Mego Logistics Project Documentation

By Adnan Rasool

Efficient logistics and route planning are crucial for reducing transportation costs and ensuring timely deliveries. In this blog, we explore how deep reinforcement learning (DRL) can be leveraged to solve the **Traveling Salesman Problem (TSP)** for optimizing delivery routes. Using **Proximal Policy Optimization (PPO)**, we train an intelligent agent to find the most efficient path across multiple destinations in India, minimizing travel distance while ensuring all locations are visited exactly once.

What is Deep Reinforcement Learning (DRL) and PPO?

Deep Reinforcement Learning (DRL) is a subset of machine learning where an agent learns to make decisions by interacting with an environment. The agent receives rewards based on its actions and optimizes its behavior to maximize long-term rewards. DRL is particularly useful for sequential decision-making problems like logistics and route planning.

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm that improves policy optimization stability by restricting large updates in policy gradients. PPO is widely used in complex decision-making tasks due to its balance of performance and ease of implementation.

Understanding the Problem

The **Traveling Salesman Problem (TSP)** is a classic optimization challenge where a salesman must visit a set of cities exactly once and return to the starting point while minimizing travel distance. This problem becomes more complex with increasing locations, making traditional brute-force methods computationally infeasible.

Our objective is to:

- Identify the shortest and most efficient route among **36 key locations in India**.
- Ensure all cities are visited once.
- Return to the starting point (**Mumbai Dock**) after completing the journey.
- Optimize using **Deep Reinforcement Learning (DRL)** with **PPO**.

Building the Distance Matrix

To begin, we construct a **distance matrix** using the **Haversine formula**, which calculates the great-circle distance between latitude and longitude coordinates:

```
import numpy as np
import pandas as pd

def haversine(lat1, lon1, lat2, lon2):
    R = 6371  # Earth's radius in km
    lat1, lon1, lat2, lon2 = map(np.radians, [lat1, lon1, lat2, lon2])
```

```
    dlat, dlon = lat2 - lat1, lon2 - lon1
    a = np.sin(dlat / 2) ** 2 + np.cos(lat1) * np.cos(lat2) * np.sin(dlon / 2) ** 2
    return 2 * R * np.arctan2(np.sqrt(a), np.sqrt(1 - a))

# List of locations with lat-long coordinates
locations = [
    ("Mumbai Dock", 18.9696, 72.8181),
    ("Delhi", 28.7041, 77.1025),
    # ... (remaining locations)
]

# Compute distance matrix
distance_matrix = np.zeros((len(locations), len(locations)))
for i in range(len(locations)):
    for j in range(len(locations)):
        if i != j:
            distance_matrix[i][j] = haversine(locations[i][1], locations[i][2], locations

# Convert to DataFrame
distance_df = pd.DataFrame(distance_matrix, index=[loc[0] for loc in locations], columns=
distance_df.to_csv("distance_matrix.csv")  # Save for training
```

This matrix serves as the foundation for our **reinforcement learning agent**.

Building the PPO-based TSP Solver

We implement a **Deep Reinforcement Learning (DRL) model** using PPO to train an agent that learns the optimal travel route. The core components include:

1. TSP Environment

The environment simulates the TSP by defining actions (moving between cities), states (current position, visited cities), and rewards (minimizing distance and penalizing revisits).

```
import torch
import torch.nn as nn
import torch.optim as optim

class TSPEnvironment:
    def __init__(self, distance_matrix, city_names):
        self.distances = distance_matrix
        self.city_names = city_names
        self.num_cities = len(city_names)
        self.start_city = 0  # Mumbai Dock as start point
        self.reset()

    def reset(self):
        self.current_city = self.start_city
        self.visited = [False] * self.num_cities
        self.visited[self.current_city] = True
        self.tour = [self.current_city]
        self.total_distance = 0
        return self._get_state()
```

