Fleet and traffic management systems
for conducting future cooperative mobility

D3.4 Final harmonization, fusion, optimization, detection and balancing approaches

Document Type	Deliverable
Document Number	D3.4
Primary Author(s)	Babis Magoutas | FRIC
Document Version / Status	1.0 | Final version
Distribution Level	CO (confidential – consortium only)
Project Acronym	CONDUCTOR
Project Title	Fleet and traffic management system for conducting cooperative mobility
Project Website	https://conductor-project.eu/
Project Coordinator	Netcompany Intrasoft SA | www.netcompany-intrasoft.com
Grant Agreement Number	101077049

CONDUCTOR project has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101077049.

Co-funded by the European Union

CONTRIBUTORS

Name	Organization	Name	Organization
Panagiotis Georgakis	FRONT	Raquel Sánchez	Nommon
Filippos Gkountoumas	FRONT	Pablo Ruiz	Nommon
Ioannis Alexandrakis	FRONT	Oliva García Cantú	Nommon
Babis Magoutas	FRIC	Zakir Farahmand	UTwente
Margarita Tsarmpopoulou	FRIC	Oskar Eikenbroek	UTwente
Serafeim Zorbas	FRIC	Anika Laschewski	UTwente
Arslan Ali Syed	TUM	Alejandro Tirachini	UTwente
Arka Ghosh	DEUSTO	Eric van Berkum	UTwente
Jenny Fajardo	DEUSTO	Matej Polzelnik	JSI
Leire Serrano	DEUSTO	Rok Hribar	JSI
Filip Stavrov	JSI	Luka Stopar	JSI

FORMAL REVIEWERS

Name	Organization	Date
Athina Tympakianaki	Aimsun	2025-04-23
Matina Lai-Ying Chau	NTUA	2025-04-25
Emmanouil Nisyrios	NTUA	2025-04-25

DOCUMENT HISTORY

Revision	Date	Author / Organization	Description
0.1	2024-12-18	FRIC	Draft Table of Contents
0.2	2025-04-11	FRIC	Inputs from all authors
0.3	2025-04-14	FRIC	Draft version sent for internal review
0.4	2025-04-25	FRIC / Aimsun / NTUA	Draft version following internal review
1.0	2025-05-01	FRIC / Netcompany	Final version for submission

LEGAL DISCLAIMER

The CONDUCTOR project is co-funded by the European Union's Horizon Europe research and innovation programme under the Grant Agreement No 101077049. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Deliverable D3.4 presents the final outcomes of the CONDUCTOR project's Work Package 3, which is dedicated to the harmonisation, fusion, optimisation, detection, and balancing approaches supporting next-generation traffic and fleet management systems. Building upon the foundations laid in deliverables D3.1, D3.2 and D3.3, this report consolidates the results of technical developments, validation activities, and cross-pilot integration, reflecting the maturation of several key components within the overall system architecture.

At the heart of this work is the development of modular, interoperable services designed to ingest, fuse, and analyse multimodal traffic and transport data in real time. These services incorporate advanced data-driven techniques such as machine learning, statistical modelling, and ensemble methods to enable responsive detection of deviations and support the dynamic optimisation of traffic flows and fleet operations. This deliverable presents progress beyond the state of the art in several areas, including the operational deployment of ensemble-based anomaly detection, integration of contextual scoring and recovery estimation models, and flexible fusion pipelines capable of adapting to heterogeneous data sources. These contributions collectively enhance the system's responsiveness to real-world traffic conditions and its ability to support coordinated decision-making across diverse mobility scenarios.

The results reported herein span all three CONDUCTOR use cases, with implementation and validation activities carried out in simulation and pilot contexts. These include multi-source data harmonisation for route-level optimisation, detection of anomalies in corridor traffic, estimation of recovery durations following disruptions, and demand balancing strategies applied in both passenger and logistics settings. Particular attention was given to ensuring technical readiness and alignment with the overarching system requirements defined in other work packages. The deliverable also reflects on key lessons learned throughout the development and deployment phases, including the challenges of cross-border data availability and the importance of integrating external factors such as weather and planned events. Where limitations were encountered—such as data gaps in specific parts of the corridor—preliminary mitigation strategies were explored, including model-based inference and proxy estimation.

Overall, D3.4 represents a significant step forward in the transition from experimental prototypes to operational services. The components presented are ready to be integrated and tested in the upcoming pilot demonstrations, forming a critical part of CONDUCTOR's vision for intelligent, flexible, and coordinated mobility management across European transport networks.

Keywords: Traffic Management, Fleet Optimization, Data Fusion, Anomaly Detection, Machine Learning, Real-Time Monitoring, CCAM, Pilot Integration, Recovery Estimation, Use Case Validation, System Readiness.

OBJECTIVES

Deliverable D3.4 represents the final technical output of Work Package 3 (WP3) within the CONDUCTOR project. It documents the complete implementation of the harmonization, fusion, optimization, detection, and balancing approaches developed over the course of the project. This deliverable builds upon and finalizes the preliminary versions presented in D3.1 (Specification and initial version of data gathering, harmonization, fusion and analysis techniques), D3.2 (Specification and initial version of optimization and balancing algorithms), and D3.3 (Intermediate implementation of WP3 components). It incorporates refinements and enhancements derived from technical feedback loops, particularly those stemming from testing and validation activities in WP5.

The work reported herein is directly linked to five core technical tasks under WP3:

- **Task 3.1 – Data Gathering and Harmonization**: Final integration of structured and unstructured datasets from diverse pilot contexts, addressing completeness, quality, and compatibility for downstream use.
- **Task 3.2 – Data Fusion and Analysis**: Implementation of fusion strategies combining multimodal inputs to enrich decision-making, including statistical pre-processing and data abstraction methods.
- **Task 3.3 – Network Load Balancing**: Realization of rule-based and data-driven balancing strategies for traffic and fleet distribution, leveraging insights from real-time and historical data flows.
- **Task 3.4 – Dynamic Optimization**: Finalization of optimization routines capable of adjusting routes, resources, and service levels dynamically in response to operational conditions.
- **Task 3.5 – Anomaly Detection**: Completion of detection mechanisms designed to identify and contextualize disruptions across traffic networks, including model calibration and performance evaluation.

In alignment with the overall objectives of WP3, this deliverable addresses the following goals as outlined in the Grant Agreement's Description of Action (DoA):

- **Objective 3.1**: Selection and implementation of techniques for data gathering, harmonization, fusion, and analysis.
- **Objective 3.2**: Development of dynamic optimization and network load-balancing methodologies.
- **Objective 3.3**: Integration of robust anomaly detection routines.

These objectives contribute directly to the overarching ambition of the CONDUCTOR project, which is to develop advanced traffic and fleet management systems capable of supporting Cooperative, Connected, and Automated Mobility (CCAM). In particular, WP3 serves as the data and algorithmic backbone of the CONDUCTOR architecture, enabling the project's decision support tools to operate effectively under both mixed and fully autonomous transport conditions. The deliverable also reinforces CONDUCTOR's commitment to scalability and adaptability, ensuring that the developed components are modular, interoperable, and validated across a variety of pilot settings. By consolidating all core WP3 functionalities into their final form, D3.4 sets the foundation for their integration into the large-scale demonstrations planned under WP6 and WP7.

1. DATA GATHERING & HARMONIZATION

1.1 Data Harmonization

In CONDUCTOR we are using data from various sources. Moreover, the project is being done by different teams from all over Europe using different systems, thus we are dealing with the data that varies in structure, format and meaning. To ensure a smoother handling and sharing of various data we adopted Data Harmonisation proces. This allows us to make the data more consistent, enables seamless integration of applications or components and enable efficient investigation of CCAM services.

In D3.1 we presented the selected data models as the basis for data harmonisation. We selected the FIWARE's smart data models based on suitability and alignment with the project's goals. The following smart data models were selected for harmonization:

- ItemFlowObserved: This data model can effectively capture readings obtained from infrastructure sensors such as loop detectors. These readings encompass various traffic engineering parameters, including average speed, flow, occupancy, and others.
- WeatherObserved: This data model represents observations of weather conditions at specific locations and times.

The following data sources were selected for the harmonization demonstration:

1. Attica Trafic Data
2. DARS Traffic Data
3. Weatherapi Weather Data
4. DARS Traffic Events Data

The harmonization mapping for first three data sources was described in D3.1. For the duration of the project only one additional data source was added and mapped, DARS Traffic Events Data

1.2 CONDUCTOR Dataspace

This section outlines the final specifications and implementation details of the Dataspace. It includes a short overview of the initial version (D3.1) and provides detailed information on the deployment and configuration of IDSA-compliant connectors, integration with a comprehensive dataset management system, and the extension of the Dataspace into a scalable and reliable big data processing platform.

In Deliverable D3.1, a Dataspace architecture was proposed as a comprehensive solution for managing the project's models and solutions. This architecture ensures data sovereignty, control, interoperability, and trustworthiness. To achieve this, tools from the Fiware ecosystem were leveraged, including a pub/sub framework. Additionally, International Data Spaces Association (IDSA) connectors were identified as a key component to facilitate secure and effective communication and data exchange within the Dataspace.

In this section we focus on the directions the conductor data exchange space has evolved, and more specifically:

- In section 1.2.1, we describe the implementation, deployment and configuration of Eclipse Dataspace Connectors, together with an example of data exchange between two dataspace participants.

- In section 1.2.2 we describe the integration of a comprehensive dataset management system to the dataspace.
- In section 1.2.3 we describe the extension of the dataspace for a containerized, scalable and resilient big data processing platform

1.2.1 Eclipse Data Space Connector (EDSC) deployment and configuration

In the CONDUCTOR dataspace, no single entity has control over all data, instead each stakeholder controls how they share their own data. Apart from the decentralized components, some centralised components like an Identity Service and a Catalog Service are deployed and can be connected to the Orion Message Broker. An overview of the updated Architecture can be found in Figure 1.

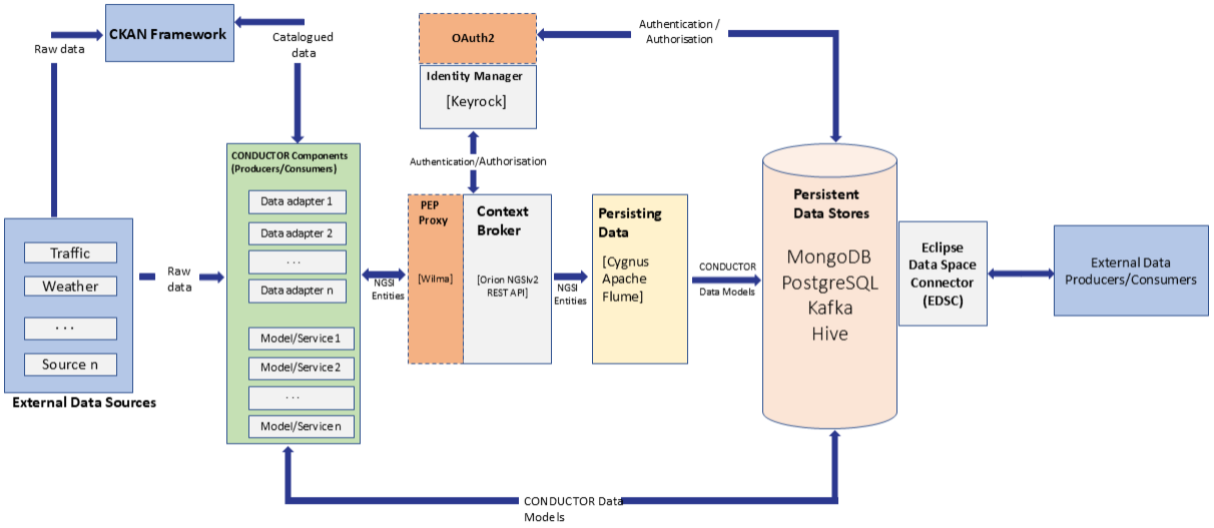

Figure 1: Updated Architecture of Dataspace

The implementation we follow is using the EDSC[1], adhering to the standards set by the IDSA and ensuring:

- **Interoperability**: for seamless integration across different stakeholders.
- **Security**: to protect sensitive data during exchanges.

The main components that we established in the dataspace are the following:

- **Control Plane**: responsible for assembling catalogues, launching contract agreements to provide grant access to data, managing data transfers, and monitoring usage compliance.
- **Data Plane**: responsible for transmitting data using a wire protocol following the requests by the control plane.
- **Identity Hub**: manages organization identity resources such as credentials for all dataspace participants.

[1] https://github.com/eclipse-edc

- **Federated Catalogue:** aggregated catalogues of multiple participants in a dataspace (crawler-based)

In general, the overall process for data exchange using the Eclipse Data Space Components is executed in two phases:

- Contract offering
- Contract negotiation

The steps that comprise the process, in the general scenario, are the following:

- **Define Asset Data**: Specify the dataset to be provided and set the data to be offered.
- **Create Asset**: Generate metadata related to the dataset, including the data address and additional information.
- **Create Policy**: Develop a Data Access Rule in ODRL format, outlining the terms under which the data is provided.
- **Configure Contract:** Link the Asset to the Policy to create a Contract Definition.
- **Query Contract Offers:** Data Consumer initiates a request to retrieve the offers catalogue from the provider.
- **Request Metadata:** The Consumer fetches additional metadata on a specific offer.
- **Accept Contract:** The Consumer accepts the contract and initiates negotiations.
- **Negotiate and Finalize Contract:** Both the Provider and Consumer agree to the contract terms.
- **Initiate Data Transfer:** The Consumer requests the start of the data transfer process.
- **Perform Data Transfer:** Execute the data transfer as agreed upon in the contract.

1.2.2 Dataset Management Integration

In the context of data space, a comprehensive data management system can help facilitate the availability and management of data essential for CONDUCTOR services. A Comprehensive Knowledge Archive Network (CKAN)-based system (e.g. Figure 2) can provide extensive data management capabilities for the collection, distribution, and utilisation of mobility data and other datasets, such as air pollution data by the different actors. CKAN is also compatible with data security, privacy, governance, and sovereignty requirements during the management of open, shared, and closed data types.

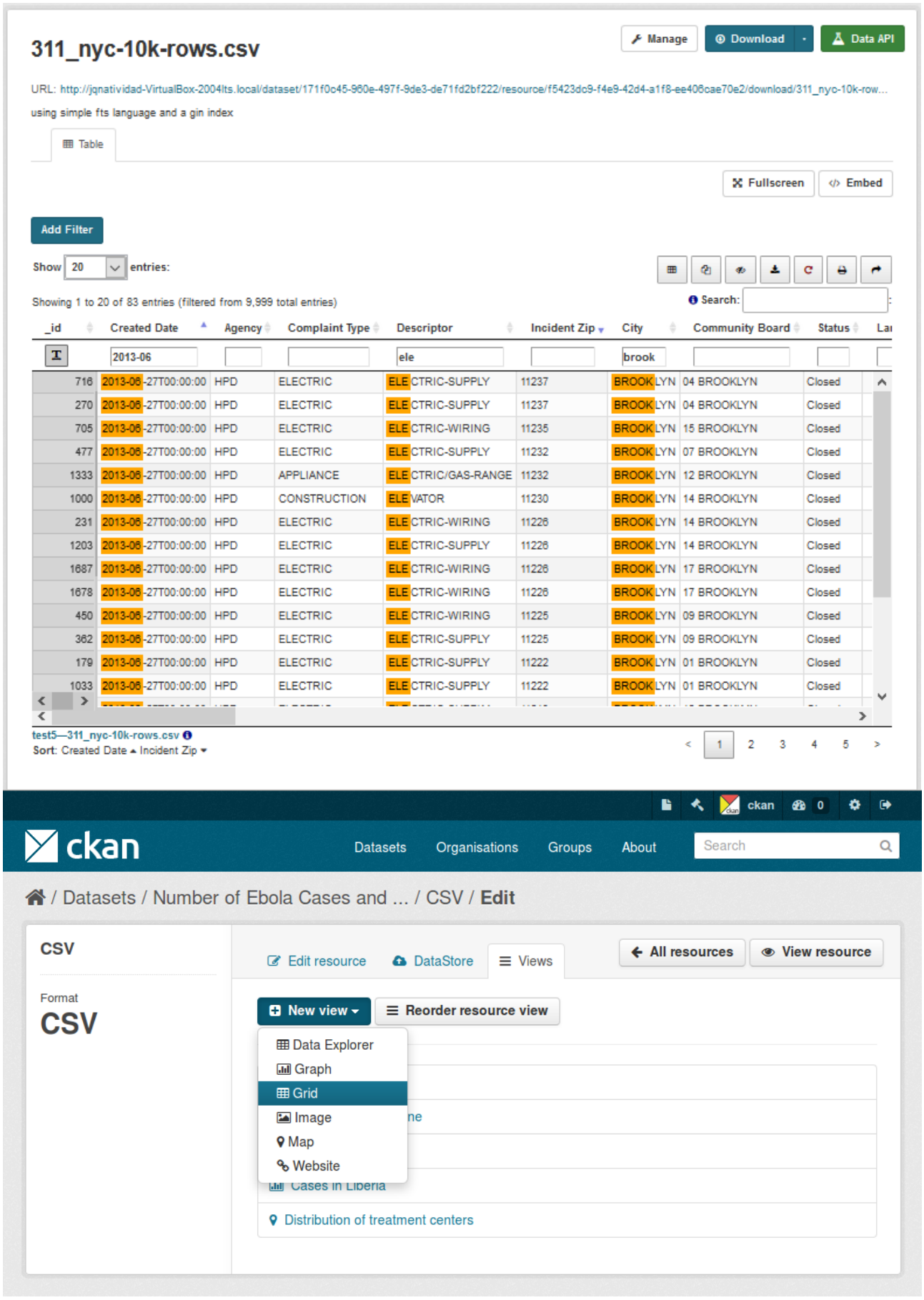

Figure 2: CKAN example GUI

Some features provided by CKAN can strongly support the Use Cases of CONDUCTOR regarding data management, as well as the integration to the dataspace components:

- **API**: CKAN provides full API access to all functionalities, enabling external applications and scripts to easily integrate. The API allows integration with dataspace components such as the connectors or the pub/sub system, while also allowing dataspace participants can write custom apps and automation on top of the data management platform.
- **Datastore:** A CKAN extension provides a database for storing structured data from files. When dataset files (e.g. CSV or Excel) are added to CKAN, their contents are extracted and loaded into a database, making the data query able in real time.
- **Metadata:** Each dataset can be thoroughly described for discovery and management, with custom metadata attributes that might include location coordinates, data quality rating, or others.
- **Search:** CKAN's search functionality enables users to quickly find the data they need, even in a large catalogue. Since all metadata is indexed, this provides a high-usability experience for users and, with API integration, allows external applications to leverage the search capabilities.
- **Visualization:** CKAN includes built-in visualization tools, for data validation and for exploration.
- **Federate:** CKAN can participate in federated networks of data portals, effectively allowing multiple catalogues to share and synchronize data. CKAN also supports the DCAT standard for data catalogue metadata, which means it can exchange information with other compatible systems.

1.2.3 Containerized Big Data architecture

For all data space components, including data processing, a container-based infrastructure is preferred. Containerized applications offer flexibility and efficient resource management, facilitating deployment and scaling for data exchange and processing within the project. Containers encapsulate applications and their dependencies, enabling seamless deployment across diverse environments.

For big data processing, Kubernetes was chosen for container orchestration, as detailed in deliverable D3.1 (Specification and initial version of data gathering, harmonisation, fusion and analysis techniques). The rest of the components have also been deployed using containers, but without a dedicated orchestration framework to avoid unnecessary overhead. Eclipse Dataspace components have been deployed and configured as Docker containers, allowing for the custom deployment of multiple services across separate and diverse server environments. Additionally, the CKAN-based data management platform has been deployed as a Docker container on a Virtual Private Server in the European Union.

2. DATA FUSION & ANALYSIS

This section describes the data fusion algorithms required for developing the CONDUCTOR decision support models and tools. These algorithms aim to transform harmonised data, collected from various sources, into medium- and high-level features for their use in decision-making models. An initial version of the algorithms was presented in deliverable D3.1. This document presents the final versions which are being used in the CONDUCTOR use cases.

2.1 Concept and Approach

The fusion of diverse mobility-related data (e.g., GPS, mobile records, surveys) enables the reconstruction of traffic and mobility patterns, which are essential for the development of effective traffic and fleet management strategies. Each data source contributes complementary information, enriching the overall understanding of mobility patterns. Moreover, different data source combinations can yield similar mobility indicators, demonstrating the method's flexibility. This approach enhances the understanding of mobility patterns and supports the optimization of transport systems.

Within the CONDUCTOR project, seven data fusion developments were identified by Nommon, INTRA, NTUA, TUM, and JSI to support the design of new traffic and fleet management strategies:

- Characterisation of delivery trips and estimation of delivery demand from mobile network, surveys and logistics operation data.
- Shared mobility demand estimation.
- Enrichment of users' profile.
- Household size assignment.
- Identification of unusual traffic patterns caused by large-scale events.
- Coupled Aimsun-FleetPy simulation data.
- Space-time context and heterogeneous data fusion.

Each development is framed within one of the CONDUCTOR UCs. Nevertheless, a from-particular-to-general approach was followed during the definition and implementation phases, ensuring that the methodologies can be extrapolated to other contexts, provided that similar data sources are available.

2.2 Characterisation of last-mile delivery trips and estimation of last-mile delivery demand from mobile network, surveys, and logistic operation data

The objective of this algorithm is the estimation of last-mile delivery demand volume and the characterisation of last-mile delivery trips to generate the last-mile delivery demand requests to be used in the Urban logistics UC of the project. The algorithm is implemented in two phases of incremental detail:

- **phase 1**: estimation of the last-mile delivery demand from surveys. In this phase, the delivery demand data provided by the Spanish National Statistics Institute aggregated at Spanish province level is disaggregated into smaller administrative levels, such as district or census tract, based on the population sociodemographic characteristics. This methodology was applied to obtain the last-mile delivery demand of an average month of the year 2023 in each district of Madrid for different population groups (see deliverable D3.1, Section 4.2.1).
- **phase 2**: characterization of delivery trips. In this phase, a dataset of historical demand data of one logistic operator of Madrid is used to characterize last-mile delivery trips.

Next, the methodology defined, and results obtained are described.

2.2.1 Data used

For this development, the following data were used:

- **Last-mile delivery trips data**. These data are provided by CITYlogin, a last-mile delivery logistic company located in Madrid, that has shared their delivery data, previously anonymised, with Nommon under private agreement for the context of this project. The data provided include information of goods travel demand, including delivery stops and delivery itineraries for one month (2242 routes in total). Specifically, for each stop the following information is provided:
    - the date of the delivery,
    - the vehicle ID (this information is only available when the parcel is delivered),
    - the sequence number of the delivery within the route of the day (this information is only available when the parcel is delivered),
    - the address of the delivery,
    - the coordinates of the delivery (latitude, longitude),
    - the estimated time of delivery (ETA) (this information is only available when the parcel is delivered),
    - information about the number, size, and weight of the products delivered.
- **Open-Source Routing Machine (OSRM) data**. OSRM is an open-source routing engine that provides routing planning from a set of coordinates and network information. These data were used to reconstruct the last-mile delivery trajectories based on the delivery coordinates.
- **Mobile Network Data (MND)**. These data are provided by one of the largest telecom companies in Spain and contains mobile phone Call Detail Records (CDRs) and probes data at antenna level of the company's users, sociodemographic information (age, gender, nationality), and the antenna network.
- **Points of interest (POIs)**. This dataset was generated by Nommon and contains the location of logistic and delivery hubs in the Madrid Region.

