptive passive mobile sensing using reinforcement learning. *2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. https://doi.org/10.1109/wowmom.2019.8792967

Laricchia, F. (2023, July 21). *Smartphone sales worldwide 2007-2022*. Statista. https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/

Khan, I., Khusro, S., Ali, S., & Ahmad, J. (2016, December 1). *Sensors are power hungry: An investigation of smartphone sensors impact* ... Research Gate. https://www.researchgate.net/publication/317494054_Sensors_are_Power_Hungry_An_Investigation_of_Smartphone_Sensors_Impact_on_Battery_Power_from_Lifelogging_Perspective

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2016, February 25). *Prioritized experience replay*. arXiv.org. https://arxiv.org/abs/1511.05952

van Hasselt, H., Guez, A., & Silver, D. (2015, December 8). *Deep reinforcement learning with double Q-learning*. arXiv.org. https://arxiv.org/abs/1509.06461

Moamen, A. A., & Jamali, N. (2015). Share sens: An approach to optimizing energy consumption of continuous mobile sensing workloads. *2015 IEEE International Conference on Mobile Services*. https://doi.org/10.1109/mobserv.2015.22

Yan, Z., Eberle, J., & Aberer, K. (2012). Optimos: Optimal Sensing for mobile sensors. *2012 IEEE 13th International Conference on Mobile Data Management*. https://doi.org/10.1109/mdm.2012.43