Results and Best Route

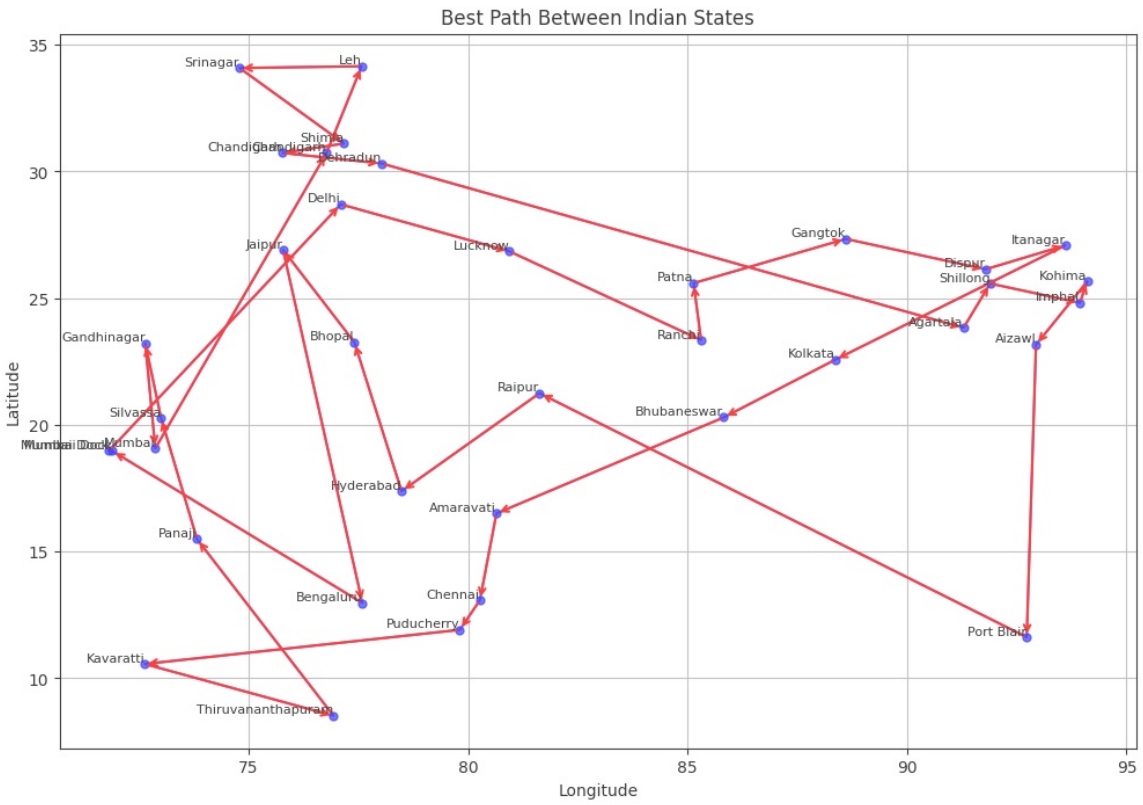
After training, our **optimal route** covered all locations efficiently, minimizing total travel distance to **21,008.55 km**:

Mumbai Dock -> Delhi -> Lucknow, Uttar Pradesh -> Ranchi, Jharkhand -> Patna, Bihar -> Gangtok, Sikkim -> Dispur, Assam -> Itanagar, Arunachal Pradesh -> Kolkata, West Bengal -> Bhubaneswar, Odisha -> Amaravati, Andhra Pradesh -> Chennai, Tamil Nadu -> Puducherry -> Kavaratti, Lakshadweep -> Thiruvananthapuram, Kerala -> Panaji, Goa -> Silvassa, Dadra and Nagar Haveli -> Gandhinagar, Gujarat -> Mumbai, Maharashtra -> Chandigarh, Haryana -> Leh, Ladakh -> Srinagar, Jammu and Kashmir -> Shimla, Himachal Pradesh -> Chandigarh, Punjab -> Dehradun, Uttarakhand -> Agartala, Tripura -> Shillong, Meghalaya -> Imphal, Manipur -> Kohima, Nagaland -> Aizawl, Mizoram -> Port Blair, Andaman and Nicobar Islands -> Raipur, Chhattisgarh -> Hyderabad, Telangana -> Bhopal, Madhya Pradesh -> Jaipur, Rajasthan -> Bengaluru, Karnataka -> Mumbai Dock

```
Total Distance: 21,008.55 km
```