2.2.2 Methodology

The methodology consists of the following steps:

1. **Trips preprocessing**. The route, sequence, and ETA information only appear when the parcel is delivered. So, some routes have inconsecutive delivery sequences (e.g., 1, 2, 5, 7, 9, 10, 11, 12). The missing numbers of the sequence correspond to parcels that were not delivered, and in consequence, were not registered. This missing information causes the route to have gaps. To characterise these trips, the complete route is needed (since, even though the parcel is not delivered, the route is travelled), hence, the trips are filtered based on the number of missing deliveries they have in the sequence according to the following criteria:
    i. Filter routes with low percentage of parcels delivered. For that, the number of parcels in the route is assumed to be the highest sequence number.

   ii. Reconstruction of routes. For those remaining routes with gaps, if the distance between the last deliver before the gap and the first deliver after it is less than 300 metres, and there are no more than 2 consecutive missing parcels, the route is reconstructed by interpolating to midpoints (considering straight line) and mean times. The routes with gaps that do not meet this condition are removed.
   iii. Filter very short or very large routes, in terms of both trip duration and number of deliveries.
2. **Trips feature identification**. Next, features characterising a typical logistic trip are identified and computed for the data pre-processed in the previous step. These features include, among others, travel time and distance related metrics, frequency of appearance in logistic centre, and radius of gyration.
3. **Pattern extraction and characterisation**. Using the features computed in the previous step, the mobility patterns are extracted using unsupervised ML techniques. Specifically, the k-means clustering classification algorithm is used. To obtain the optimal number of clusters, the algorithm is run for different values of k (i.e., number of clusters), and for each of the results the inertia score is computed to measure the quality of the clusters. To find the optimal value of k, the elbow method was used, which involves plotting the inertia score against the number of clusters, for all the values of k. As k increases, the inertia decreases, but after a certain point, the rate of decrease slows down, forming an 'elbow' in the curve. This elbow represents the optimal number of clusters, balancing model complexity and clustering quality. Once the clusters are obtained, the patterns obtained are characterised based on the features distribution.
4. **Projection of last-mile delivery GPS traces to the MND antenna network**. Once an initial pattern characterisation is achieved, the delivery trips are projected to the MND antenna network. For that, the MND antenna network is used to tile the region into the coverage areas of each antenna, using a Voronoi tessellation. The zones or polygons obtained with this tessellation are called Voronoi polygons or simply Voronois. This way, the trajectories are translated into Voronoi delivery trajectories, with the same spatial resolution as the event data provided by the MND, making both data sources comparable.
   This process is done in three steps. First, OSRM is used to generate traces that join the GPS coordinates into the network. This trace is then projected to the Voronoi tessellation. Finally, each delivery point is mapped to the Voronoi polygon that contains it, and the trajectory is translated in terms of Voronoi polygons instead of GPS coordinates.
5. **Trip feature identification**. Using the routes projected in previous step at a Voronoi level, a new set of features characterising typical logistic trips, more appropriate for this spatial resolution, are identified and computed.
6. **Pattern extraction and characterisation at Voronoi level**. Using the features computed in the previous step at Voronoi polygon level, the mobility patterns are extracted using unsupervised ML techniques. Specifically, the k-means clustering classification algorithm is used. To obtain the optimal number of clusters, the elbow method is applied again. Once the clusters are obtained, the patterns obtained are characterised based on the features distribution.

2.2.3 Results

Next, the results of the algorithm for the characterisation of last-mile delivery trips are described, grouped into the same sequential steps of the development.

1. **Trips preprocessing**.

i. Figure 3 shows the delivery percentage distribution of all the routes of the dataset, in which the percentage bins are shown in steps of 0.05. Based on this figure, only routes with at least 95% of the deliveries performed are kept. After this filter, 1199 trips remained (out of 2242, 53.50%).
ii. After the reconstruction criterion, 1176 trips remained.
iii. Figure 4 shows the distribution of the deliveries in terms of the number of deliveries and duration. Based on this information, those routes lasting less than 4 hours or more than 12, and with less than 30 parcels delivered are removed. After this filtering, 206 trips remained (9.2% of the total routes).

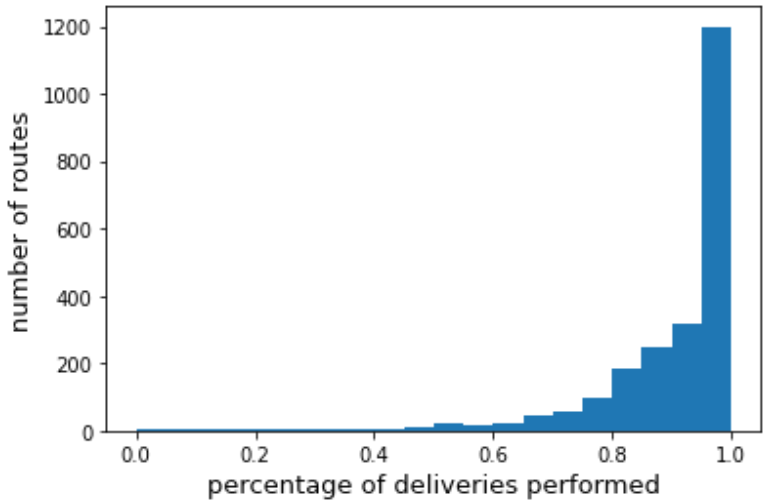

Figure 3: Delivery percentage distribution

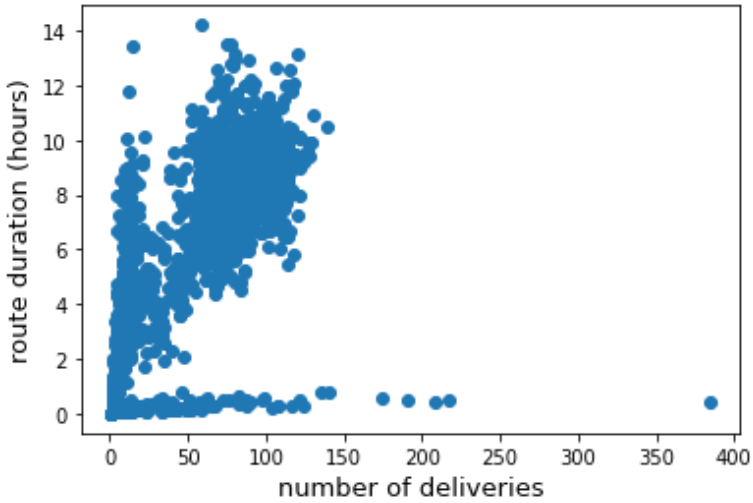

Figure 4: Distribution of the deliveries in terms of the number of deliveries and duration

2. **Trips feature identification**. The features considered are:
   - ratio of radius of gyration and travelled distance,
   - ratio of inter-delivery average distance and maximum inter-delivery distance,
   - ratio of length of the maximum sequence of deliveries of less than 1 km and the total number of deliveries,
   - ratio of length of the maximum sequence of deliveries of less than 15 minutes and the total number of deliveries,

- maximum number of deliveries in 1 hour,
- number of delivery sequences of less than 1 km,
- number of delivery sequences of less than 15 min.

3. **Pattern extraction and characterisation**. Figure 5 shows the inertia score obtained for different values of k. As can be seen, there is an elbow at 3. This is the number of clusters selected. Figure 6 depicts the three patterns obtained (cluster 1 in light blue, cluster 2 in dark blue, and cluster 3 in red). These patterns can be characterised as:
   - Pattern 1 (light blue): urban delivery trips with small radius of gyration and short delivery times and distances.
   - Pattern 2 (dark blue): longer urban delivery trips, with greater inter-delivery distances and larger radius of gyration.
   - Pattern 3 (red): interurban delivery trips between different municipalities, with large delivery times and distances.

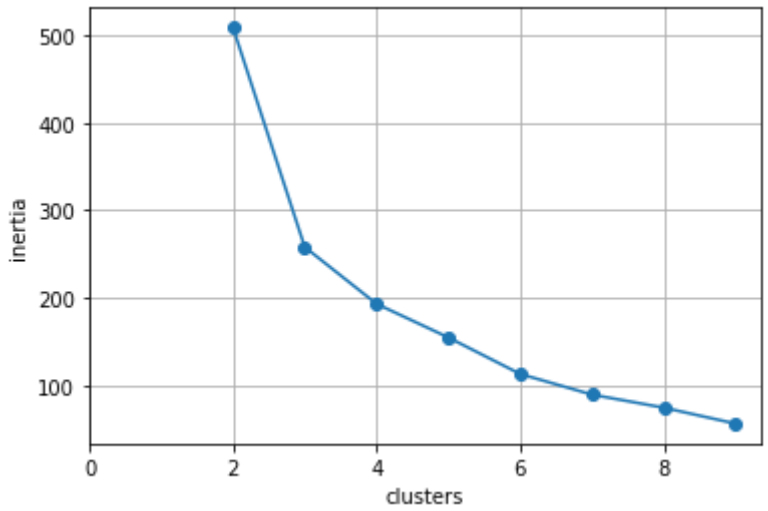

Figure 5: Inertia score for different number of clusters

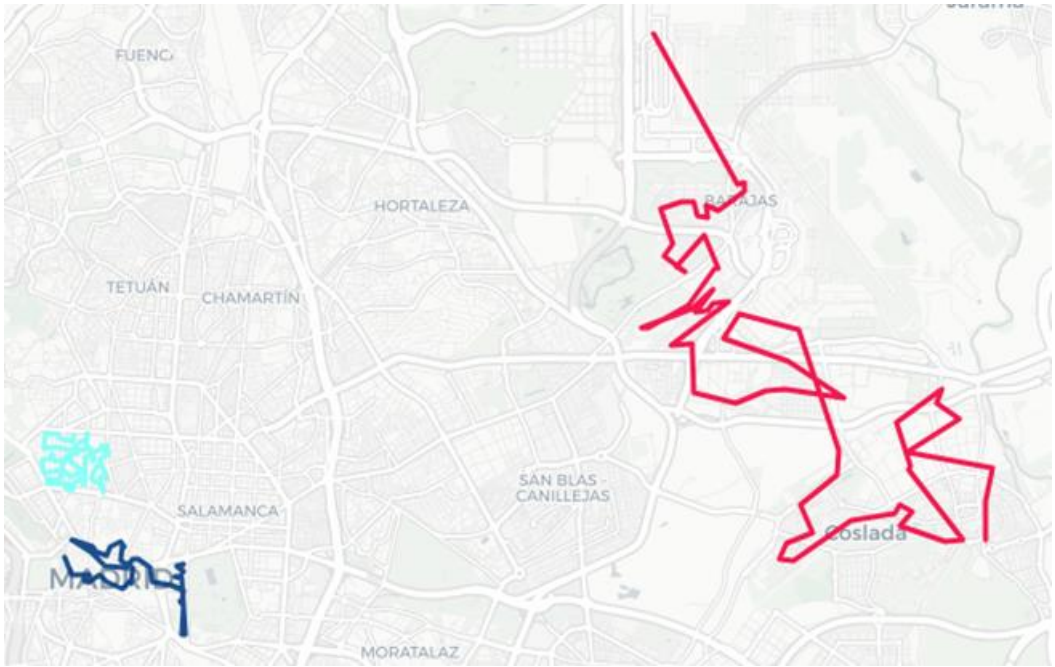

Figure 6: Examples of trajectories of each cluster

4. **Projection of last-mile delivery GPS traces to the MND antenna network**. Next, delivery trips are projected to the Voronoi tessellation. Figure 7 illustrates this process. In the right-hand side plot, the Voronoi polygon trajectory is generated using the centroids of the Voronoi polygons intersected with the route.

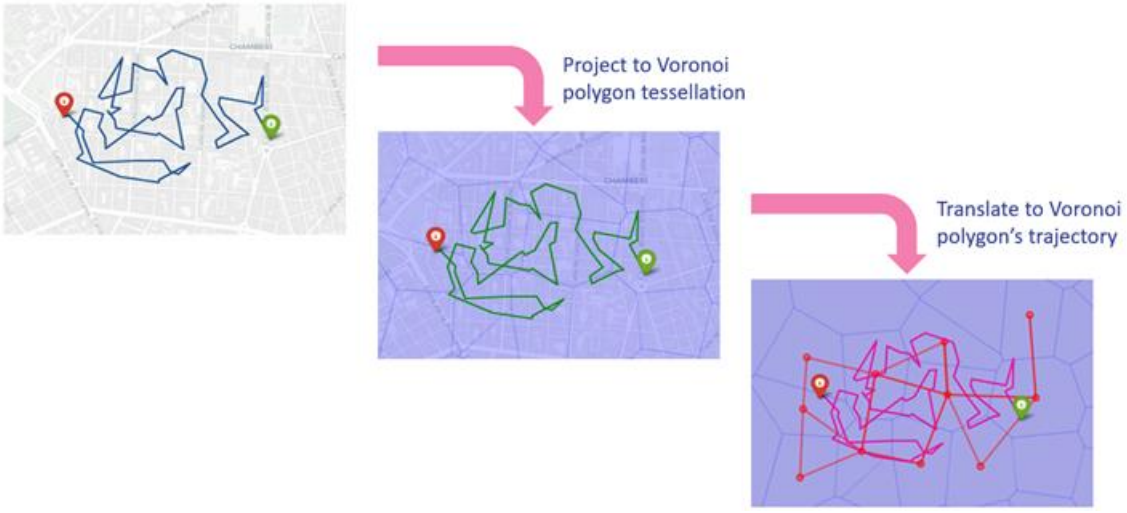

Figure 7: Projection of the last-mile delivery GPS traces to the MND antenna network

5. **Trips feature identification**. In this case, the features considered are:
   - radius of gyration of the route (in km),
   - travel distance (in km),
   - ratio between the radius of gyration and the travel distance,
   - average time in each Voronoi polygon visited during the route,
   - ratio between the number of unique Voronoi polygon visited and the total length of Voronoi polygons of the trip (to find loops),
   - ratio between the number of Voronoi polygons with multiple registers and the total number of Voronoi polygons visited,
   - time spent in the Voronoi polygons with multiple registers and the total travel time, and
   - ratio between the total travel time and the Q25, Q50, Q75, Q90 and maximum times in the Voronoi polygon.
6. **Pattern extraction and characterisation at a Voronoi level**. Figure 8 shows the inertia score obtained for different values of k. As can be seen, there is an elbow at 5. This is the number of clusters selected. Figure 9 shows the distribution of the features within each pattern. Based on these distributions, the patterns can be characterised as:
   - Pattern 0: Short interurban trips with many stops, small distance between Voronoi polygons of consecutive stops and medium radius of gyration.
   - Pattern 1: Long interurban trips with few stops (short Voronoi polygons sequence), big distance between Voronoi polygons of consecutive stops, and large radius of gyration.
   - Pattern 2: Short urban trips with many stops, very small radius of gyration, and distance between Voronoi polygons of consecutive stops (almost all adjacent).
   - Pattern 3: Long urban trips with less stops, small radius of gyration and very short time in the Voronoi polygon.
   - Pattern 4: Long logistic trips outside the Madrid region. Very big distance between Voronoi polygons of consecutive stops and very large radius of gyration.

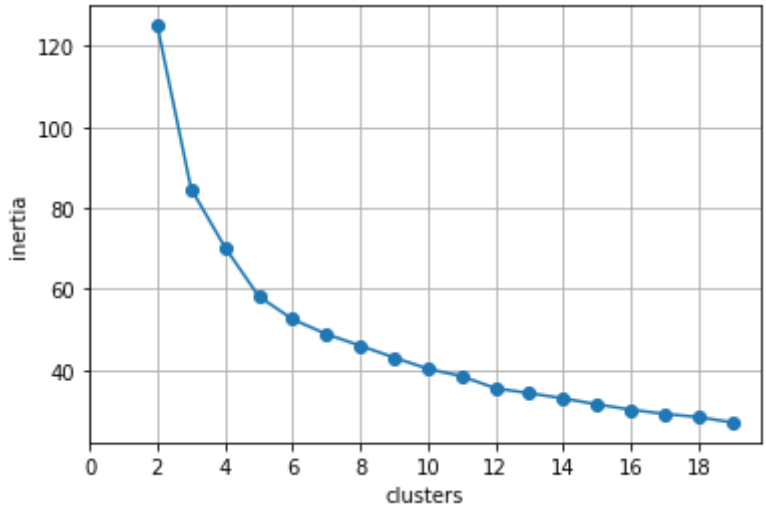

Figure 8: Inertia score for different number of clusters

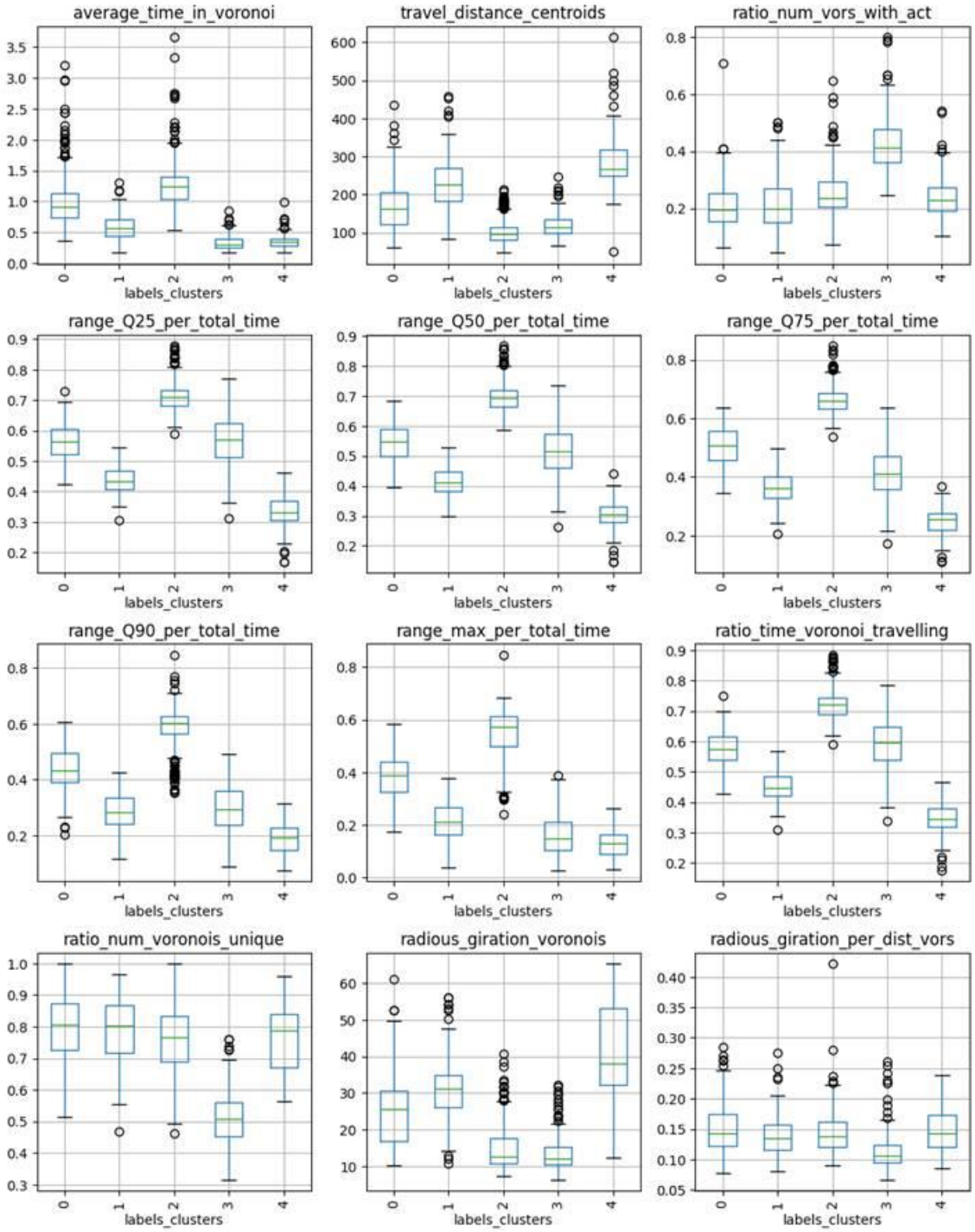

Figure 9: Distribution of the trip features in each of the five patterns obtained

2.3 Shared mobility demand estimation

The objective of this algorithm is the characterisation and estimation of future urban passenger transport demand for CCAM-enabled DRT services, as well as the estimation of the modes that will be substituted by CCAM. This will be used in the UC1-Madrid and UC3 of the project. The estimation of the shared mobility demand is structured into 2 main steps:

1. CCAM-DRT demand generation under different future scenarios.
2. Mode substitution model, to characterise the trips that are likely to shift to CCAM-DRT services.

2.3.1 Data used

The data sources used for this development are:

- **Spain census data**. This data source contains information from the Spanish National Statistical Office (INE), which provides demographic information for Spain at a high level of detail. The sample size is around 4.2 million people (10% of Spanish total population). In particular, the data considered to be useful is the Spanish Census, where population data, characterised by age group and gender is available at census tract level.
- **Land use data (SIOSE - HILUCS classification)**. National land use polygons classified using the INSPIRE HILUCS system, adapted for Spain and linked to the SIOSE database.
- **Sociodemographic data** (Comunidad de Madrid) from the INE. It provides sociodemographic indicators at the level of small administrative zones across the Madrid region. It includes metrics such as income inequality (P80/P20 ratio), average and salary income, population size and age structure, household composition, and the Gini index of each zone.
- **Carsharing demand data of Madrid**, from FLUCTUO, a shared mobility data aggregator. This dataset contains detailed records of carsharing trips in the carsharing operational areas of Madrid, used as a proxy for CCAM-enabled DRT demand, as carsharing services represent the most similar transport mode currently available in the city. The dataset spans a complete year, from June 2022 to May 2023 (both months included), and integrates information from all carsharing operators active in Madrid during that period. Each trip is included as an individual register in the dataset, with the trip start and end timestamps and geographical coordinates.
- **Mobility Household Survey (EMD, for its acronym in Spanish)**. The Mobility Household Survey is carried out by the Regional Transport Authority and analyses the Madrid residents' daily trips on a working day. It is based on 85.000 personal and telephone interviews carried out between February and May, 2018. The dataset includes the transport mode used for each trip, with over 300 trips recorded in taxi and ride-hailing services.
- **OD matrices with transport mode from Mobile Network Data (MND)**. Origin-Destination matrices based on district-level zones in Madrid (INE zoning), derived from mobile phone data provided by one of Spain's largest telecom companies. The dataset includes Call Detail Records (CDRs), probe data at the antenna level, the company's antenna network, and sociodemographic information (age, gender, nationality), which is incorporated —in an aggregated manner— in the output OD matrices.

