Execution Guide for Ambulance Optimisation

The repository https://github.com/AGAMPANDEYY/Ambulance-Optimisation-EfkonStrabag.git for optimization of Ambulance (i) deployment on Potential Sources (P) has following modules:

1- **Pre-processing** -----> to match accident chainage to nearest equipment's Lat-Lon.
2- **Clustering** ----> to calculate Demand Points
3- **OD Time Matrix** ---> to calculate Ambulance to Demand Point shortest travel time of Ambulance
4- **PuLP Optimization Module** ----> Mathematical formulas with f to maximize coverage of each demand point with given Primary(r1) and Secondary(r2) response time and alpha reliability.

How to run the 4 modules on custom dataset?

1- Pre-processing (Depends on the dataset)

Use this module only when the dataset used does not have accident lat-lon and frequency.

If only accident chainage is used, this module maps the accident chainage to the nearest equipment chainage and assigns the corresponding Lat-Lon of the equipment to the accident chainage.

Input- Raw dataset with mandatory fields--->

- Accident_Chainage,
- Accident_Frequency,
- Equipment_Chaiange,
- Equipment_Lat,
- Equipment_Lon

Output---> .csv with Accident_Chainage, Accident_Lat-Lon

Code to run:

1- src/pre-processing/pre-processing.ipynb

2- Clustering (Both for Fatal & all accidents)

This module now clusters the fatal accidents and gives the centroid of each cluster as a demand point.

There are 3 Clustering Algorithms used:

1- KMeans ----> Use when the accidents are uniformly spaced LatLon
2- DBSCAN & OPTICS ---> Use when accidents are Non-Uniformly spaced (Density different)

Input: Processed dataset from the Pre-processing step with fields:

- Accident_Chainage
- Accident_Lat
- Accident_Lon

Output: .csv with Clustered Demand Points including fields:

- Cluster_ID
- Demand_Point_Lat
- Demand_Point_Lon
- Demand_Frequency

Code to run:

1- src/clustering/Clustering.ipynb (Choose KMeans/DBSCAN/OPTICS)

3. OD Time Matrix

This module calculates the shortest travel time from each ambulance location to each demand point.

Input:

- Ambulance locations with fields: Ambulance_ID, Ambulance_Lat, Ambulance_Lon
- Demand points with fields: Demand_Point_ID, Demand_Point_Lat, Demand_Point_Lon

Output: .csv and numpy array with OD Time Matrix with dimension DPxS (DP is demand point and S is source)

Code to run:

1- src/time_matrix/osrm-distance-matrix.ipynb (API based DM)
2- src/time_matrix/chainage_distance_matrix.ipynb (Chainage Distance Matrix)
3- src/time_matrix/time_matrix.ipynb (covert Distance Matrix to Time Matrix)

4. PuLP Optimization Module

This module uses mathematical optimization to maximize the coverage of each demand point with the given primary (r1) and secondary (r2) response times and alpha reliability.

Input:

OD Time Matrix from the previous step with fields:

- Ambulance_ID,
- Demand_Point_ID,
- Travel_Time.

Additional parameters:

1- Primary response time (r1),
2- Secondary response time (r2),
3- Alpha reliability,
4- P (Total number of ambulances to be deployed),
5- N (Total Demand Points)
6- m (Total number of Potential Sources)
7- p_j (Max. number of ambulances allowed at each location)

Output: Optimized ambulance deployment plan including fields:

- Source Lat Lon
- Binary (0,1) for ambulance deployment
- Number of ambulance deployed (1 in optimal cases for our model)

Code to run:

- src/optimization/optimisation_algo.ipynb
- src/optimization/optimisation_pipeline.ipynb (for loop iterations for multiple parameters and Visualisation)
- src/optimization/optimization-comparison.ipynb (for comparing old and optimised deployment at each potential Trauma centres)

This guide provides a step-by-step execution process for running each module on a custom dataset. Adjust the input and output paths according to your file locations and dataset specifications.