Visualizing the Optimized Delivery Route

```
# Code for plotting the route
import matplotlib.pyplot as plt
plt.plot(longitudes, latitudes, marker='o', linestyle='-', color='b')
plt.title("Optimized Delivery Route")
plt.show()
```



Shipment Allocation and Loading Order

```
# Create a list of dictionaries for DataFrame creation
```

```
shipment_data = []
for location in best_route:
    product = shipment_allocation.get(location, "No specific product assigned")
    shipment_data.append({"Location": location, "Product": product})

# Create the pandas DataFrame
shipment_df = pd.DataFrame(shipment_data)
shipment_df
```

Location	Product
Mumbai Dock	Central warehouse and dispatch hub
Delhi	Modern urban fashion
Lucknow, Uttar Pradesh	Chikankari embroidery wear
Ranchi, Jharkhand	Everyday casual clothing
Patna, Bihar	Formal office wear
Gangtok, Sikkim	Hiking and trekking gear
Dispur, Assam	Lightweight cotton clothing
Itanagar, Arunachal Pradesh	Winter essentials
Kolkata, West Bengal	Festive sarees
Bhubaneswar, Odisha	Ethnic and casual wear
Amaravati, Andhra Pradesh	Traditional sarees and ethnic wear
Chennai, Tamil Nadu	Light cotton wear
Puducherry	Relaxed summer clothing
Kavaratti, Lakshadweep	Light cotton wear
Thiruvananthapuram, Kerala	Cotton and linen wear
Panaji, Goa	Summer and beachwear
Silvassa, Dadra and Nagar Haveli	No specific product assigned
Gandhinagar, Gujarat	Festive wear
Mumbai, Maharashtra	No specific product assigned
Chandigarh, Haryana	Activewear
Leh, Ladakh	Adventure gear
Srinagar, Jammu and Kashmir	Thermal wear and woolen clothing
Shimla, Himachal Pradesh	Heavy-duty winter gear
Chandigarh, Punjab	Thermal wear
Dehradun, Uttarakhand	Outdoor and trekking apparel
Agartala, Tripura	Handloom sarees
Shillong, Meghalaya	Rain-resistant apparel