2.3.2 Methodology

A flowchart illustrating the general methodology is presented in Figure 10. As previously mentioned, the approach consists of two main steps: (i) the generation of CCAM-DRT demand under different future scenarios, and (ii) the development and application of a mode substitution model.

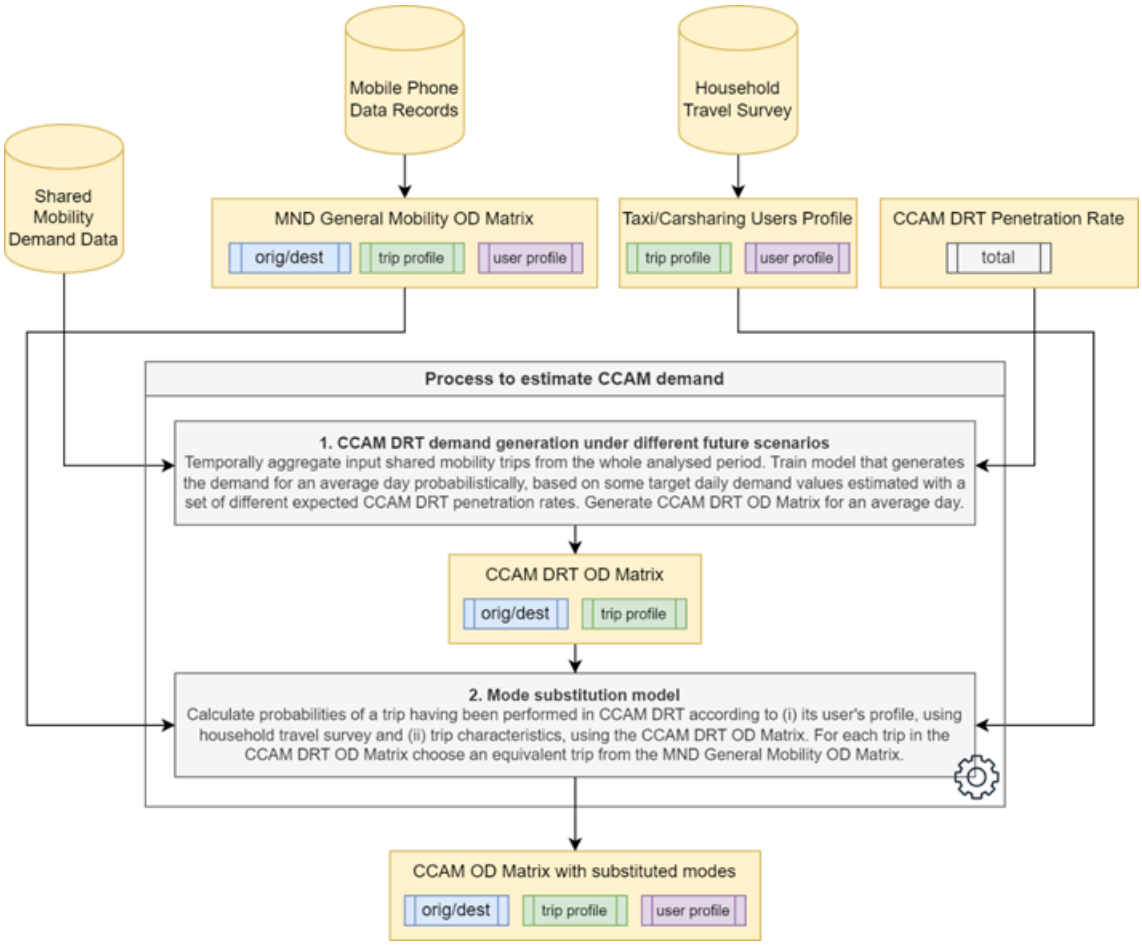

Figure 10: Shared mobility demand estimation methodology flowchart

1. CCAM-DRT demand generation under different future scenarios

In this stage, shared mobility trips—represented by the FLUCTUO dataset—are used as the main input, serving as a proxy to estimate the future demand for CCAM-DRT services. Since the observed demand from current shared mobility services is relatively low, these data are aggregated temporally (data from different days are grouped) so they can be analysed to train a demand generation model that learns the temporal and spatial patterns of usage. Specifically, the probability of trips occurring at each hour and from/to each geographical area is computed. These probabilities are then used to simulate the demand for an average day.

To define the total demand for this average day, a target number of trips must be set. For this purpose, a set of penetration scenarios—derived from the research conducted in the MOMENTUM project—is defined. These scenarios reflect different levels of expected future adoption of CCAM-DRT services.

The output of this stage is a set of synthetic trips—structured similarly to the input carsharing data—restricted to a defined study area, generated for a single representative day, and scaled to match the target daily demand corresponding to each penetration scenario.

2. Mode substitution model

To assess the modal impact of CCAM services—not only in terms of the number of trips generated but also in understanding which transport modes are likely to be replaced, and which types of travellers and trips are more likely to adopt these services—a mode substitution model is applied at this stage.

The model estimates the probability of each synthetic CCAM-DRT trip replacing a trip from another mode, based on behavioural insights drawn from the Mobility Household Survey for taxi and ride-hailing trips (user profile) and from the actual CCAM-DRT matrices produced by stage 1 of this algorithm (trip profile). Factors such as age, gender, distance and time of day are considered.

For each CCAM trip, an equivalent trip from the MND OD Matrix is selected based on the user and trip profile as previously mentioned. Since MND OD Matrix trips already have a current transport mode (private vehicle, public transport or walk), this way we can obtain the "mode of origin" of trips that shifted to CCAM-DRT services.

The output consists of a MND OD Matrix (containing origin mode, user and trip profile variables for each trip) enriched with an additional column that indicates which of the trips in the matrix shift to CCAM-DRT services under the specified penetration scenario.

2.3.3 Results

Next, the results obtained for the application of the shared mobility demand estimation algorithms are shown. The results are structured into 2 sections, showing the results of the corresponding main stage of the algorithm:

1. CCAM-DRT demand generation under different future scenarios

The estimated penetration rates for future scenarios, based on the MOMENTUM project deliverable ***D2.1 New Mobility Options and Urban Mobility: Challenges and Opportunities for Transport Planning and Modelling***, are described in Table 1.

Table 1: Estimated penetration rates for future CCAM DRT demand

	Scenario 1	Scenario 2	Scenario 3
Penetration rate	2% (82269 trips)	12% (493610 trips)	22% (904952 trips)

Based on these penetration rates and the FLUCTUO dataset, the demand generation model helps characterize the key features that impact the adoption of this transport service. The demand generation model is a supervised ML regression model, specifically a Gradient Boosting Regressor. It estimates the number of trips per transport zone and hourly interval, using a rich set of explanatory features derived from census data, land use, and sociodemographic indicators. The output of the model is then disaggregated into individual trips by drawing from the observed spatial-temporal distributions in the FLUCTUO dataset.

As seen in Table 2, the model achieved solid performance results, with an R^2 of 0.759, MAE of 412, and RMSE of 489241, showing its ability to capture a significant proportion of the variability in the observed data. Figure 11 presents a scatter plot comparing the actual trips (from FLUCTUO) with the estimated trips, including the identity line. The figure shows a good alignment between predicted and observed values for most of the range, although a few outliers appear in high-demand zones, where trips are generally underestimated. This deviation may be expected given the skewness of shared mobility usage across the city and the few zones that have such high trip volumes.

Table 2: Accuracy metrics of DRT demand estimation regression model

	R^2	MAE	RSME
Penetration rate	0.7589	412.04	489,240.80

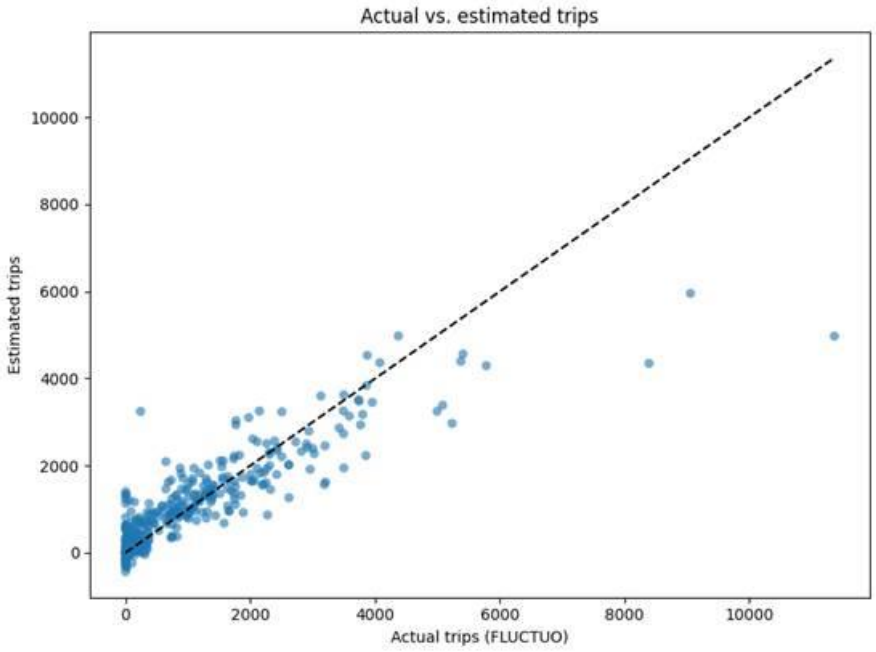

Figure 11: Scatter plot comparing the actual FLUCTUO trips with the trips estimated by the ML model

In terms of feature relevance, the most important predictors were (see Figure 12):

- "zone_lu_urban": urban land use proportion within the zone). Ranked highly, suggesting demand is concentrated in denser, urbanized areas.
- "zone_intersected_population": population within the zone to predict. Contributing strongly to the model, indicating a link between local population density and shared mobility demand.
- Other relevant features, including distance groups as well as different land-use areas and sociodemographic characteristics.

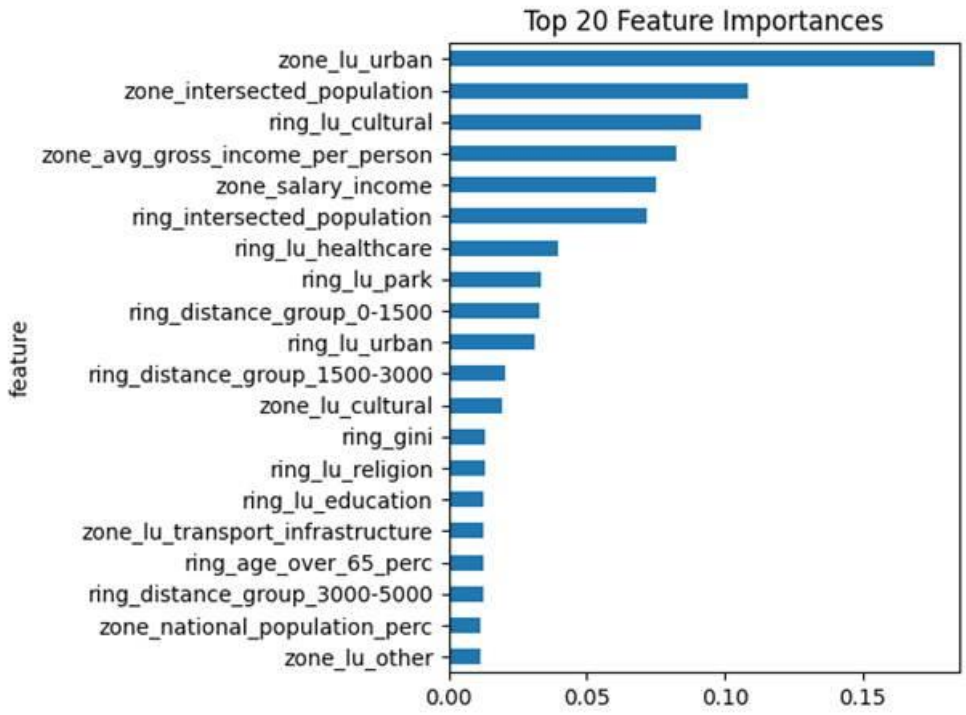

Figure 12: Feature importance of the ML trip estimator

The origin points of the trips generated for Scenario 3 are shown in Figure 13, whereas Figure 14 shows the trip distance distribution of the aforementioned trips.

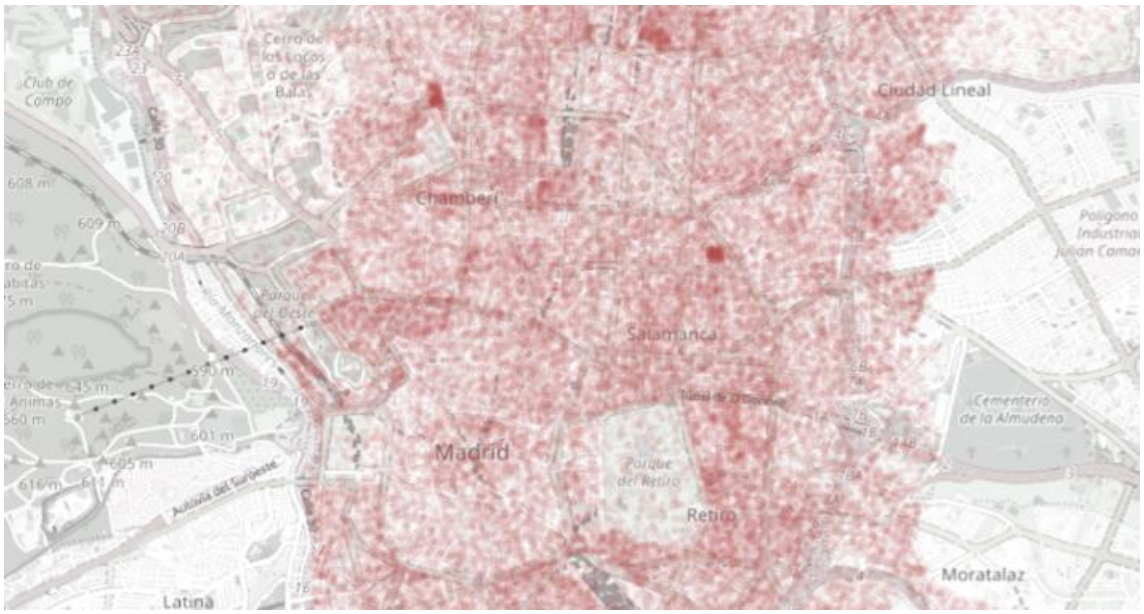

Figure 13: Visualisation of the origins of CCAM-DRT trips generated for Scenario 3

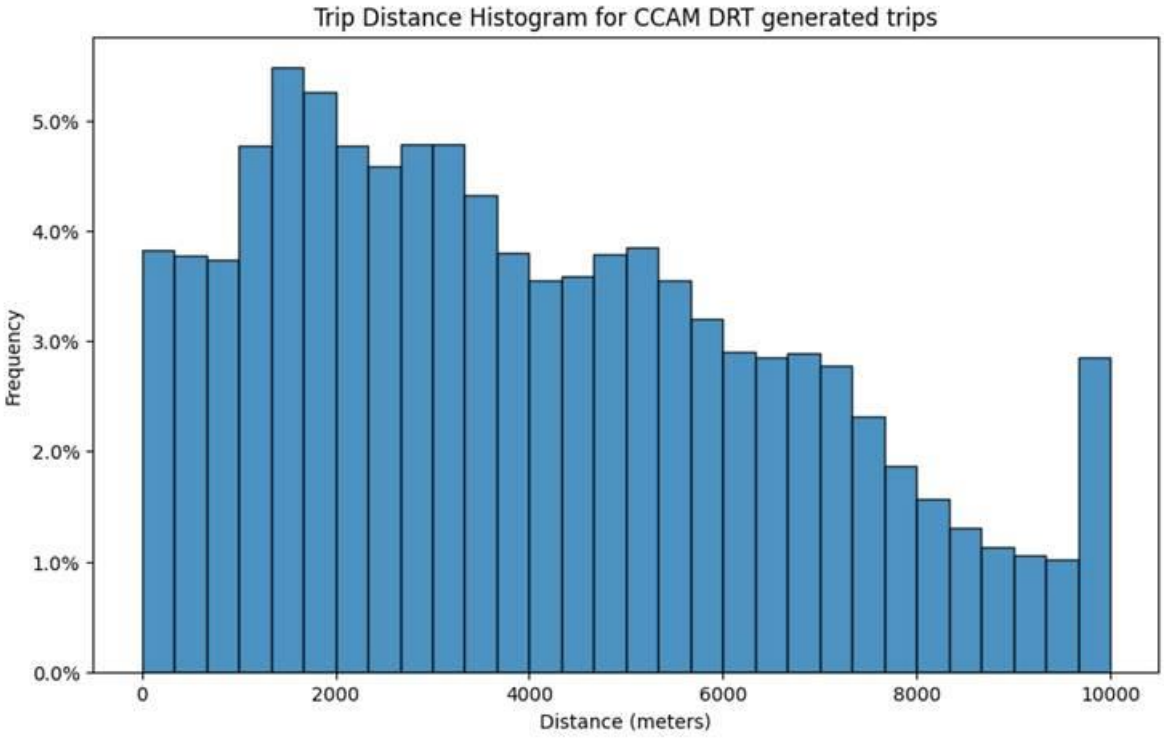

Figure 14: Trip distance distribution of the CCAM-DRT trips generated for Scenario 3

2. Mode substitution model

The mode substitution model builds upon a preliminary classification task aimed at estimating the likelihood that a given trip was made using CCAM-DRT, based on the socio-demographic and behavioural profile of the traveller and some trip characteristics (age, gender, residence municipality, trip distance) from the household travel survey. To address this, a Random Forest classification model was trained. This algorithm was selected due to its robustness, interpretability, and ability to handle complex, non-linear relationships between features.

Given the significant class imbalance in the survey data—specifically, the low number of trips made by taxi or ride-hailing services (fewer than 400 out of the total number of recorded trips)—the model was configured to compensate for this imbalance by assigning higher weights to underrepresented classes. This ensures that the classifier does not disproportionately favour the more frequent modes of transport and can effectively learn patterns associated with minority classes.

The model is not intended to assign a binary label to each trip, but rather to estimate the probability that a trip was made using CCAM-DRT. Thus, to evaluate its performance, we use the Receiver Operating Characteristic (ROC) curve, which illustrates the trade-off between true positive and false positive rates across different probability thresholds (there is no set probability threshold for this use case). The Area Under the Curve (AUC) serves as a summary metric of the model's discriminative ability. The following figure (Figure 15) shows the ROC curve, with an AUC of 0.79, indicating good predictive performance.

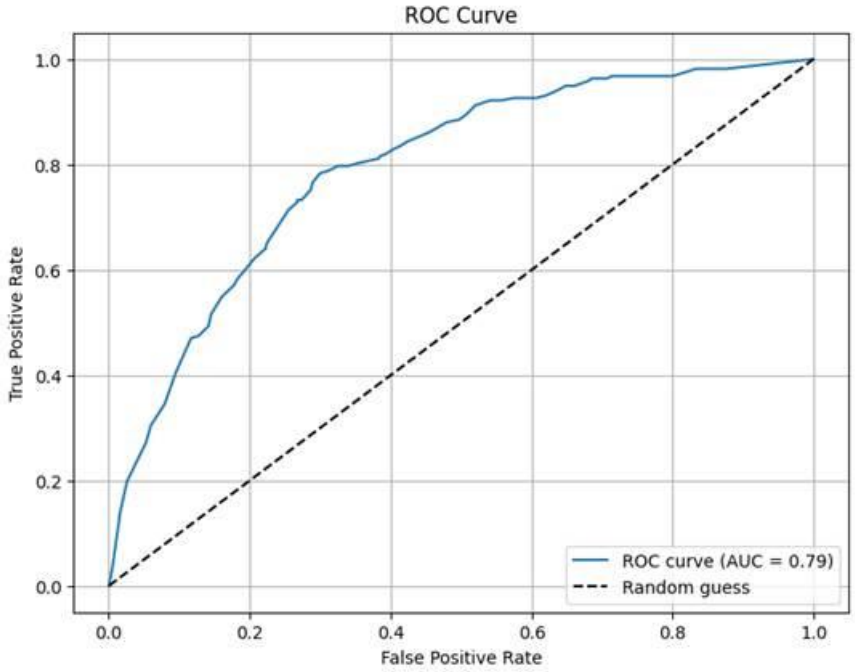

Figure 15: ROC Curve and AUC for the user profile classification model of the mode substitution model

The importance of the different features used for the model is shown in Figure 16:

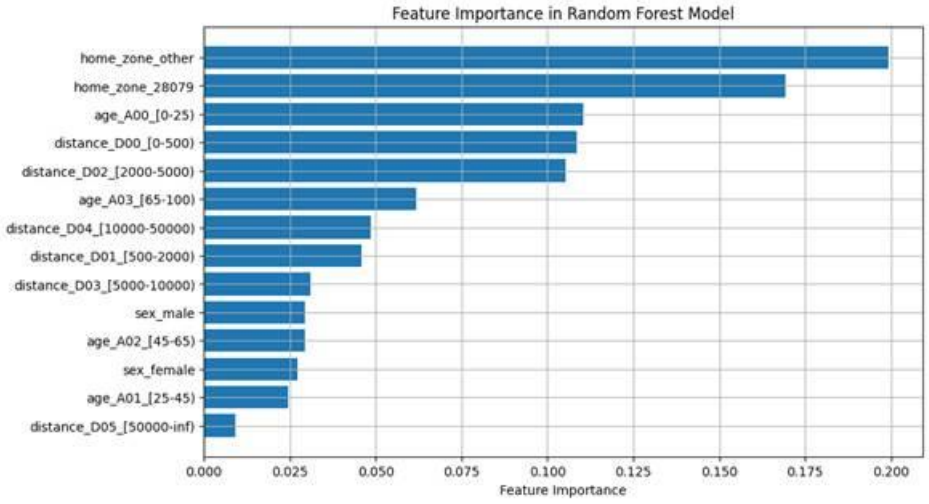

Figure 16: Features of the mode substitution model classification algorithm, ranked by importance

The mode substitution model results in an output that allows for the characterisation of the trips that will shift to CCAM-DRT services under future scenarios. The results indicate that the variables with the most influence in the adoption of these transport services from the ones that were analysed are trip distance, age and origin mode. The results shown below have been produced for Scenario 2 (with 12% penetration rate), although results from all scenarios are analogous.

In Figure 17, the age distribution of all trips in the original MND OD matrices is compared with the age distribution of the trips that were assigned to CCAM-DRT. As the user profile probabilities of this transport service were extracted from taxi and ride-hailing trips from the household transport survey, the results reflect the expected distribution: young travellers are less likely to choose this transport

mode, whereas the group with the most CCAM-DRT trips when compared to their total trips are people over 65 years old.

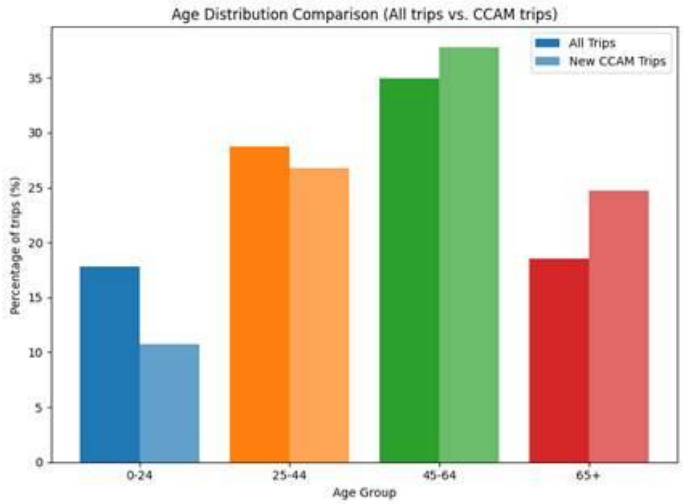

Figure 17: Mode substitution model results: age distribution comparison for generated CCAM-DRT trips

If we focus on trip distance, as expected, CCAM-DRT services are generally used for trips that are longer than the average trip (Figure 18).

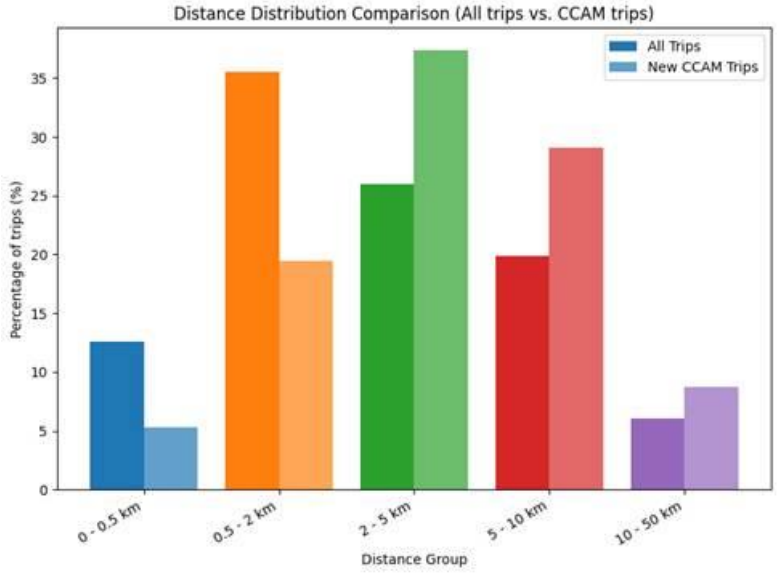

Figure 18: Mode substitution model results: distance distribution comparison for generated CCAM-DRT trips

Finally, the main result of the mode substitution model is the origin mode of the trips that would be substituted by CCAM-DRT (Figure 19). Compared to the total mode split (left), the mode split of the new CCAM-DRT trips (right) comprises a higher percentage of private vehicle and public transport, which adds up to almost 40% of the origin modes.

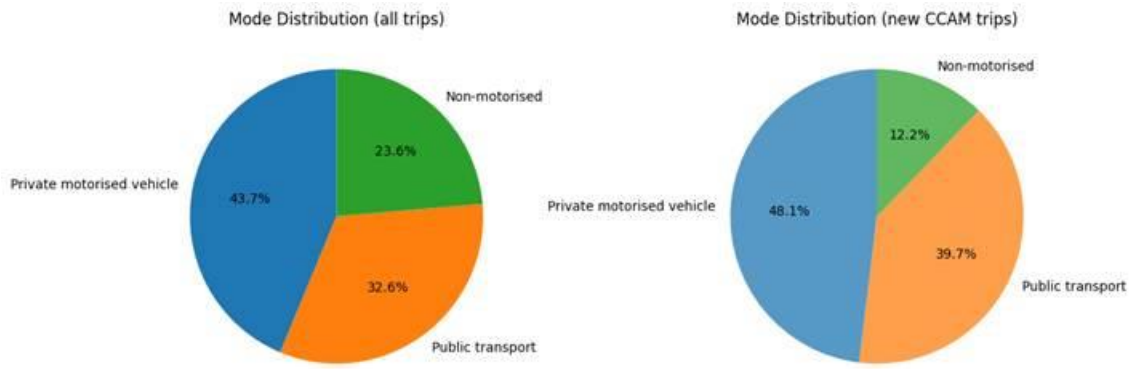

Figure 19: Mode substitution model results: origin mode quota for generated CCAM-DRT trips

2.4 Enrichment of users' profile

The objective of this development is the estimation of car ownership and household size features from travellers' mobility patterns. In particular, this development aims to enrich mobility-travel diaries obtained from MND, which are being used to estimate total demand and the potential impact of DRT-CCAM services on the demand of existing modes Madrid's Use Cases. Car ownership and household size are identified as relevant CCAM acceptance after a literature review (see Deliverable D1.1 Report on stakeholder requirements, user needs and social innovations). Hence, a better characterisation of the MND users' profile will allow a better estimation of the potential DRT-CCAM demand.

Next, the developments for the estimation of each feature are described.

2.4.1 Car ownership assignment

The objective of this development is to estimate whether a traveller owns a car or not. For that, machine learning (ML) techniques are used to fuse information from surveys and MND.

The hypothesis behind this approach is that a user's car ownership can be explained by their sociodemographic, economic features and mobility patterns, as well as those of individuals residing in the same area, and the transport services available in the area of residence.

2.4.2 Data used

The data sources used for this development are:

- **Spain census data** (same data source as in Section 2.3.1).
- **Household income distribution**. This data source contains information from the Spanish National Statistical Office (INE) about the average income of the residents per census tract level.
- **Public transport supply data**. This data source contains the location of the metro, train, and city and intercity buses stops. This information is provided by the Madrid Regional Transport Consortium (CRTM).
- **EMD** survey (same data source as in Section 2.3.1).
- **MND** (same data source as in Section 2.2.1).
- **Circulating fleet data**. Dataset that contains the total number of vehicles registered in each municipality of the Madrid region.

- **Vehicle register of the City of Madrid**. Dataset that contains all the vehicles registered in the City of Madrid at district and neighbourhood level.

2.4.3 Methodology

The basic idea of the implementation is to train a ML model on the EMD survey data and then apply it to predict car ownership of the MND users. For that, the most important part is the feature selection: the features used to train the model shall be available in both datasets, with the same spatial and temporal (if applicable) granularity.

This approach assumes that both datasets are representative of the population of the Madrid region, and hence, the patterns learned by the model from one of them are transferable to the other.

The methodology defined comprises the following steps:

1. **Features computation**. The features selected include:
   - user features: features characterising the sociodemographic and economic profile and mobility patterns of a user, and
   - zone features: sociodemographic features, mobility patterns of the population that resides in the same zone as the user and available public transport services.
2. **Geographical disaggregation**. To compute the features, it is very important to define an appropriate geographical resolution, in such a way that the zone features are informative and representative.
3. **Grouping of similar zones**. Users have very different behaviours depending on their area of residence, and zones have really different features depending on their location as well. Also, the resident sample in the survey is very irregular among the different municipalities of Madrid (because of the population distribution in the region). To avoid a bias towards overrepresented zones in the survey, preventing the model to learn the patterns of less populated zones, the zones are grouped in clusters using an unsupervised ML algorithm, specifically, the k-means algorithm.
4. **Data preparation**. To prepare the data for the ML analysis, the following steps are applied:
   - Zones selection: zones with small sample sizes are removed for training the model, as the patterns they can provide are most likely not representative of the user behaviour of the zone. Those zones with a sample less than 5 users per class in the survey were not considered for training.
   - Data split: as many regular ML developments, the dataset is split in two sets: training set and test set. The same split was applied for the three clusters, 80% for training and 15% for testing, stratifying the split by class and residence zone to ensure that both classes and all the zones are equally represented in both sets.
5. **Model selection and tuning**. For each cluster, a classification machine learning algorithm is trained to classify the survey users according to two classes:
   - Class 1: car ownership = yes, and
   - Class 2: car ownership = no.

   To select the best model for each cluster, a two-step process is followed:

   1. Model selection: for each cluster, a performance comparison on the train set was conducted among Random Forest, Gradient Boosting, Adaptive Boosting, Stochastic Gradient Descent, and Multilayer Perceptron classifiers.
   2. Grid search: for the model selected in the previous step, a grid search was applied to determine the best values of some key parameters of the training process. This method considers all possible combination of the values provided for each parameter to train the model and provides the score for each combination. This allows the

selection of the parameters combination that yields the best score. The parameters included in the grid search are:

a. Class balanced ratio: the number of users per class is imbalanced, and this may cause that the model, to improve the accuracy results, prioritises one class above the other. To prevent this, a balanced class ratio is applied to the training set in the three clusters. This ratio establishes the percentage of the users belonging to class 1 that are picked to train the model.
b. Features selection: to identify the most relevant features, the Recursive Feature Elimination (RFE) technique is applied. RFE works by iteratively training the model, ranking feature importance, and eliminating the least significant features in each iteration until the optimal subset is found. This method helps reduce overfitting, improve generalization, and enhance computational efficiency by eliminating redundant or irrelevant variables.
c. Dimensionality reduction: the Principal Component Analysis (PCA) technique is applied. This technique transforms high-dimensional data into a lower-dimensional space while preserving as much variance as possible. This improves computational efficiency and enhances model performance by reducing the risk of overfitting. However, the transformed components may lack interpretability.

6. **Estimation of car ownership**. Based on the results of previous step, the models for each cluster are trained. Then, their predictive performance is evaluated on the test set in terms of five standard performance metrics for classification problems:
   - Accuracy. It is the proportion of users correctly classified.
   - Sensitivity (also called true positive rate). It is the quotient between the number of users of class 1 correctly classified and the actual number of users of that class.
   - Specificity. It is the quotient between the number of users of class 2 correctly classified and the actual number of users of that class.
   - F1-score. It is the harmonic mean of precision (the proportion of correctly predicted instances of class 1 out of all predicted instances as class 1) and sensitivity.
   - Area under the Receiver Operating Characteristic Curve (AUC ROC). It measures a classifier's ability to distinguish between classes. The ROC curve plots the sensitivity against (1 – specificity) (also called false positive rate) at various classification thresholds. The AUC represents the probability of correctly ranking users of class 1 over users of class 2. AUC values range from 0.5 (random performance) to 1.0 (perfect classification).

This methodology is tested and validated in the Madrid region. Nevertheless, it can be applied to any region in which similar information is available. Furthermore, it can be applied to assign any other profile feature as long as it meets the starting hypothesis.

2.4.4 Results

Next, the technical implementation of the methodology and results obtained are described, according to the sequential steps introduced in the previous section.

1. **Features computation**. The features computed, 70 in total, are:
   - User sociodemographic features: age, household size, income.
   - User mobility patterns: trip distance distribution (quartiles), commuting distance, percentage of trips per combination of origin and destination purposes (among home, work, or other).
   - Zone sociodemographic features: age distribution of the population (age ranges: 0-19, 20-39, 40-64, 65-74, ≥75), population density, public transport density (number of train, metro, and bus stops per zone area and per residents), whether the residence zone is

inside the Madrid M30 ring road, whether the work zone is inside the Madrid M30 ring road, distance to Madrid city centre, and number of registered cars in the zone.

- Zone mobility patterns: trip distance distribution (quartiles) of the zone residents, of commuting distance distribution (quartiles) of the zone residents, percentage of trips per combination of origin and destination purposes (among home, work, or other) of the zone residents, percentage of trips per mode of the zone residents (private vehicle, non-motorised, and other).

2. **Geographical disaggregation**. The features are computed for the following spatial disaggregation:
   - District level for the municipality of Madrid.
   - Municipality level for the rest of municipalities of the Madrid region.
3. **Grouping of similar zones**. Zones were grouped in three clusters using the k-means algorithm based on the population and the number of users in the survey with and without private vehicle (i.e., survey sample per zone). Figure 20 shows the population distribution and the survey sample distribution for each zone. Figure 21 shows these clusters. As can be seen, there is a clear correspondence between the population and sample distribution (Figure 20) and the clusters grouping (left-hand side of Figure 21).
4. **Data preparation**.
   - Zones selection: The zones considered and discarded are depicted in the right-hand side of Figure 21. As can be seen, all the discarded zones belong to cluster red (left-hand side of Figure 21), composed mostly of rural, less populated, areas of Madrid.
   - Data split: Table 3 shows the number of instances in each set per class.

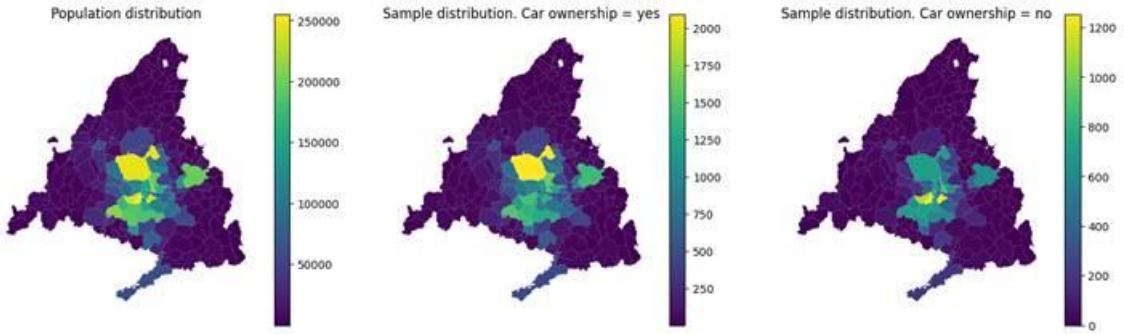

Figure 20: Population distribution (left) and survey sample distribution per class (centre and right)

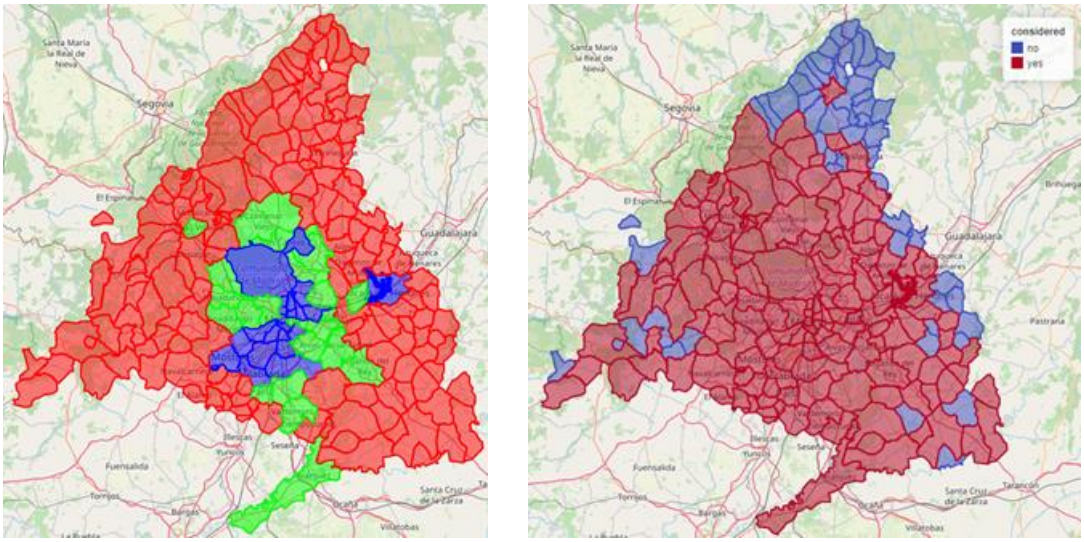

Figure 21: Cluster of zones (left). Classification of zones to train the ML models (right): in red, zones considered, and, in blue, the ones discarded

Table 3: Number of instances in each set per class. The balanced ratio is also included. It is computed as the number of instances of the class 1 by the total number of instances in the set

Classes distribution	Blue cluster		Green cluster		Red cluster	
	Train (80%)	Test (20%)	Train (80%)	Test (20%)	Train (80%)	Test (20%)
Class 1	21111	5279	11959	2991	5945	1488
Class 2	12397	3099	4371	1092	1678	418
Balanced ratio	0.63	0.63	0.72	0.73	0.78	0.78

5. **Model selection and tuning**.
    1. Model selection: In all three cases, Random Forest was the best-performing algorithm, so it was the selected choice.
    2. Grid search: Table 4 shows the list of values considered for each parameter for the grid search process on each cluster. The best combination of parameters for each cluster is presented in Table 5.

Table 4: List of values for each parameter for the grid search

Parameter	Values
Class balanced ratio	[0.7, 0.6, 0.55, 0.5, 0.45, 0.4, 0.3]
Number features RFE	[5, 10, 15, 20, 25, 30, 35, 40]
Number variables PCA	[3, 5, 8, 10, 15, 20, 25, 30, 35, no PCA]

Table 5: Best combination of parameters from the grid search process for each cluster

Parameter	Class balanced ratio	Number features RFE	Number variables PCA
Blue cluster	0.45	15	no PCA
Green cluster	0.50	15	no PCA
Red cluster	0.55	15	no PCA

The 15 more relevant features identified by the RFE technique for each cluster, and hence used to train each model, are:

- Blue cluster: age, household size, income, population density, user commuting distance, trip distance distribution of the user (the 4 quartiles), percentage of residents in the age ranges 20-39 and 65-74, percentage of trips per mode of the zone residents (only private vehicle and other), and percentage of user trips with OD purpose home-other.
- Green cluster: age, household size, income, population density, user commuting distance, trip distance distribution of the user (the 4 quartiles), percentage of residents in the age range 0-19, percentage of trips per mode of the zone residents (only private vehicle and non-motorised), percentage of user trips with OD purpose home-other, and bus stops density (number of bus stops per residents).
- Red cluster: age, household size, income, population density, user commuting distance, trip distance distribution of the user (the 4 quartiles), percentage of residents in the age ranges 0-19 and 75-101, percentage of trips per mode of the zone residents (only private vehicle and other), first quartile of the trip distance distribution of the residents.

6. **Estimation of car ownership**. One Random Forest was trained for each cluster using the parameters obtained from the grid search process, i.e., applying the balanced ratio on the class 1 of the training set and RFE with 15 features. Table 6 presents the predictive performance of the models for each cluster in the test set, based on the five previously described metrics: accuracy, F1-score, sensitivity, specificity, and AUC ROC. As can be seen, the three models demonstrate a satisfactory performance, achieving F1-score values above 0.80 and AUC ROC values ranging from 0.85 and 0.90. These results indicate that the models have effectively learned the patterns that characterize each class, successfully distinguishing between both classes and their respective behaviours.
   Figure 22 shows the accuracy of the three models on the test set for each zone. The models achieve an accuracy above 0.80 in most of the zones. As can be seen, the zones with worse predictive results (in dark) correspond to those zones not considered for training (zones in blue in the right-hand side plot of Figure 21). Figure 23 shows the accuracy distribution per zone, distinguishing between those zones used for training or not. Despite the model for the red cluster has never seen those zones before, it manages to predict with relative precision, showing its ability to generalise to new zones. This shows that the features selected effectively characterise the car ownership of the users.

Table 6: Predictive performance of the model for each cluster on the test set

Metric	Accuracy	F1-score	Sensitivity	Specificity	AUC ROC
Blue cluster	0.81	0.82	0.75	0.95	0.85
Green cluster	0.86	0.87	0.84	0.94	0.89
Red cluster	0.88	0.89	0.87	0.92	0.89

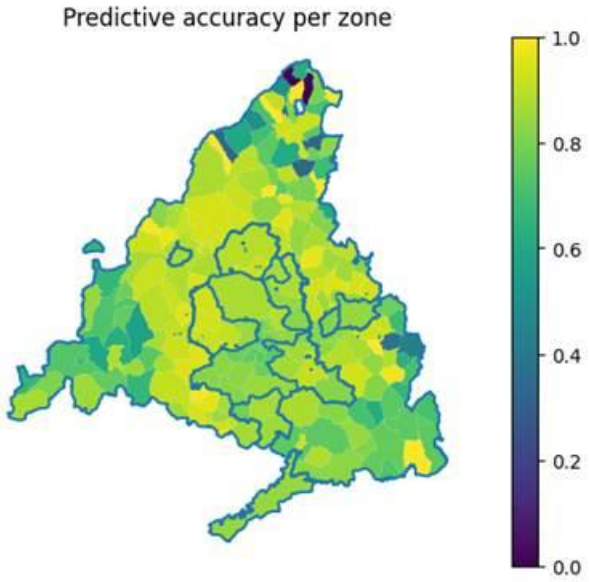

Figure 22: Predictive accuracy of the models per zone

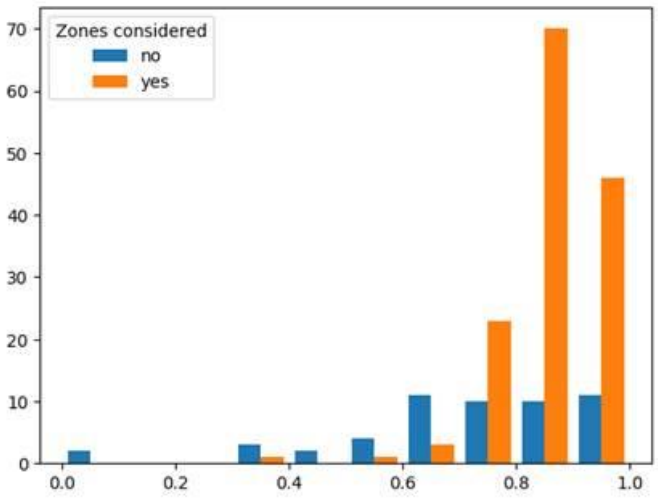

Figure 23: Accuracy distribution. In orange, zones included in the training process, in blue, zones not included

2.5 Household size assignment

The objective of this development is to estimate the household size of the MND users based on household information from surveys. For that, the household size distribution in terms of sociodemographic features and place of residence is used to probabilistically assign a household size to the MND users. A key part of this development is that the features used to distribute household size shall be available in both datasets (the surveys and the MND).

2.5.1 Data used

The data sources used for this development are:

- **Spain census data** (same data source as in Section 2.3.1).

- **Household size survey**. This data source
- contains information from the INE about the average household size and percentage of unipersonal households per census tract level.
- **Household distribution**. This data source contains information from the INE about the household size distribution per age group (with a granularity of 5 years, i.e., 0-4, 5-9, …) and Autonomous Community of Spain. The
- **MND** (same data source as in Section 2.2.1).

2.5.2 Methodology

An iterative algorithm for the assignment is defined, consisting of the following steps:

For each census tract:

1. **Assignment of the unipersonal households**. Based on the percentage of unipersonal households per census tract and the number of households (computed dividing the total population of the census tract by its average household size), the number of unipersonal households in the census tract is computed. Then, using the unipersonal household size distribution per age group in the Autonomous Community as a probability function, users are sampled one by one until the number of unipersonal households of the census tract is reached.
2. **Assignment of multi-person household size**. Those users not assigned in previous step are assigned a household size one by one according to the following process:
    1. Household size assignment. Get the household size distribution in the Autonomous Community for the population of the user's age group. Using this distribution as a probability function, assign household size to the user.
    2. Update household size distribution. As the average household size is provided at census tract level and the household size distribution per age group is provided at Autonomous Community level, this distribution may not match the one of the census tract (nor its average household size). To couple both, after each household size assignment, the distribution for the population of the assigned user's age group is updated taking into account the actual average household size of the census tract (provided by the INE) and the average household size of the assignment up to this point. This is done as follows:
        i. Compute the ratio between both average household sizes:

$$\text{ratio} = \frac{\text{average household size INE}}{\text{average household size assigment}}.$$

        ii. The percentage p_k of households of size k is updated as:

$$\widehat{p_k} = p_k * \text{ratio}^{-k},$$

        for all the households of sizes, $k \in K$. To keep this percentages as a probability, once all the percentages are updated, they are normalised:

$$\widetilde{p_k} = \frac{\widehat{p_k}}{\sum_k \widehat{p_k}}$$

        The distribution $\{k: \widetilde{p_k}\}_{k \in K}$ is the new household size distribution for that age group.