Imphal, Manipur	Traditional handwoven garments
Kohima, Nagaland	Accessories like caps and scarves
Aizawl, Mizoram	Ceremonial and cultural attire
Port Blair, Andaman and Nicobar Islands	Light beachwear
Raipur, Chhattisgarh	Casual clothing
Hyderabad, Telangana	Modern fashion
Bhopal, Madhya Pradesh	Multi-purpose clothing
Jaipur, Rajasthan	Designer lehengas and sarees
Bengaluru, Karnataka	Trendy urban fashion
Mumbai Dock	Central warehouse and dispatch hub

```
# Generate loading order (reverse order of delivery)
loading_order = list(reversed(best_route[1:-1]))

# Display Loading Order
print("\nLoading Order (Last to be delivered is loaded first):\n")
for location in loading_order:
    product = shipment_allocation.get(location, "No specific product assigned")
    print(f" Load: {product} for {location}")
```

```
Shipment Allocation Plan:

Mumbai Dock:-> Central warehouse and dispatch hub
Delhi:-> Modern urban fashion
Lucknow, Uttar Pradesh:-> Chikankari embroidery wear
Ranchi, Jharkhand:-> Everyday casual clothing
Patna, Bihar:-> Formal office wear
Gangtok, Sikkim:-> Hiking and trekking gear
Dispur, Assam:-> Lightweight cotton clothing
Itanagar, Arunachal Pradesh:-> Winter essentials
Kolkata, West Bengal:-> Festive sarees
Bhubaneswar, Odisha:-> Ethnic and casual wear
Amaravati, Andhra Pradesh:-> Traditional sarees and ethnic wear
Chennai, Tamil Nadu:-> Light cotton wear
Puducherry:-> Relaxed summer clothing
Kavaratti, Lakshadweep:-> Light cotton wear
Thiruvananthapuram, Kerala:-> Cotton and linen wear
Panaji, Goa:-> Summer and beachwear
Silvassa, Dadra and Nagar Haveli:-> No specific product assigned
Gandhinagar, Gujarat:-> Festive wear
Mumbai, Maharashtra:-> No specific product assigned
Chandigarh, Haryana:-> Activewear
Leh, Ladakh:-> Adventure gear
Srinagar, Jammu and Kashmir:-> Thermal wear and woolen clothing
Shimla, Himachal Pradesh:-> Heavy-duty winter gear
Chandigarh, Punjab:-> Thermal wear
Dehradun, Uttarakhand:-> Outdoor and trekking apparel
Agartala, Tripura:-> Handloom sarees
Shillong, Meghalaya:-> Rain-resistant apparel
Imphal, Manipur:-> Traditional handwoven garments
Kohima, Nagaland:-> Accessories like caps and scarves
```

```
Aizawl, Mizoram:-> Ceremonial and cultural attire
Port Blair, Andaman and Nicobar Islands:-> Light beachwear
Raipur, Chhattisgarh:-> Casual clothing
Hyderabad, Telangana:-> Modern fashion
Bhopal, Madhya Pradesh:-> Multi-purpose clothing
Jaipur, Rajasthan:-> Designer lehengas and sarees
Bengaluru, Karnataka:-> Trendy urban fashion
Mumbai Dock:-> Central warehouse and dispatch hub

Truck will return to Mumbai Dock after completing all deliveries.

Loading Order (Last to be delivered is loaded first):

 Load: Trendy urban fashion for Bengaluru, Karnataka
 Load: Designer lehengas and sarees for Jaipur, Rajasthan
 Load: Multi-purpose clothing for Bhopal, Madhya Pradesh
 Load: Modern fashion for Hyderabad, Telangana
 Load: Casual clothing for Raipur, Chhattisgarh
 Load: Light beachwear for Port Blair, Andaman and Nicobar Islands
 Load: Ceremonial and cultural attire for Aizawl, Mizoram
 Load: Accessories like caps and scarves for Kohima, Nagaland
 Load: Traditional handwoven garments for Imphal, Manipur
 Load: Rain-resistant apparel for Shillong, Meghalaya
 Load: Handloom sarees for Agartala, Tripura
 Load: Outdoor and trekking apparel for Dehradun, Uttarakhand
 Load: Thermal wear for Chandigarh, Punjab
 Load: Heavy-duty winter gear for Shimla, Himachal Pradesh
 Load: Thermal wear and woolen clothing for Srinagar, Jammu and Kashmir
 Load: Adventure gear for Leh, Ladakh
 Load: Activewear for Chandigarh, Haryana
 Load: No specific product assigned for Mumbai, Maharashtra
 Load: Festive wear for Gandhinagar, Gujarat
 Load: No specific product assigned for Silvassa, Dadra and Nagar Haveli
 Load: Summer and beachwear for Panaji, Goa
 Load: Cotton and linen wear for Thiruvananthapuram, Kerala
 Load: Light cotton wear for Kavaratti, Lakshadweep
 Load: Relaxed summer clothing for Puducherry
 Load: Light cotton wear for Chennai, Tamil Nadu
 Load: Traditional sarees and ethnic wear for Amaravati, Andhra Pradesh
 Load: Ethnic and casual wear for Bhubaneswar, Odisha
 Load: Festive sarees for Kolkata, West Bengal
 Load: Winter essentials for Itanagar, Arunachal Pradesh
 Load: Lightweight cotton clothing for Dispur, Assam
 Load: Hiking and trekking gear for Gangtok, Sikkim
 Load: Formal office wear for Patna, Bihar
 Load: Everyday casual clothing for Ranchi, Jharkhand
 Load: Chikankari embroidery wear for Lucknow, Uttar Pradesh
 Load: Modern urban fashion for Delhi
```

By using reinforcement learning, we ensure **cost-effective, efficient, and optimized** travel for logistics and delivery operations. The PPO algorithm allows the model to learn from previous mistakes, adapt to new data, and generalize solutions efficiently. Future improvements may include dynamic adjustments for real-time traffic conditions and **multi-agent** solutions for fleet optimization.