3. **Check average household size of the census tract**. For that, the average household size of the assignment is compared to the average household size according to the INE. Specifically, the stopping criterion established is:

$$\text{average household size assigment} =$$

$$\text{average household size INE} \pm 0.1 \cdot (\text{average household size INE})$$

If this condition is met, the assignment for this census tract is finished. Otherwise, step 2 is repeated user by user, checking the stopping criterion after each reassignment, until the condition is met.

This methodology is tested and validated in the Madrid region. Nevertheless, it can be applied to any region in which similar information is available. Furthermore, it can be applied to assign any other profile feature as long as similar information is available.

2.5.3 Results

Next, the results obtained for the application of the algorithm for the household size estimation of the Madrid sample users are shown. Figure 24 shows the scatter plot between the average household size per census tract provided by the INE and the one obtained with the assignment. The R2 score between both sets is 0.89, showing the effectiveness of the algorithm defined.

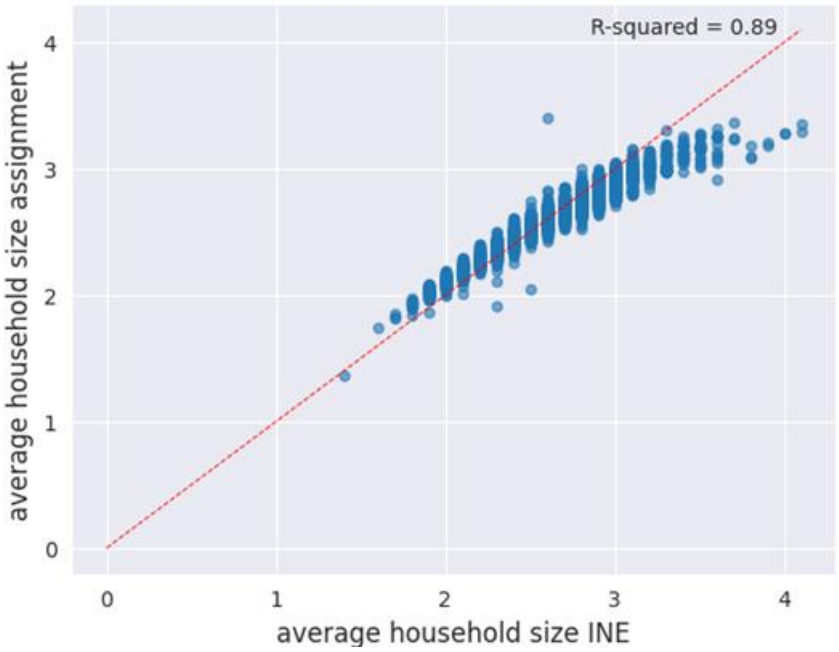

Figure 24: Scatter plot between the average household size per census tract provided by the INE and the one obtained with the assignment

2.6 Identification of unusual traffic patterns caused by large-scale events

An AI-powered decision support tool, titled “Traffic Events Risk Assessment and Route Selection” has been developed to enhance situational awareness and assist with mobility planning. The system comprises two main components: a Fuzzy Inference Engine to classify traffic conditions per road network node, as well as a Multi-criteria Decision Analysis Module that prioritises traffic events and recommends optimal routes.

2.6.1 Fuzzy Inference Engine

The Fuzzy Inference Engine applies fuzzy logic to evaluate and classify traffic conditions in a manner that closely resembles human reasoning. At the core of the model are three key input variables that reflect essential aspects of road traffic: vehicle density, traffic speed, and the gap between vehicles. Vehicle density represents the number of vehicles within a specific segment of the road network and is categorized as low, medium, or high. Traffic speed, which captures the average speed of moving vehicles, is classified as slow, normal, or fast. The third input, the gap between vehicles, measures the average spacing and is interpreted as short, medium, or long. Each of these input variables is defined over a continuous range and then translated into linguistic terms using membership functions.

The output of the model is a traffic condition score, which ranges from 0 to 100. This score is then interpreted into one of three descriptive categories: free-flowing (for scores below 40), moderate (for scores between 40 and 69), and congested (for scores of 70 and above). These labels provide an intuitive understanding of current road conditions while retaining the precision of the underlying numerical result.

The interpretation process is driven by a carefully designed rule base that combines the input variables in meaningful ways to produce realistic outcomes. Specifically, the system uses a set of 27 fuzzy logic rules that express conditional relationships between the input variables and the resulting traffic condition. For example:

- When vehicle density is high, traffic speed is low, and the gap between vehicles is short or medium, the system infers a congested condition.
- If vehicle density remains high but the speed is high and the gap is long, the model concludes that traffic is free-flowing, recognizing that density alone does not necessarily imply congestion.
- In cases where vehicle density is medium and the speed is normal or high with long gaps, the system typically labels the condition as free-flowing, acknowledging the smoother flow associated with moderate traffic volumes and good spacing.
- Conversely, a scenario involving medium density, low speeds, and short gaps results in a congested classification.
- For low vehicle density, the rules tend to favor free-flowing outcomes, particularly when speed is normal or high and vehicle spacing is adequate.

An API is available in the following link: Traffic Fuzzy Logic API - Swagger UI. The request body should be a JSON array of traffic data records. Each record should conform to the following structure:


```
[
    {
        "id": "0692-11",
        "dateUpdated": "2025-01-29T18:20:00",
        "countersLocation": "0692",
        "countersLocationDesc": "Zadvor",
        "countersRoadDesc": "R3-645",
        "countersSection": "1188",
        "countersDirection": "11",
        "countersDirectionDesc": "Ljubljana - Zadvor",
        "countersLaneDesc": "",
        "countersGeolocationX": "468775",
        "countersGeolocationY": "99786",
        "countersSpeedLimit": "50",
        "countersDate": "2025-01-29",
        "countersTime": "0001-01-01 19:20:00+00 BC",
        "countersNumberVehicles": "132",
        "countersAverageSpeed": "51",
```

```
            "countersGapBetweenVehicles": "25.1",
            "countersStatus": "1",
            "countersStatusDesc": "Normal traffic"
        },
        {
            "id": "0692-21",
            "dateUpdated": "2025-01-29T18:20:00",
            "countersLocation": "0692",
            "countersLocationDesc": "Zadvor",
            "countersRoadDesc": "R3-645",
            "countersSection": "1188",
            "countersDirection": "21",
            "countersDirectionDesc": "Zadvor - Ljubljana",
            "countersLaneDesc": "",
            "countersGeolocationX": "468775",
            "countersGeolocationY": "99786",
            "countersSpeedLimit": "50",
            "countersDate": "2025-01-29",
            "countersTime": "0001-01-01 19:20:00+00 BC",
            "countersNumberVehicles": "108",
            "countersAverageSpeed": "44",
            "countersGapBetweenVehicles": "35.7",
            "countersStatus": "1",
            "countersStatusDesc": "Normal traffic"
        },..]

```


The response will return a list of computed traffic conditions with the following structure:


```
[
    {
        "location": "Zadvor",
        "direction": "Ljubljana - Zadvor",
        "traffic_condition_score": 60.0,
        "traffic_condition_label": "Moderate"
    },
    {
        "location": "Zadvor",
        "direction": "Zadvor - Ljubljana",
        "traffic_condition_score": 50.0,
        "traffic_condition_label": "Moderate"
    },
    {
        "location": "Stara Cerkev",
        "direction": "Kočevje - Ljubljana",
        "traffic_condition_score": 62.0,
        "traffic_condition_label": "Moderate"
    },..]

```


2.6.2 Multi-criteria Decision Analysis Module

This service offers a decision support framework grounded in the TOPSIS (Technique for Order Preference by Similarity to Ideal Solution) methodology, an established and widely respected technique in multi-criteria decision analysis[2]. The process starts with the construction of a weighted decision matrix, where each alternative is assessed according to several criteria, each reflecting a different dimension of performance or value. These criteria are normalised and adjusted according

[2] Pandey, V., Komal & Dincer, H. (2023). A review on TOPSIS method and its extensions for different applications with recent development. *Soft Computing 27*, 18011-18039.

to user-defined importance weights to ensure a fair and meaningful comparison. The methodology then computes the geometric distance of each alternative from an ideal solution (defined by the best achievable performance across all criteria) and from an anti-ideal solution (representing the worst possible outcomes). Finally, the distances of each alternative from both the ideal and anti-ideal solutions are compared, allowing the calculation of a closeness coefficient for each alternative. Based on these coefficients, the service then scores and ranks the alternatives accordingly.

For example, consider a scenario where the alternatives represent different routes for a trip, and the criteria for evaluation include travel time, emissions, total distance, and the number of events detected in the road network (e.g., accidents, road closures, or traffic jams). Each route will be evaluated on these criteria: a faster travel time is preferred, lower emissions are better, shorter distance is more favourable, and fewer events detected generally indicate a smoother journey. The decision matrix is created by collecting data on each route's performance against these criteria. The service then normalises this data, ensuring that each criterion is measured on the same scale, and assigns weights based on the decision maker. Once the data is normalised and weighted, the service computes the distances of each route from the ideal solution, which could represent the best performance across all criteria (e.g., the quickest travel time, lowest emissions, shortest distance, and minimal events). By comparing the alternatives' distances to both the ideal and anti-ideal solutions, the service calculates a closeness coefficient for each route, indicating how closely each route approximates the ideal solution and how far it is from the worst outcome. Finally, the service scores and ranks the alternatives based on these closeness coefficients, helping the decision maker to easily identify the most optimal route.

The service does not stop at numerical output (i.e. scores). It includes a suite of visual analytics to enhance interpretability and transparency. Detailed line plots and radar charts illustrate how each alternative performs relative to others, highlighting strengths and weaknesses briefly (Figure 25 and Figure 26). Bar graphs break down the scores criterion by criterion, offering a deeper understanding of the decision drivers (Figure 27). Annotated ranking plots and directed graphs then summarise the overall results, guiding stakeholders clearly toward the best-informed decision (Figure 28 and Figure 29).

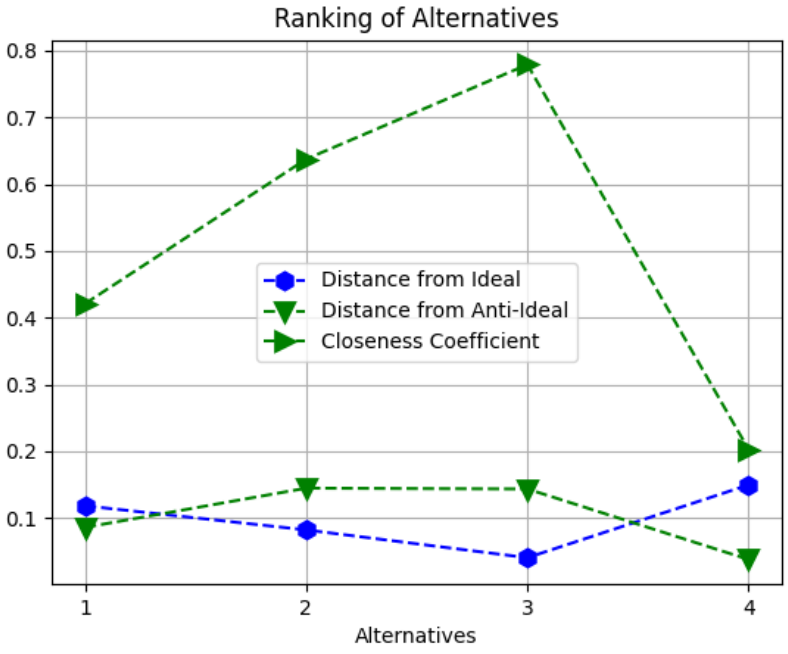

Figure 25: Line plot for illustrating how each alternative performs relative to others

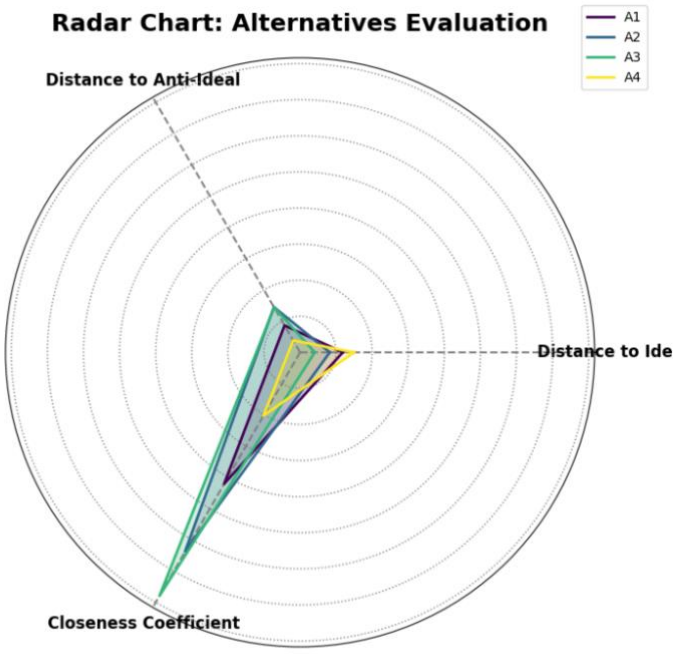

Figure 26: Radar chart for illustrating how each alternative performs relative to others

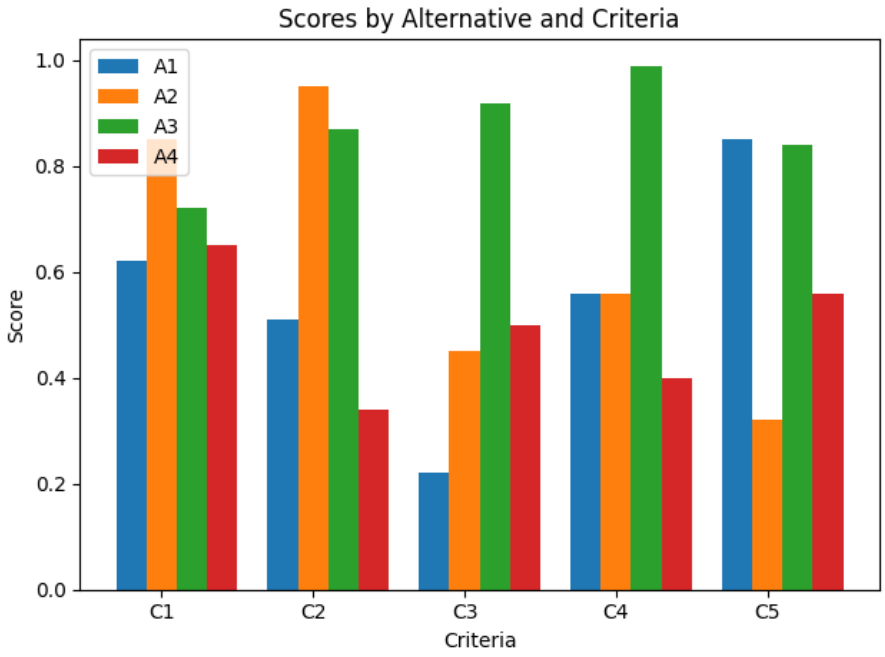

Figure 27: Bar graph for breaking down the scores criterion by criterion

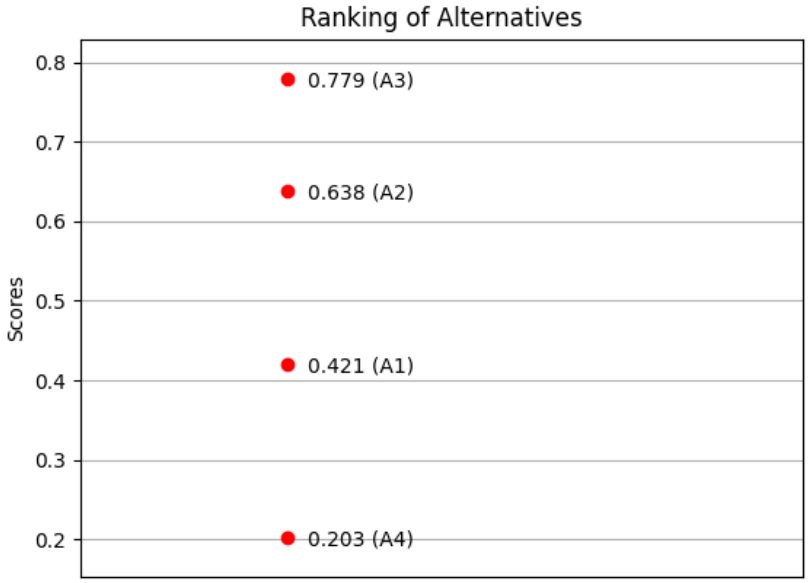

Figure 28: Annotated ranking plot for summarizing the overall results

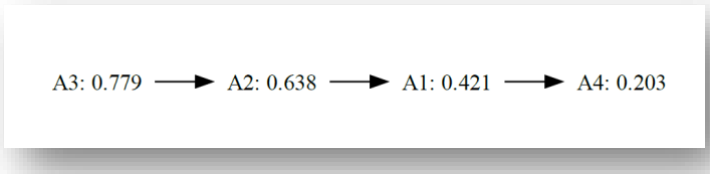

Figure 29: Directed graph for summarizing the overall results

An API is available in the following link: [Topsis Scoring API - Swagger UI](). The JSON request body is structured to include the decision matrix, weights, and other parameters necessary for the TOPSIS methodology. Here's a breakdown:

- matrix: This 2D array represents the decision matrix where each row corresponds to an alternative, and each column corresponds to a criterion.
- weights: A list of values representing the importance of each criterion.
- norm_method: Defines the normalization method to be used. "v" indicates vector normalization.
- ideal_solution_method: Specifies the method to determine the ideal solution. "m" indicates a method based on maximizing or minimizing values to define the ideal.
- plot_results: A Boolean value indicating whether visual analytics, such as plots and charts, should be included in the response.
- names_of_alternatives: A list of alternative names.
- names_of_criteria: A list of criterion names.

```
{
  "matrix": [
    [0.62, 0.51, 0.22, 0.56, 0.85],
    [0.85, 0.95, 0.45, 0.56, 0.32],
    [0.72, 0.87, 0.92, 0.99, 0.84],
    [0.65, 0.34, 0.5, 0.4, 0.56]
  ],
  "weights": [0.4, 0.3, 0.05, 0.05, 0.2],
  "norm_method": "v",
  "ideal_solution_method": "m",
  "plot_results": true,
  "names_of_alternatives": ["A1", "A2", "A3", "A4"],
  "names_of_criteria": ["C1", "C2", "C3", "C4", "C5"]
}
```

The JSON response provides the calculated closeness coefficient for each alternative, which indicates how closely each alternative approaches the ideal solution:

- closeness_coefficient: This array contains the closeness coefficient for each alternative. The higher the score, the closer the alternative is to the ideal solution.
    - Each object in the array includes:
        - alternative: The name of the alternative.
        - score: The closeness coefficient score for the alternative.

```
{
    "closeness_coefficient": [
        {
            "alternative": "A1",
            "score": 0.4209733307361603
        },
        {
            "alternative": "A2",
            "score": 0.6375610828399658
        },
        {
            "alternative": "A3",
            "score": 0.7791176438331604
        },
        {
            "alternative": "A4",
            "score": 0.20269949734210968
        }
    ],
    "execution_time": 20.7369,
    "message": "TOPSIS calculation completed successfully."
}
```

2.7 Coupled Aimsun-FleetPy Simulation Data

2.7.1 Introduction

TUM developed techniques for the efficient integration of urban logistics in DRT services. These techniques will be used to simulate UC3 in Madrid. The freight and DRT demand data for Madrid are generated by Nommon which are used as input for a co-simulation of Aimsun Next and FleetPy for the city of Madrid. The Madrid network will be provided by Aimsun.

FleetPy is a Python-based DRT simulation tool developed by TUM. It does not have an integrated traffic microsimulation functionality; the travel times are mainly calculated using scaled free-flow

travel times obtained from Open Street Map[1] . Thus, it lacks a detailed consideration of other vehicles participating in the overall traffic. To fill this gap, FleetPy is coupled with Aimsun Next using the Python API. The fleet is controlled, i.e., vehicle schedules are computed, in FleetPy while vehicle movements are conducted within the Aimsun environment. The bridge allows the consideration of a more realistic traffic simulation in the FleetPy control decisions which replicates the unexpected delays the DRT might face in real traffic. The details of the Aimsun-FleetPy bridge are provided in D2.1 (Specification and initial version of the adapted traffic and fleet management models) and D2.5 (D2.5 Final implementation of models).

In the D3.1, it was mentioned that the Aimsun-FleetPy bridge will be used for UC3 and the respective data required for the data was accordingly described. However, over the course of the project, it was decided that for the scope of Madrid UC3, the available microscopic simulation model for the motorway ring road M30 of Madrid, was not suitable for this application. Instead, a macroscopic transport model of the whole city of Madrid is used, covering the urban environment. This change is also mentioned in D2.5, where the technique of exporting skim matrices from Aimsun Next's macroscopic traffic assignment is described for incorporating realistic travel times in FleetPy. According to this modification, the following sections describes the main data used for Aimsun Next and FleetPy to simulate the UC3 in Madrid. The following sections are very similar to the corresponding sections in D3.1, however, necessary changes are made to reflect the adoption of macroscopic model for UC3 instead of microscopic model.

2.7.2 Data Used

The main data inputs for UC3 are provided by Nommon and Aimsun and fused together in a co-simulation of FleetPy and Aimsun Next. The co-simulation of Aimsun Next and FleetPy will not be in real-time as the Aimsun-FleetPy bridge for microscopic simulation will not be used. Rather, travel time of edges will be exported to FleetPy after separating macroscopic traffic assignments in Aimsun Next. Thus, the main data inputs to UC3 will consists of the following:

- A calibrated macroscopic (or mesoscopic) model of Madrid in Aimsun Next.
- The origin-destination (OD) pair of the DRT passenger request and the time when the request is made.
- The OD pair of the freight requests. UC3 assumes to serve freight requests from depots. Therefore, the origins will be freight requests will be limited to those depots.
- Rest of the FleetPy simulation parameters to describe the fleet control algorithms used.

2.7.3 Methodology

For a successful simulation of UC3, some manipulation of the input data is required as described below.

The first and the foremost is the export of the Aimsun Next's network to FleetPy. Python scripting in Aimsun Next is used for this purpose. Then, the freight and DRT demand data is mapped to the exported FleetPy network. For simplicity, FleetPy generally limits the locations that can be visited by the DRT fleet to be located on the city network nodes. Thus, instead of the exact geographical locations, the closest node of the city network is used. The pickup or delivery of freight or passengers are, therefore, not considered to be in between network edges, rather, they are assumed to be exactly on network nodes. To reduce the computational efforts of simulating UC3, it is also decided that a smaller set of nodes within Madrid network will be used for picking up and dropping off passengers as well as the freight requests. This smaller set of nodes are termed as meeting points. Thus, the DRT service considered will not be offered door-door service.

The second is regarding the clustering of the freight requests. The study assumes same-day freight requests, which are significantly less time critical than the DRT passenger requests and can be delivered at any time within the same day. However, it is assumed that the DRT service should at least provide some time window (ranging in hours) within which the freight requests are served. Since it is assumed that the freight requests are mainly last-mile delivery requests known in advance unlike the DRT passenger requests, the freight requests need to be clustered in a way that the estimated freight delivery time windows are fulfilled. Such a clustering can be geographically, temporally or a combination of both, performed in a preprocessing step. The main challenge faced in this regard is that the service quality of DRT passengers must not be compromised significantly. Geographically, these clusters do not need to be artificial regions, rather, they can also be based on administrative units such as city districts.

The third data manipulation is regarding the traffic state data collected from the Aimsun Next simulation. To plan vehicle routes and assign vehicles to DRT requests, FleetPy requires information on the travel times and travel distances between different OD pairs. For this purpose, the macroscopic simulation in Aimsun Next is carried out separately and the average travel time of each network edge is exported via skim matrices. These travel times are then used in FleetPy to calculate the travel times and distances using the Dijkstra algorithm.

2.8 Space-time context and heterogeneous data fusion

2.8.1 Introduction

In this section, we present the development of heterogeneous data fusion in the second part of the project and the usage in the UC2. The Context Graph is a graph-based data fusion framework that provides a semantic description of the stored entities while explicitly encoding their spatiotemporal and hierarchical context. By linking each data point to a structured context layer, it enables advanced reasoning, contextual similarity search, and automated feature extraction across heterogeneous data sources.

In the Context Graph, both contexts and entities are stored explicitly in the graph, while measurements and other time-varying variables are encoded as properties on the edges between them. For example, measuring and storing a city's temperature, measured on March 1st 2025, at 11:30, is done as:

```
MATCH (p:City {name: "Ljubljana"}), (c:CityDay{year: 2025, month: 3, day:
1}, CityHour{hour: 11, minutes: 30})
CREATE (c)-[:WEATHER_MEASUREMENT {Temp: 28}]->(p)
```

Contexts are structured hierarchically from the most coarse `CityMonth` context to the more granular `CityHour`. Spatial adjacency is represented with undirected edges `ADJ_SPACE` (implemented as bi-directional edges), while parental relationships are represented with edges `PARENT_SPACE`. Similarly, temporal adjacency is represented by directed edges `ADJ_TIME` and parental relationships are represented by edges `PARENT_TIME`. The figure below shows the explicit encoding of contexts.

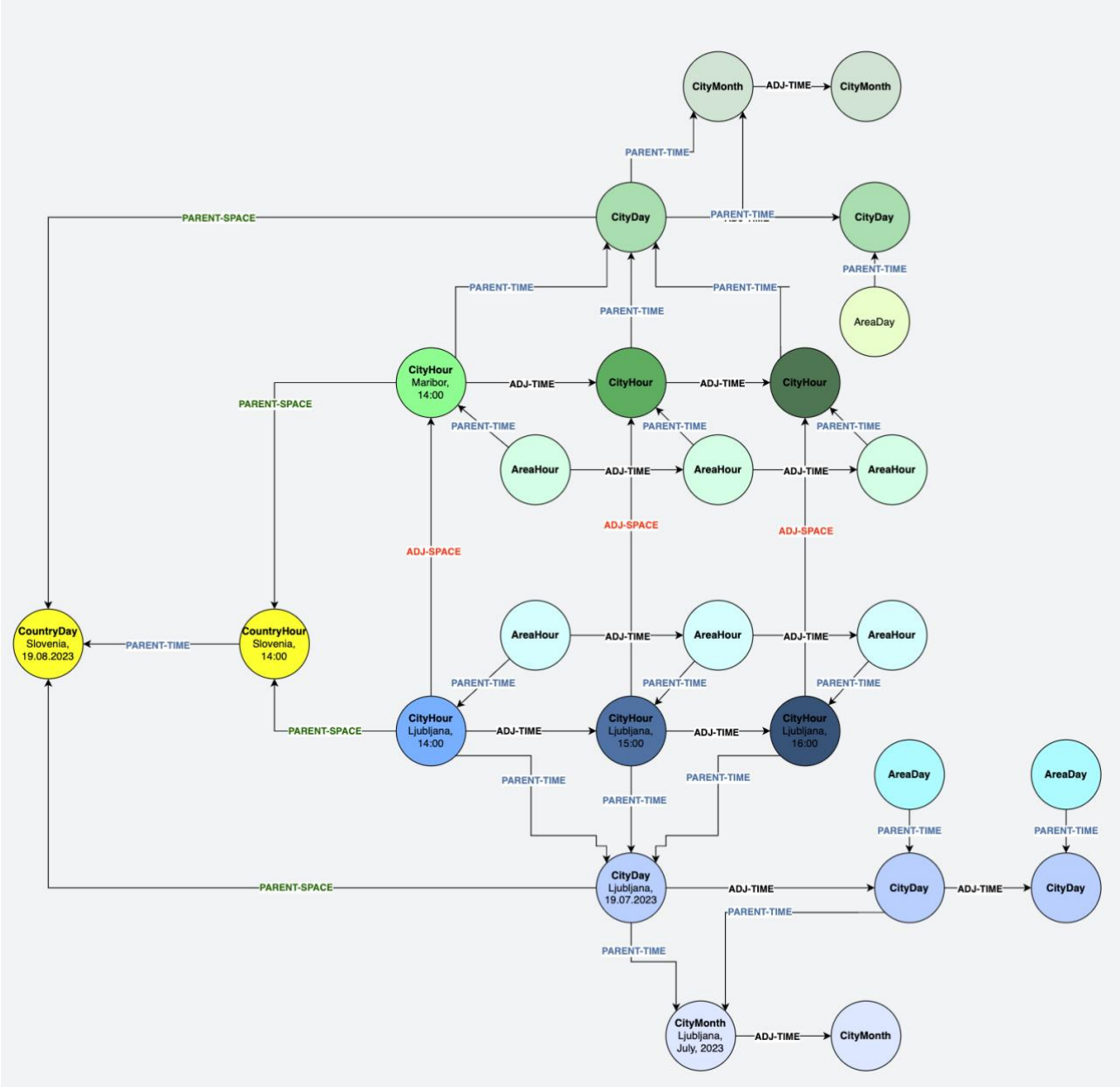

Figure 30: Explicit context encoding encodes points of spacetime as joint spacetime nodes

In the second part of the project, we scaled the Context Graph and further experimented with its design. We compared two different context encodings: (i) explicit context encoding, and (ii) implicit context encoding. Our results showed that explicit context encoding provides better performance for context-based retrieval and contextual data aggregation. This is largely because of the better indexing options.

Essentially, due to the difficulties with query-tweaking and slow data imports, further experimentation with the Context Graph was dropped. While the Context Graph does provide a viable option for storing heterogeneous data in a single structure, we found that the technologies are not mature for practical large-scale implementation. While in small-scale implementations graph algorithms can be used to aggregate local contextual information, the connected nature of the large-scale implementation, with super nodes (typically coarse contexts like `CityYear`), makes it impractical to automate queries in the API layer. After the graph reaches a certain scale, both the queries and the data model need to be redesigned. Due to the limitations of the current

graph databases, the queries must be analyzed and optimized to provide practical performance. Likewise, supernodes must be split in order to allow traversal and aggregation. A typical example is the Weather node, which would be connected to every `CityHour` and `CityDay` node in the context structure. To avoid a high degree, this node is split into several `WeatherMonth` nodes, which allows for practical traversal in our current graph scale.

2.8.2 Explicit vs Implicit Context Encoding

In this section, we present two alternative design options for the Context Graph. In the first alternative, called explicit context encoding, space and time are represented jointly in spacetime nodes (e.g. `CityHour`). This has the advantage of easier indexing and more efficient traversal. In this design, looking up a context only requires fetching a single node. The second alternative encodes the spatial and temporal contexts as disjoint structures. Here, looking up a context requires querying two nodes. However, in this representation less space is required to store the two disjoint contexts. This second alternative also seems to be a more intuitive option for many researchers. The explicit encoding is shown in the Figure 30 above, while the implicit encoding is shown in the Figure 31 below. In implicit spacetime contextualisation, the spatial and temporal contexts are encoded separately. In this representation, measurements are stored on hyperedges represented by the yellow nodes in the figure.

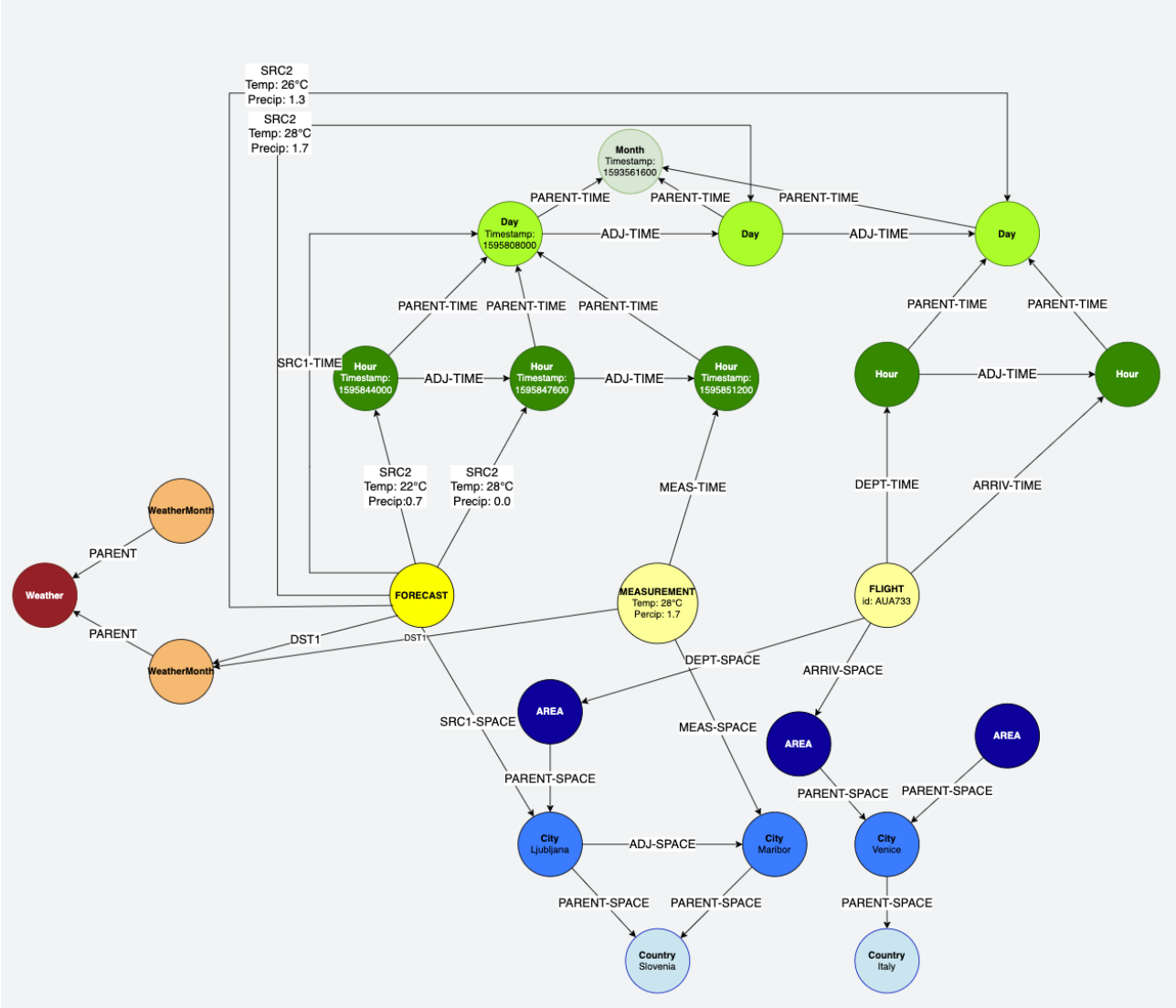

Figure 31: Implicit context encoding encodes points of spacetime as joint spacetime nodes

Our experience showed that explicit context encoding is more effective for context-based retrieval and aggregation use cases. When scaling the graph, the implicit approach frequently caused issues, requiring a long time to process relatively simple queries. This is especially true for aggregations over contexts. We therefore opted to stay with explicit context encoding.

2.8.3 Data Structures

Here we present some of the data structures stored in the Context Graph. We focus on the data most related to the Slovenian pilot which includes traffic data and weather data.

Traffic measurements, shown in the 32 below, are also contextualised by `CityHour`. Traffic measurements are stored on the link between the entity `CounterLocation` and context `CityHour`. Because of scaling issues, `CounterLocation` was broken up into several `COUNTER_MONTH` nodes. The location of the counter is stored as node `CounterLocation` and is linked to a road segment via edge `BELONGS` and then further to the nearest city. Because of scaling issues, these supernodes were broken up into `COUNTER_MONTH` nodes. The measurements are stored on the edge `TRAFFIC_MEASUREMENT` linking `COUNTER_MONTH` to its context `CityHour`.

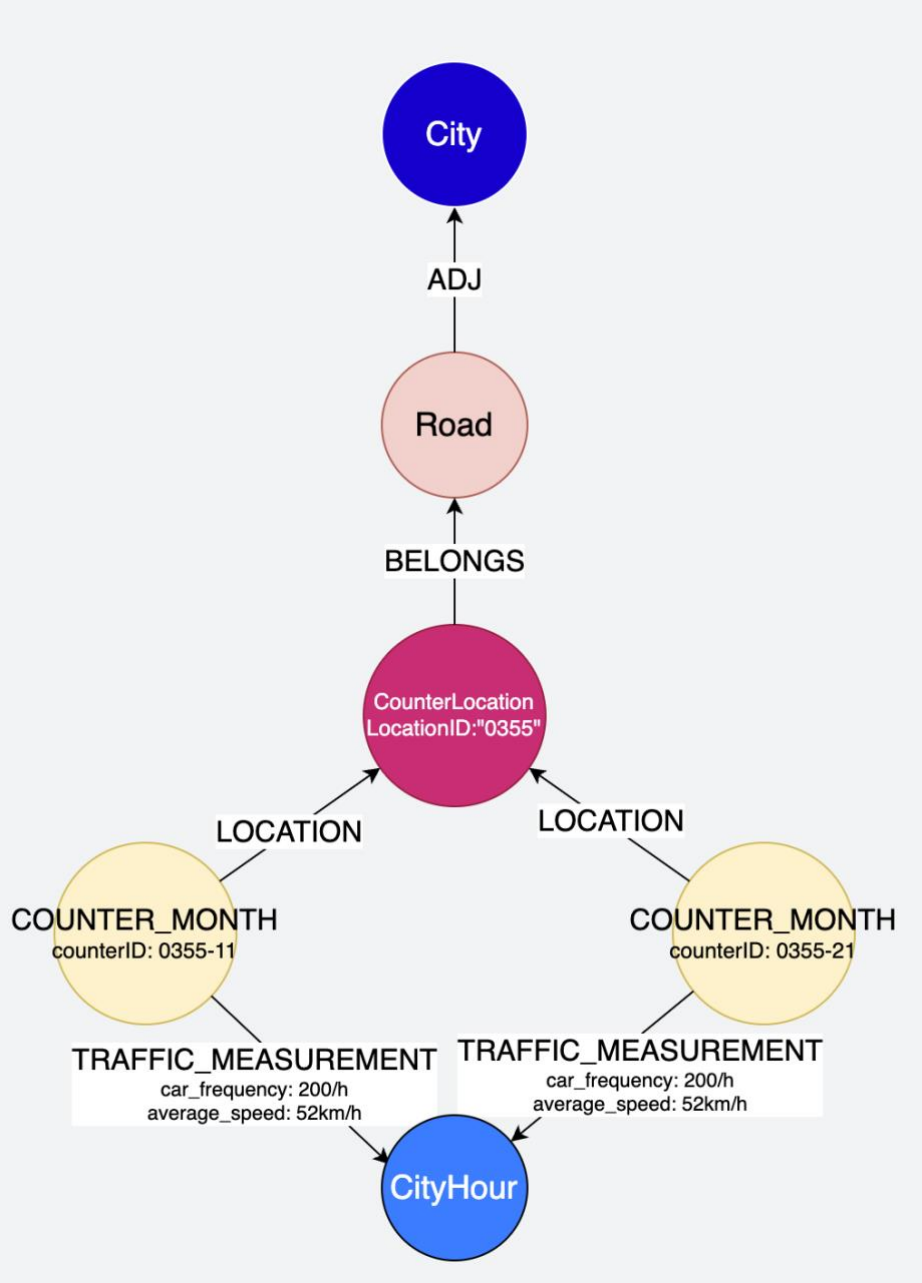

Figure 32: Traffic measurements data structures as stored in the Context Graph

Traffic events are stored as hyperedges (i.e. hyperedge `EVENT`) linking the road segment (i.e. node `Road`) to the `CityHour` contexts that indicate the start and end of the event (Figure 33). Because Neo4j does not support hyperedges, the hyperedge `EVENT` is physically stored as a node.

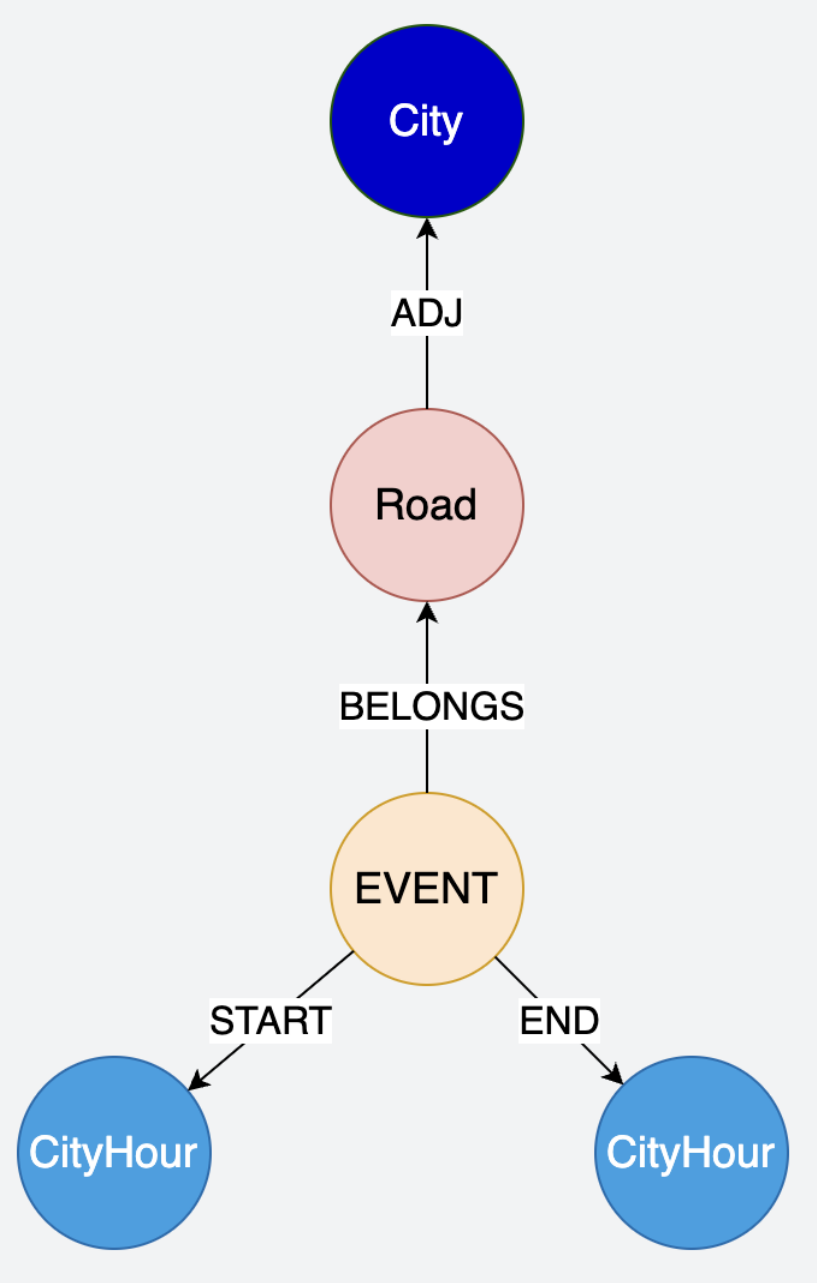

Figure 33: Traffic events data structures as stored in the Context Graph

Weather measurements and weather forecasts are linked to both `CityDay` and `CityHour` contexts. Like with traffic events, to avoid supernode issues, the node Weather was hierarchically split into `WeatherMonth` nodes. Weather measurements are stored on the edge WEATHER_MEASUREMENT linking node `WeatherMonth` to `CityHour` and `CityDay` contexts.

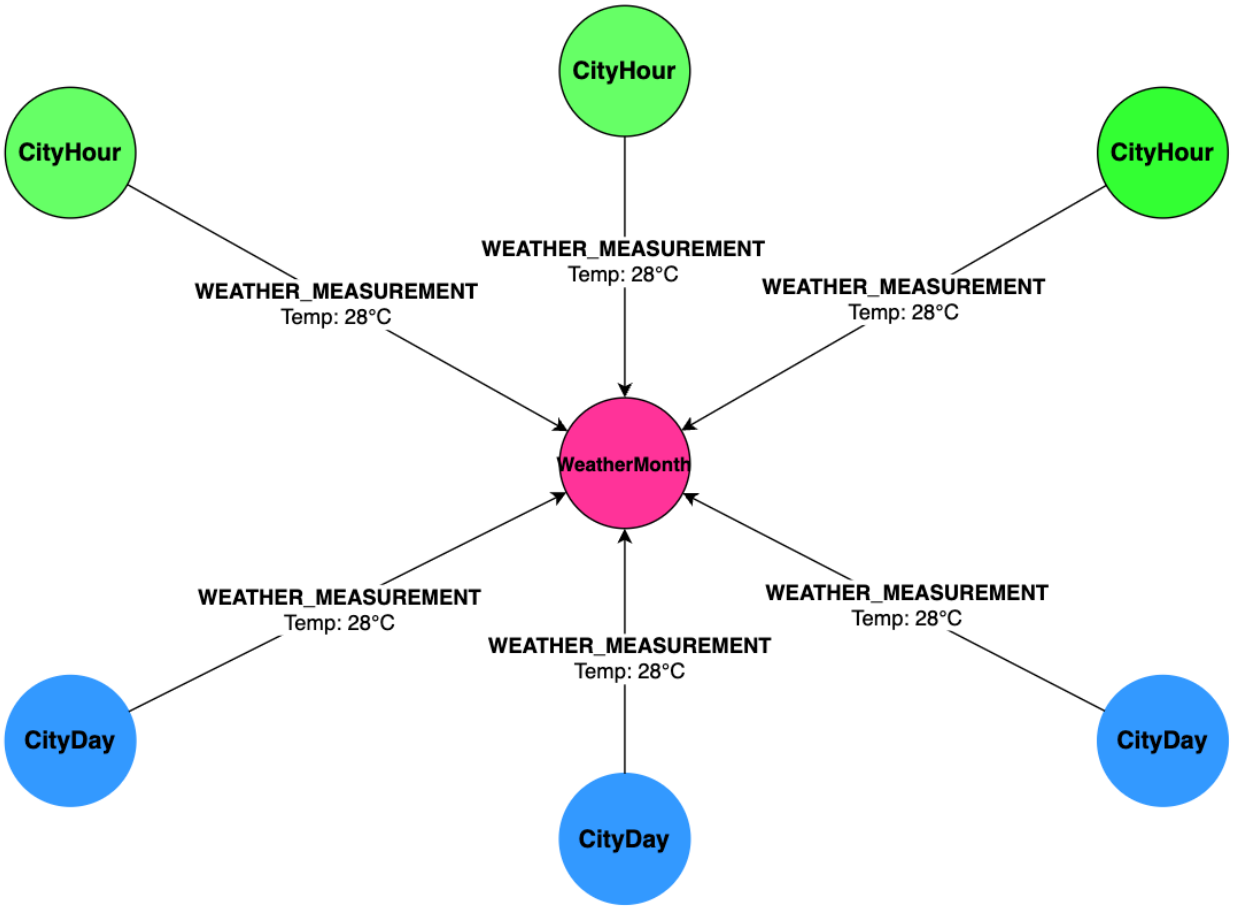

Figure 34: Weather measurement data structures as stored in the Context Graph

Weather forecasts require additional structure that would store the forecasts for 2 weeks in advance for each day. Thus, a weather forecast is a hyperedge called `FORECAST` linking `WeatherMonth` to two `CityDay` contexts (Figure 34). The first context represents the publication day of the forecast and is connected via the `BASIS` edge. The second context is the target day for which the forecast is made and is linked using edge `FORECAST_MEASUREMENT`. For each base day, we store 14 forecasts for two weeks into the future. The design of the weather forecasts is shown in the Figure 35 below. Weather forecasts link `WeatherMonth` to two contexts. The first is the publication day and is linked using edge `BASIS` while the second is the target day and is linked via edge `FORECAST_MEASUREMENT`.

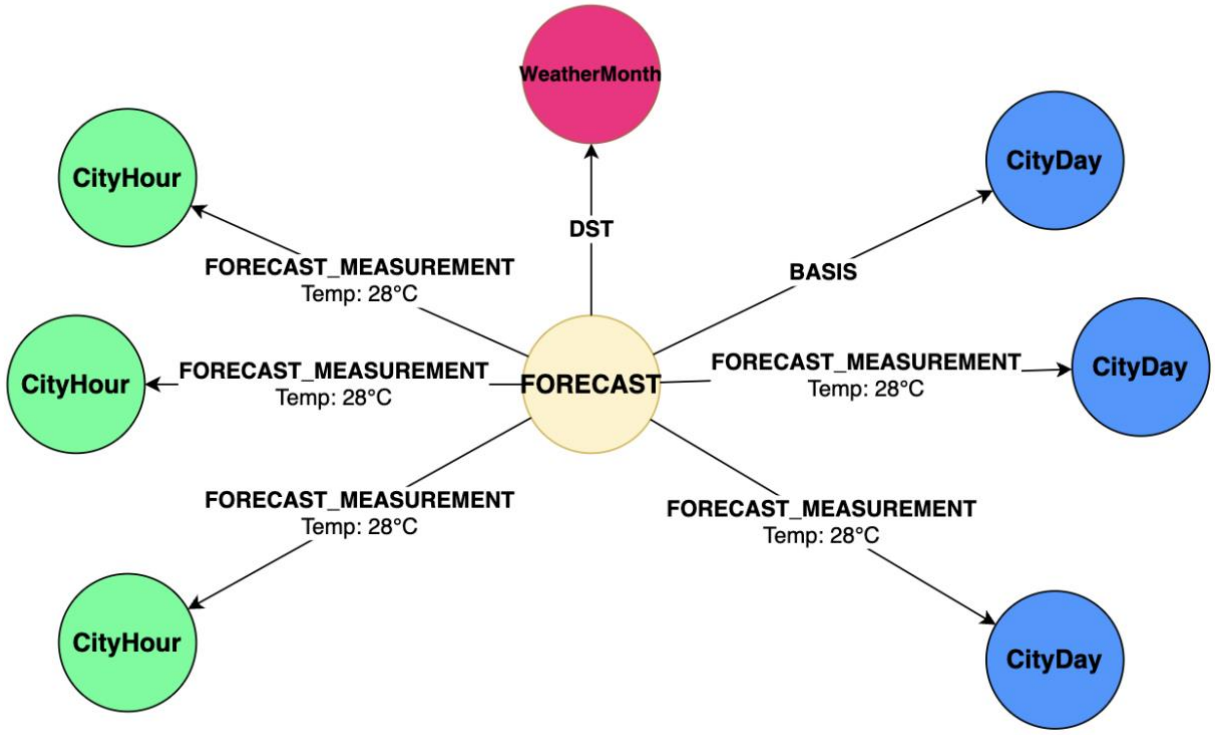

Figure 35: Weather forecasts data structures as stored in the Context Graph

As we have the data structures in place and the data ingestion is running continuously, we used the Context Graph architecture to create the API for accessing Slovenian traffic data and Slovenian traffic events data, which is available to the CONDUCTOR partners. The main use of this Context Graph API is in UC2 for the Traffic Events Assessment Services.

2.8.4 Practical Considerations

While graph databases offer expressive modelling capabilities and flexible edge traversal, their use as a primary store for time series measurements introduces significant practical limitations.

First, graph databases are not optimized for high-throughput, append-only operations typical of time series ingestions. Inserting large volumes of measurements leads to performance bottlenecks and required query optimization and tweaking when the system scales. This is especially true when storing measurements on edges, which require locking both adjacent nodes when performing an insertion or update, causing significant bottlenecks.

Second, querying time series data in a graph database is less efficient than in columnar or time series databases. Operations like time-window aggregations, downsampling, or rolling statistics require complex traversal logic, which increases latency, and in some cases makes the query impractical.

Graph storage also incurs higher memory and storage overhead to explicitly store nodes, edges, and their metadata. For dense measurements, this overhead scales poorly and can become prohibitive.

For these reasons, we find the use of a graph database to store time series measurements impractical and suggest using a columnar database instead. As for the graph database, the best use case would be rich semantic data saved in entities. This would enable having a big context of data. From there, it would be possible to build a classifier for traffic management and future traffic forecasting.

3. NETWORK LOAD BALANCING

3.1 Traffic management solution for signal vehicle couple control

A crucial aspect of load-balancing optimization in transportation networks is the efficient distribution of routes for Connected and Autonomous Vehicles (CAVs). Leveraging advanced sensing and communication technologies, CAVs can dynamically adjust their routes in real time based on prevailing traffic conditions. This capability allows them to redistribute across multiple pathways, mitigating bottlenecks and alleviating congestion.

When integrated with machine learning algorithms for predictive traffic management, dynamic routing optimization enables a proactive approach to load balancing by anticipating and circumventing potential bottlenecks. Furthermore, cooperative communication among CAVs facilitates coordinated route planning, ensuring smoother traffic flow and reducing congestion. Adaptive interaction with traffic signal systems further enhances efficiency by optimizing signal control, regulating flow, and minimizing delays at critical intersections.

Beyond improving overall network performance, environmentally conscious route allocation in CAVs contributes to sustainability by reducing emissions and fuel consumption. By optimizing travel routes and minimizing idle time, CAVs foster a more eco-friendly and efficient urban mobility ecosystem.

This section explores advancements in traffic management strategies involving Signal-Vehicle Cooperative Control (SVCC) and CAVs for effective network load balancing.

3.1.1 Proposed innovation

To address this challenge, we evaluated the following hypotheses: Multiple Traffic Management Centres (TMCs) can oversee a fleet of fully CAVsthrough a hierarchical control framework. Additionally, we assume that each origin-destination (OD) pair is linked by one or more predefined routes, with the relevant TMC possessing complete knowledge of each CAV's origin and destination.

In this study, we examined both fully autonomous and mixed traffic conditions. Our objective is to optimize the traffic control strategy for key arterial roads within the network while simultaneously determining the optimal path assignment for the entire CAV fleet. Specifically, this involves optimizing the proportion of CAVs assigned to each OD pair and corresponding route to enhance overall network efficiency.

3.1.2 Specification

This section presents a set of principles designed for the management and routing of CAV fleets. The proposed approach is fundamentally based on advanced sensitivity analysis, leveraging simulations to evaluate various scenarios and their impact on the network at different CAV penetration rates. To ensure that overall network demand accurately reflects real-world traffic patterns, the simulation model is meticulously developed to align with well-calibrated existing models.

The simulation itself is a sophisticated system, employing mesoscopic models to comprehensively analyse traffic conditions on the M30 corridor, in Madrid, over an extensive study area. This entails incorporating realistic connectivity assumptions, accurately detecting CAVs within the simulation, integrating traffic signal control plans, and providing detailed route guidance. Notably, the

methodology combines decentralized control for individual CAVs with hierarchical traffic management stations that oversee the broader CAV deployment.

Additionally, the strategy includes an extensive feasibility study, along with the development of key performance indicators to assess the impact of CAVs, rerouting strategies, and external events. This holistic approach ensures a thorough evaluation of the proposed system's effectiveness in optimizing traffic control and safety, both within the city of Madrid and in larger urban environments.

3.1.3 Progress of the work performed

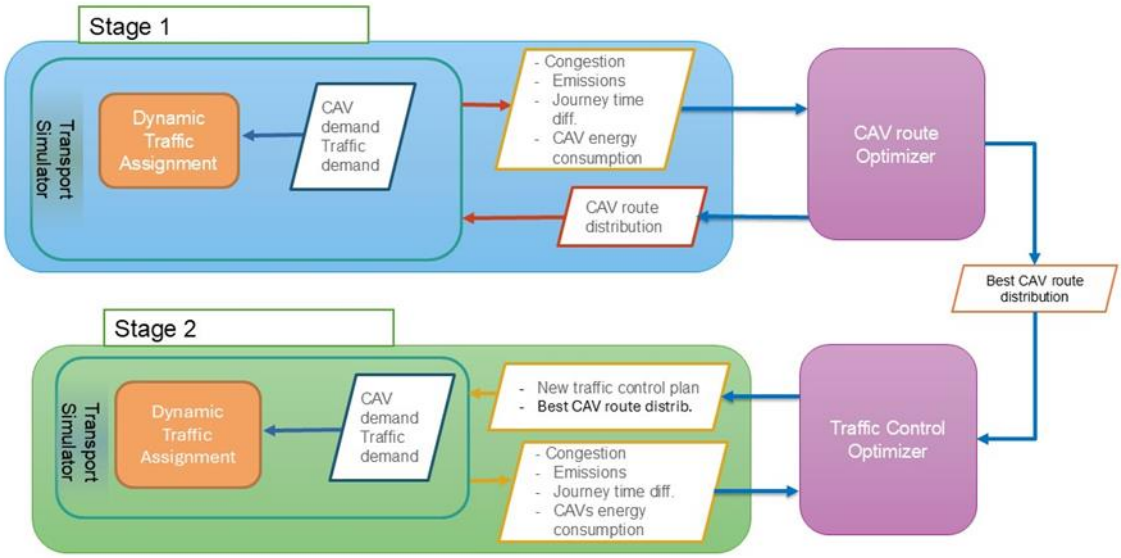

Figure 36: Schematic diagram for sequential SVCC optimization

An illustration of Signalized Vehicle-Coupled Control (SVCC) in operation is presented in Figure 36. For each predefined OD route, the SVCC scheme optimizer proposes a refined traffic control strategy for a selected set of intersections and corresponding CAV distribution percentages. The SVCC scheme follows a sequential optimization approach, first determining the optimal percentage of CAVs assigned to each OD route. Using this information, the model then identifies the most effective traffic control strategy.

The SVCC OD route percentage optimizer derives its results by analysing key performance metrics, including congestion levels, CAV energy consumption, overall emissions, and travel times for both CAVs and conventional vehicles. In each iteration, updated CAV distribution percentages for each OD route are generated based on these metrics.

In Stage 1, the Dynamic Traffic Assignment (DTA) module calculates Key Performance Indicators (KPIs) such as congestion, emissions, the travel time disparity between CAVs and conventional vehicles, and CAV energy consumption. These KPIs are then fed into the traffic control optimizer module, in stage 2, which develops an updated traffic control plan. The revised plan is subsequently incorporated into the transport simulation program, generating a new model that optimizes intersection phase timings and CAV route distributions.

The primary objective of this optimisation process is to reduce congestion, minimise emissions, decrease the travel time gap between CAVs and conventional vehicles, and enhance CAV energy efficiency, thereby improving overall network performance.

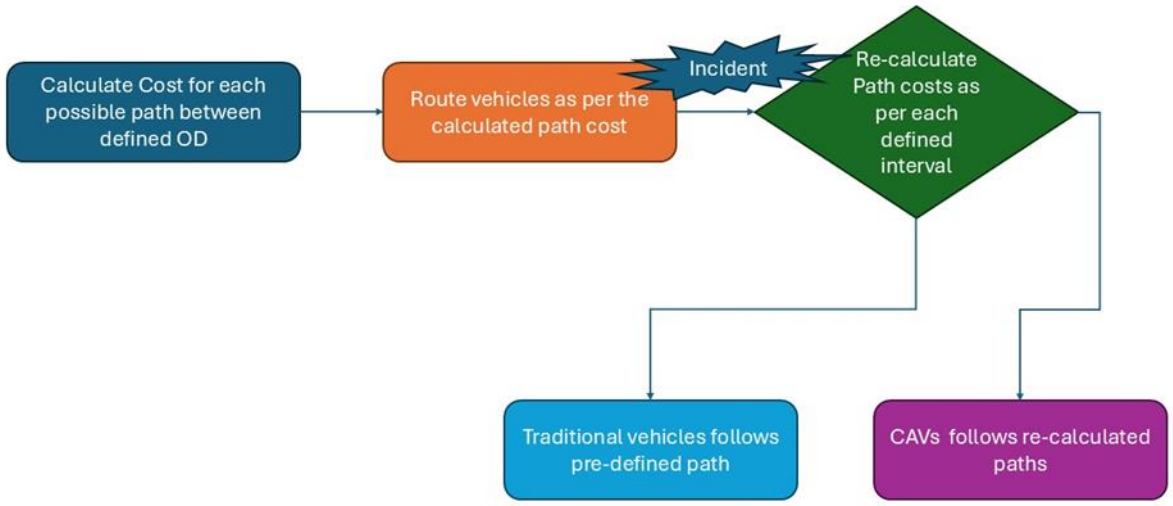

Figure 37: Flowchart for dynamic incident management with CAVs

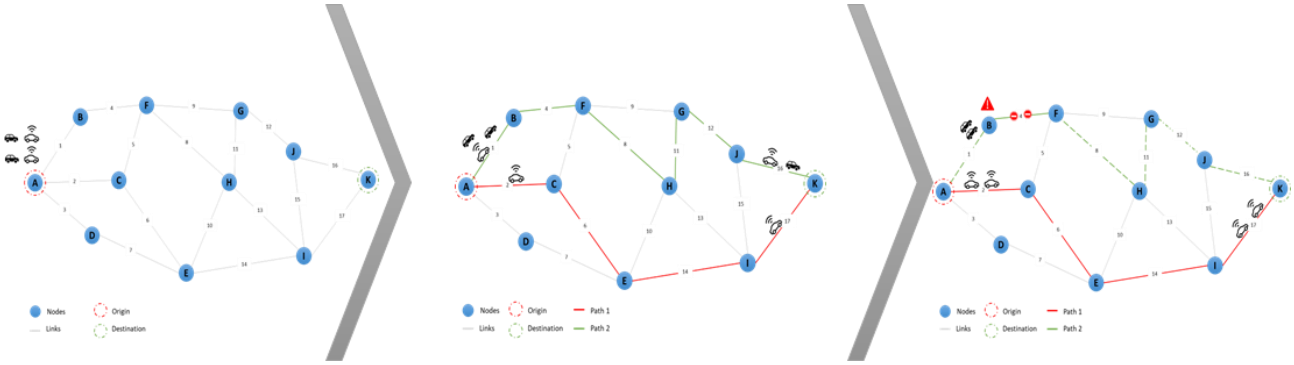

Figure 38: Dynamic re-routing of CAVs based on current status of network

Another key functionality developed within this module is the dynamic rerouting of CAVs based on a decentralised control scheme. Once the initial CAV route assignment and linked signal optimisation phase are completed, the dynamic rerouting mechanism is activated. In this phase, CAVs on predefined critical routes are dynamically reassigned based on path costs, which are recalculated at predefined simulation intervals. A flowchart illustrating this process is presented in Figure 37.

To further elaborate on this scheme, consider the scenario depicted in Figure 38. The leftmost figure represents the initial state, where node *A* serves as the origin and node *K* as the destination. At the start of the trip, the costs for all available paths between *A* and *K* are computed. As shown in the second figure, two primary routes are selected: a red path and a green path. The CAV demand is distributed across both routes, while conventional vehicles follow the green path as per the fixed path scheme.

In the rightmost figure, an incident occurs between nodes B and *F*, causing a complete blockage of the green path. This disruption triggers the rerouting mechanism, prompting the recalculation of path costs. Since CAVs are managed by the TMC, they are immediately redirected to the remaining viable route (red path). However, conventional vehicles, which lack centralized coordination, remain stuck

at node B due to the incident. This dynamic rerouting capability allows CAVs to reach their destinations efficiently, in contrast to conventional vehicles that experience delays due to network disruptions.

By implementing this decentralised control scheme, the overall throughput of the transport network can be enhanced while also improving the system's resilience against unforeseen incidents and disruptions. In this research direction, we have tested the centralised CAV routing and signal optimisation. Decentralised incident management using CAV routing has been successfully tested along the main route and signal optimisation process. Currently, we are trying to adapt the module for UC1 Madrid's M30 scenario.

3.2 Social routing with multimodal perspective

Deliverable D3.2 presents a multi-modal social rerouting model, that is a travel demand management measure where a portion of travellers is asked to make individual sub-optimal yet acceptable travel choices for the improvement of travel conditions of the population (social re-routing). This measure aims to bridge the gap between user equilibrium and system optimum, by re-distributing traffic in a way that is beneficial for the system and does not significantly deteriorate the travel conditions for individuals.

The literature review in D3.2 revealed that travellers may not exhibit rational decision making, e.g., by using a fast but not necessarily shortest path between their origin and destination. In literature, this is known as 'boundedly rational choice behaviour'. This characteristic can be used exploited for the development of network loading balancing techniques, since system performance can be improved by slightly degrading the level of service for a group of users with eliciting a change in travel behaviour, and thereby the objectives of the system and of users can be balanced. In fact, some travellers may not notice these deviations, or may not act on it, and as such can be labelled acceptable (see Vreeswijk et. al., 2013). However, the margins of such thresholds are fuzzy, i.e., the indifference band is person- and situation-specific. In deliverable D3.2, a network load balancing method was presented, where passengers received advice to use specific routes to steer the network towards a system optimum. This is a difficult task in general since the travel conditions, and thereby the compliance with advice depends on the response of individuals to the advice, but the re-routing behaviour of receptive travellers may also elicit responses from travellers that are not targeted or reached by travel advice but may adapt their travel choices in response to the changing behaviour of others (Eikenbroek et. al., 2022; Szep et. al., 2023). As such, these responses need to be anticipated to prevent unintended effects. In a public transit setting, additional challenges appear since the timetable and limited vehicle capacity may limit the potential detours that can be offered. In this setting, the social rerouting framework presented in D3.2 adopts a bilevel framework, where network-wide feedback effects are anticipated through a novel passenger equilibrium assignment, through which passenger responses are predicted.

The hierarchical framework can be used to formulate a mixed-integer program, implicitly finding the best-possible advice to receptive travellers while respecting the indifference band and improving efficiency in terms of total generalized travel time, accounting for both travel time and crowding levels in relation to the limited capacity and the possibility that it may be impossible to board a vehicle if capacity is reached. The potential of social rerouting in public transport was assessed by performing numerical experiments using data from the Twente public transport network, revealing that if social travellers only accept detours of 5 or 10%, the maximum possible improvement in network efficiency is 0.6 and 0.9%, respectively (with a mimic efficiency improvement of 1.1% in system optimum).

The initial version of the model considered a deterministic setting, i.e., without uncertainty. In the remainder of this section, we discuss the possibility to account for various uncertainties that appear when advising routes for the benefit of the system. Accounting for emerging information is important, since demand and supply are inherently variable (e.g., delays and disruptions), and travellers react to and anticipate changing levels of information, e.g., through a travel planner or re-routing based on previous experiences. When modelling travel behaviour under uncertainty, (Eikenbroek et. al., 2022) identified two basic models. Travellers can be assumed to be non-adaptive, i.e., choosing a single habitual route independent of the occurring scenario, or travellers exhibit (fully) adaptive behaviour, using a travel planner to adapt their behaviour accordingly. In a stochastic network load balancing setting, the question is how to design social travel advice while accounting for the uncertainties that may appear and the corresponding responses in travel behaviour.

We introduce notations from D3.2. Given is a directed traffic network $G = (V, E)$, with V being the set of nodes, and E being the set of directed edges (road, links, or arcs), $e = (i, j)$, with $i, j \in V$. There is a set of OD pairs $w = (t_w, s_w) \in V \times V$. Each OD-pair $w \in \mathcal{W}$ has a corresponding demand $d_w > 0$, and is connected by a set of simple directed paths or routes $\mathcal{R}_{\mathfrak{k}}$. The set of $\mathcal{R}$ of all paths in the network is the union of the path sets per OD-pair, i.e., $\mathcal{R} = \bigcup_{w \in W} \mathcal{R}_{\mathcal{w}}$. A distribution of the demand d is a pair of flow vectors $(f, x) = (f_r, r \in \mathcal{R}; x_e, e \in E)$ so that $\Lambda f = d, \Delta f - x = 0, f \geq 0$, where $\Lambda \in \mathbb{R}^{|\mathcal{W}| \times |\mathcal{R}|}$ is the OD-path incidence matrix, with $\Lambda_{\mathrm{wr}} = 1$ if route r is in $\mathcal{R}_{\mathcal{w}}$, and $\Lambda_{wr} = 0$ otherwise. $\Delta \in \mathbb{R}^{|\mathrm{E}| \times |\mathcal{R}|}$ is the arc-path incidence matrix, defined by $\Delta_{\mathrm{er}} = 1$ if link e is in route r, and $\Delta_{er} = 0$ otherwise. Each arc e in the network has a corresponding (link) flow-dependent, separable non-negative, continuous, convex and strictly monotone disutility or travel cost function $l_e: \mathbb{R}_+ \to \mathbb{R}_+$. The route cost $c_r(f)$, induced by traffic flow (f, x), is the sum of travel costs of all edges constituting that path: $c_r(f) = \sum_{e \in r} l_e(x_e)$. The set of feasible flow distributions is denoted by $\mathcal{F}$.

Let us assume that the uncertainty appears in (i) the portion of travellers that can be nudged towards individually sub-optimal and system-beneficial routes and (ii) the travel conditions, and more specifically the travel times. In our setting this means that an a-priori unknown share of travellers is willing to act for the benefit of the system, and only if the expected a-posteriori travel time on the suggested path is at most $\varepsilon > 0$ larger than travel time on the shortest path. We consider probability space $(\Omega, \mathcal{F}, P)$, with both travel times and portion of *receptive* demand dependent on the elements ω ('scenarios') of Ω, and we assume Ω to be a finite set, with mass $P: \Omega \to [0,1]$, so that $\sum_{\omega \in \Omega} P(\omega) = 1\$$, and $P(\omega) > 0$. Note that here we do not make any assumptions regarding independent of travel times. Since supply and demand are uncertain, we model the route and link flows to be we dependent on the scenario, that is $(f(\cdot), x(\cdot))$ is a pair of ω-dependent route and link flows $\big(f(\omega), x(\omega)\big) \in \mathbb{R}^{|\mathcal{R}|} \times \mathbb{R}^{|E|}$. As a result, route costs and link flows also become scenario-dependent.

Considering (fully) adaptive behaviour, $\left(\bar{f}(\cdot), \bar{x}(\cdot)\right)$ with $\left(\bar{f}(\omega), \bar{x}(\omega)\right) \in \mathcal{F}$ for each $\omega \backslash in \backslash Omega$ is in equilibrium if the following condition holds for all $w \in \mathcal{W}, r \in \mathcal{R}$, the following condition hold:

$$\text{for any r,q} \in \mathcal{R}_{\mathcal{w}}\ \bar{f}_r(\omega) > 0 \Rightarrow c_r\big(\bar{f}(\omega), \omega\big) \begin{cases} \leq c_q\big(\bar{f}(\omega), \omega\big)\ if\ \bar{f}_q(\omega) = 0 \\ = c_r\big(\bar{f}(\omega), \omega\big)\ if\ \bar{f}_q(\omega) > 0 \end{cases} \quad (1)$$

In case of non-adaptive behaviour, $\left(\bar{f}(\cdot), \bar{x}(\cdot)\right)$ with $\left(\bar{f}(\omega), \bar{x}(\omega)\right) \in \mathcal{F}$ for each $\omega \backslash in \backslash Omega$ is in equilibrium if the following condition holds for all $w \in \mathcal{W}, r \in \mathcal{R}$, the following conditions hold

$$\text{for any r,q} \in \mathcal{R}_{\mathcal{w}}\ \bar{f}_r(\omega) > 0 \Rightarrow c_r\big(\bar{f}(\omega), \omega\big) + \quad (2)$$
$$\lambda_r(\omega) \begin{cases} \leq c_q\big(\bar{f}(\omega), \omega\big) + \lambda_q(\omega)\ if\ \bar{f}_q(\omega) = 0 \\ = c_r\big(\bar{f}(\omega), \omega\big) + \lambda_q(\omega)\ if\ \bar{f}_q(\omega) > 0 \end{cases}$$

with additionally $\bar{f}(\omega^0) = \bar{f}(\omega^1)$ for any $\omega^0, \omega^1 \in \Omega$, and $\lambda(\cdot)$ so that $E_\omega\{\lambda(\cdot)\} = 0$. Intuitively, these conditions say that traffic is distributed such that flow is assigned to a path that is not the shortest in a scenario, but is the shortest *on average,* i.e., in expectation.

We study the effects of uncertainty on social rerouting, assuming the system goal is to minimize expected total travel time. In this case, all travellers receive route advice, which is in expectation at most $\varepsilon > 0$ worse than the expected shortest path (see Eq. 2). The acceptance rate $p(\cdot)$ is unknown beforehand, and therefore scenario-dependent, with a decline rate of $1 - p(\omega)$ in case of scenario ω. Those that do not comply with the social travel advice, travel along the shortest path (see Eq. 1). The challenge is to have acceptable expected travel times such that receptive travelers comply with social travel advice, anticipating that a random portion also rejects advice. In this setting with separable and increasing travel time functions, this problem is a mathematical program with equilibrium constraint that reduces to a bilevel optimization problem.

To illustrate the relevance of considering uncertainties in the context of network load balancing we consider the following adapted version of Braess' paradox, with the travel time and demand provided in Figure 39 below. When traffic is in Wardrop equilibrium, demand is distributed over the three routes such that travel times are equal (85.38). In system optimum, the vertical link is not used. Total travel time in system optimum is approximately 388, while in Wardrop equilibrium total travel time is 427.

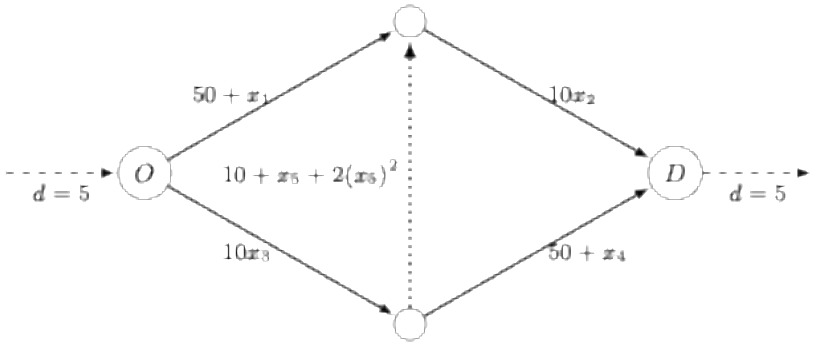

Figure 39: Adapted Braess paradox network

We expand the social rerouting model of D3.2 and consider the effect and potential of social rerouting in a stochastic setting. We assume that there are two scenarios i.e., $\Omega = \{\omega^1, \omega^2\}$ with ω-dependent acceptance rate

$$p(\cdot) = \begin{cases} p(\omega^1) = 1/5 \; with \; probability \; 1/4 \\ p(\omega^2) = 4/5 \; with \; probability \; 3/4 \end{cases}.$$

Here, to comply with the user-induced constraints regarding the maximum-possible detour, the different scenarios have to be anticipated. In Figure 2, we display the expected total travel time as function of the maximum detour accepted by those travellers that comply with the advice. For comparison, we also display the expected total travel assuming all travellers are receptive for advice, and if 65% of the travellers are receptive for advice (blue), which equals the expected acceptance rate. Figure 40 reveals that acceptance rate uncertainty provides opportunities for social routing since incidental longer travel times are accepted. For example, with $\varepsilon = 6$, in scenario ω^1 travel times on all used routes are equal, while in ω^2 the travel time on routes 1 and 2 are 82.42, while route 3 has a travel time of 74.42. That is, the expected detour is 6, but an incidental detour of 8 is accepted.

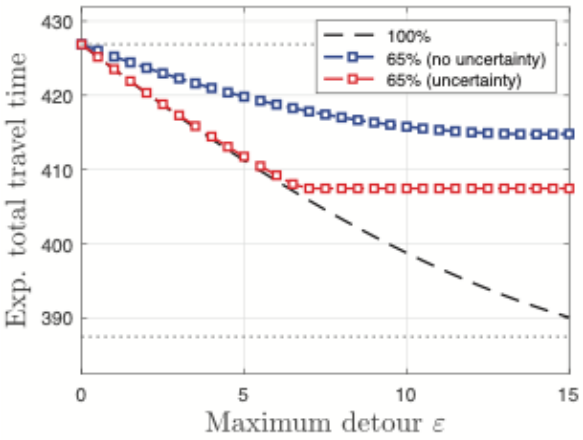

Figure 40: Acceptance rate uncertainty for social routing

Social rerouting persuades some travellers to choose routes for the benefit of the system. In the multimodal setting of D3.2, the social rerouting framework is applied and tested in the public transport network of Twente, illustrating that 25% of the maximum improvement in efficiency can be obtained with 20% of travellers willing to act socially. We considered a stochastic extension of this model, assuming the portion of travellers willing to act socially is random following an exogenous empirical discrete distribution. This leads to additional challenges since travel times depend on route flows, and thus also become random. Consequently, travel advice should be adopted accordingly, illustrated to provide opportunities to reach policy objectives, e.g. since incidental longer routes may be accepted as long as one is compensated by suggesting faster or even individually optimal routes in other cases.

3.3 Prediction Models for Demand Responsive Transport

As part of the framework for demand responsive transport a forecasting model is required that can predict future demand for shuttle transport between network of GoOpti partners on routes Slovenia – Italy, Slovenia – Croatia, and Slovenia – Austria. The idea is to provide the "observability" of demand prediction to feed the optimisation algorithms for traffic routing and fleet operation. This forecasting component should be able to predict the number of passengers on a given route at a given time in hourly resolution up to one year into the future. Final specifications are, therefore, aligned with the ones detailed in D3.2:

- Predictions are limited to passengers' that have a drop-off during specific time slots at designated locations (defined by predetermined routes explained above).
- The pickup or drop-off needs to be considered as originating from the city if the actual location is located within 30 km of the city centre.
- The predictions need to be made for both directions of the route.
- The prediction resolution will be predefined (1 hour or more).
- Data that can be used for training and testing of the models is limited to GoOpti's own historical data in potential combination with weather, flight, and traffic information.
- Two types of predictions are expected, short term up to 14 days and long term up to 1 year.

There are only two deviations from these specifications from D3.2. The first is that we found there is no need for two separate forecasting models (short-term and long-term) since the results presented

in this section show that a single model can sufficiently capture both of those cases in a holistic manner. The second deviation is the addition of:

- Predictions should include information about its accuracy such as estimation of confidence intervals or quantiles.

3.3.1 Model Architecture

Based on the specifications and through coordination between JSI and the Use Case leader GoOpti we agreed that the demand forecasting model should estimate five different quantiles (q=0.05, 0.25, 0.5, 0.75, 0.95) for each prediction of the number of passengers. This means that a rudimentary distribution of the expected number of passengers is available for the use in the optimization algorithms for traffic routing and fleet operation. The model, therefore, returns five count estimations (5^{th} percentile, first quartile, median, third quartile, and 95^{th} percentile) based on the available features. After comprehensive data analysis and initial modelling, we identified that the most relevant input features are time features for the time of departure and the time of purchase, as well as their difference. These time features include time of the day, time of the week, time of the year and the linear time. Another set of features are the already known number of passengers (reservations made before the purchase time) for time buckets adjacent to the departure time. We found 9 adjacent (hourly) buckets to hold sufficient information about the current state of reservations for a given departure time.

There are some features that were not used in our pipeline. After further considerations we recognized that the weather and the traffic information is very unlikely to be relevant because the current data base, to a very large degree, contains orders related to airport transfers. These orders are made several days ahead of departure time and do not relate to traffic and weather conditions in Slovenia but rather (possibly) to conditions at the flight destinations. On the other hand, flight information should be quite relevant for such modelling, however, we were unable to obtain such flight data that would span from 2014 to today due to high financial costs. In this way the model needs to uncover the relevant flight information from other features. Given the large size of order data made available by GoOpti it is reasonable to expect that, to some degree, the flight schedule could be extracted by the model solely from the distribution of order departure times. But this is only the case if flights are fairly regular (e.g., periodic) and this is an assumption that we needed to accept due to flight data being unavailable.

We tested several different types of model architectures and forecasting schemes. For example, a model can be specialized for a single route (e.g., Ljubljana – ZAG) or it can take this route information (origin and destination) as input features. Likewise, a model can be specialized to predict for one single forecasting horizon (e.g., for 53 hours into the future), span of forecasting horizons (e.g., 14-21 days into the future), can take forecasting horizon as an input feature, or it can return predictions for the entire year in a single sweep. We found that a single holistic model works best. This means that the model takes route information as input features (together with other features) and returns (five different quantiles worth of) predictions for the entire year into the future. We found several reasons that contribute the choice of this type of architecture. Training many models compared to a single one requires more computational resources meaning less intensive optimization. With many models we also have a smaller amount of training data points per model since the full data set is split to many specialized cases. We found that overfitting can, therefore, be more pronounced with many models compared to a single one. This means that more care needs to be taken when training many models. Another reason is that with a single model we are able to take advantage of the knowledge transfer between all the special cases since patterns that are beneficial for one special task can also be beneficial for others which should make training less complicated.

Based on these observations we converged toward an architecture depicted in Figure 41. Because of a very large training data set (amounting to several 100 GB in expanded form) we opted to move feature generation into the model as much as possible as opposed to it being precomputed. By this we were able to reduce the training data size to such a degree that it is able to reside in the RAM which results to much faster training. We achieved this by saving purchase and departure times as integers (current_time_index) and use an embedding layer to attain all time features in a computationally efficient way. Route information was also encoded using integers (origin_index and destination_index) which were passed to an embedding layer that returned a continuous representation of the route information. Data about currently known orders were supplied in node current_counts and a convolutional layer was used to efficiently access adjacent values meaning that we did not need to save several copies of a slightly shifted time series of currently known number of passengers. Time difference between departure and purchase time is generated in node subtract_26 and normalized. All these features are then concatenated and supplied to a computational block composed of several dense layers with dropout. The output of the model is composed of three components. The first is the most important, estimation of positions of five quantiles for the number of passengers (quantile), while the other two were found to be beneficial for training. Because the data includes large proportion of time buckets without any passengers, we found that estimating the probability that at least one passenger is in a time bucket (classification), in addition to the actual count, helps the model to more efficiently learn when the counts should be exactly zero. The last output of the model (current_count) estimates the currently known number of passengers. This helps the model to not lose too much information about the currently known number of passengers that can happen when information flows through several layers of computation. These two additions accelerated the model training and improved its accuracy by about 5%.

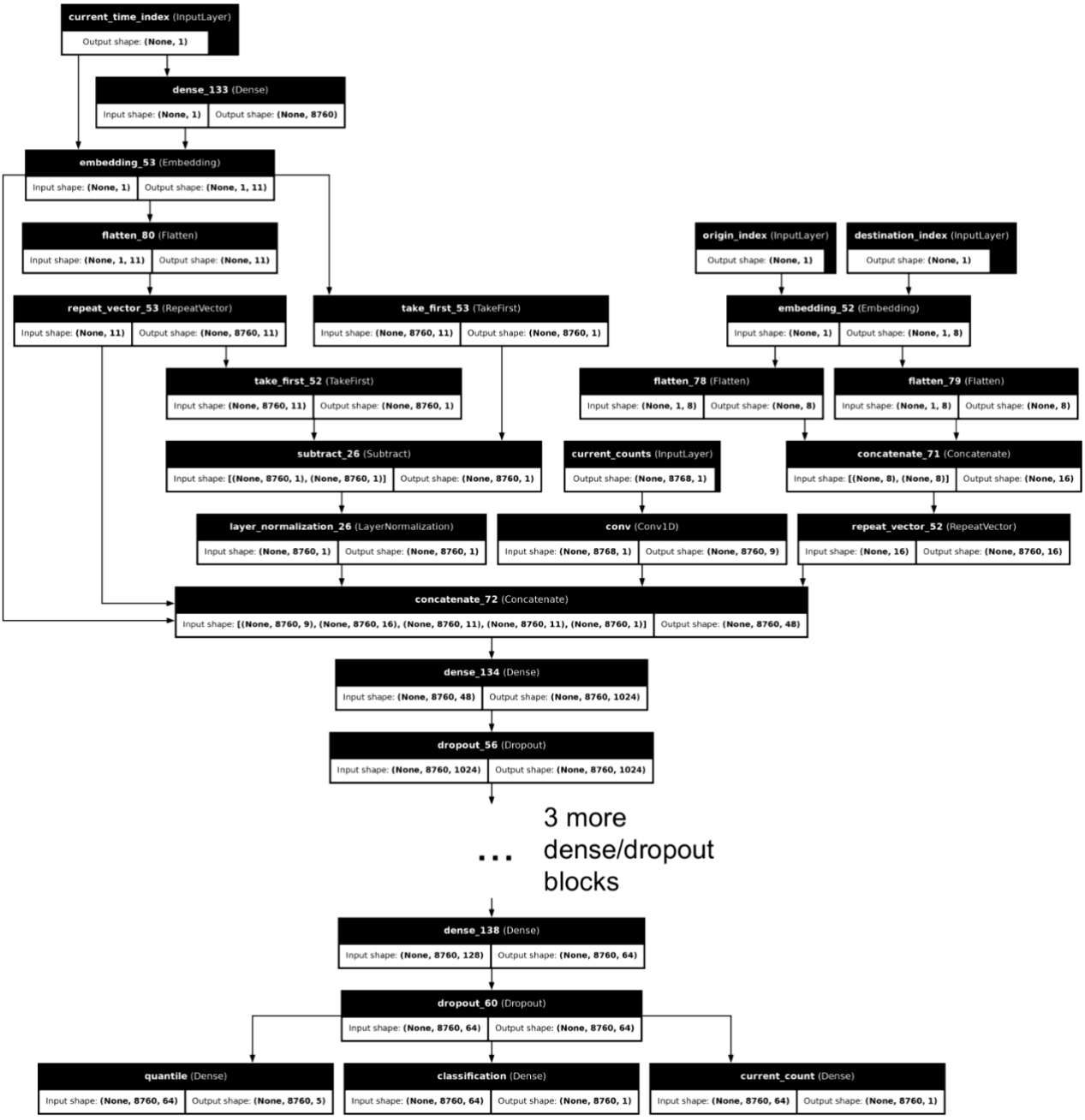

Figure 41: Model architecture.

3.3.2 Model Training and Hyperparameter Optimization

To train the model we require a metric that quantifies to what degree the output of our model is misaligned compared to ideal outputs in our data base. This quantity, called error, is minimised during training using optimisation. For current_count we used mean absolute error, for classification we used binary cross-entropy, and for quantile we used sum of five pinball loss functions, one for each quantile estimation. The training was performed using gradient descent algorithm (Adam and RMSProp). Dropout and early stopping were used to prevent overfitting. A characteristic number of epochs performed during training was around 10.

The loose architecture depicted in Figure 41 still has many tuneable hyperparameters whose values can significantly affect the accuracy of the model. To explore the impact of this we performed a hyperparameter optimisation by minimising validation error of the model using grid search. We

considered several combinations of hyperparameters including widths and depth of the dense block of the model, training algorithm, intensity of dropout, and the size of positional encoding. Figure 42 depicts how the validation error changes with respect to different hyperparameter values. The winning model architecture is depicted in Figure 42.

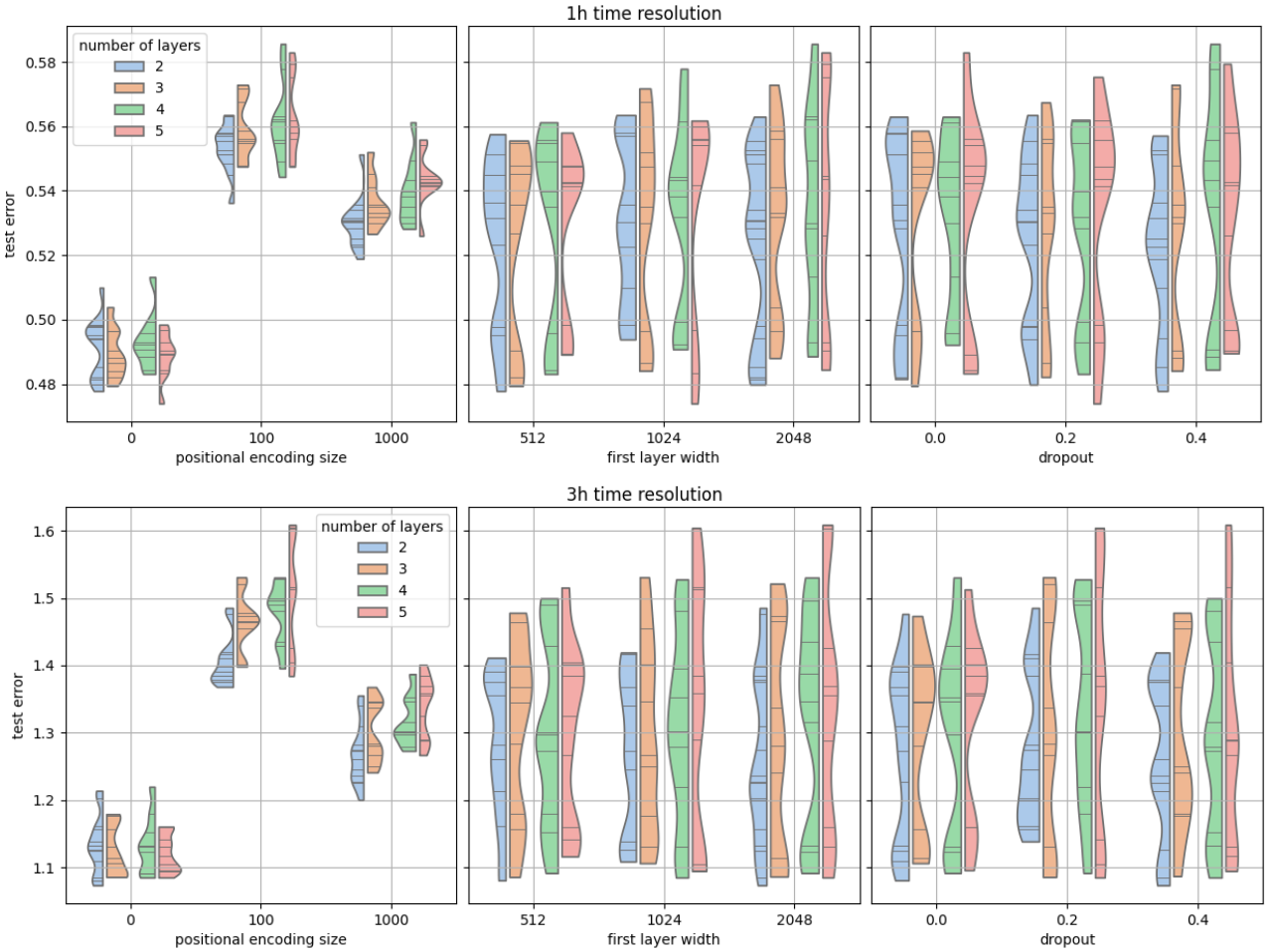

Figure 42: Final model architecture with tunable hyperparameters.

3.3.3 Validation

The accuracy of the model was validated on a test data set comprising of orders between 1st of January 2024 and 31st of December 2024, while the model was trained on data from 1st of May 2014 to 31st of December 2023. Two examples of predictions with our model are shown in Figure 43. For shorter forecasting horizons (e.g., up to 1 month into the future) our model predictions are very well aligned with the true number of passengers. Also, the estimations of quantiles are not spread far from the real values indicating that the model is quite confident that its prediction is close to the actual value. For larger forecasting horizons (e.g., 3 months into the future or longer) we see that the model is less certain about its predictions and the prediction is much smoother compared to the actual values. We attribute this to the fact that the model has much less information since there are not as many reservations known yet for so far away into the future. So, the model is unsure about in which specific hours the spikes in demand will appear, unlike in the case of shorter-term predictions where some known orders already allow the model to gauge positions of these spikes in demand.

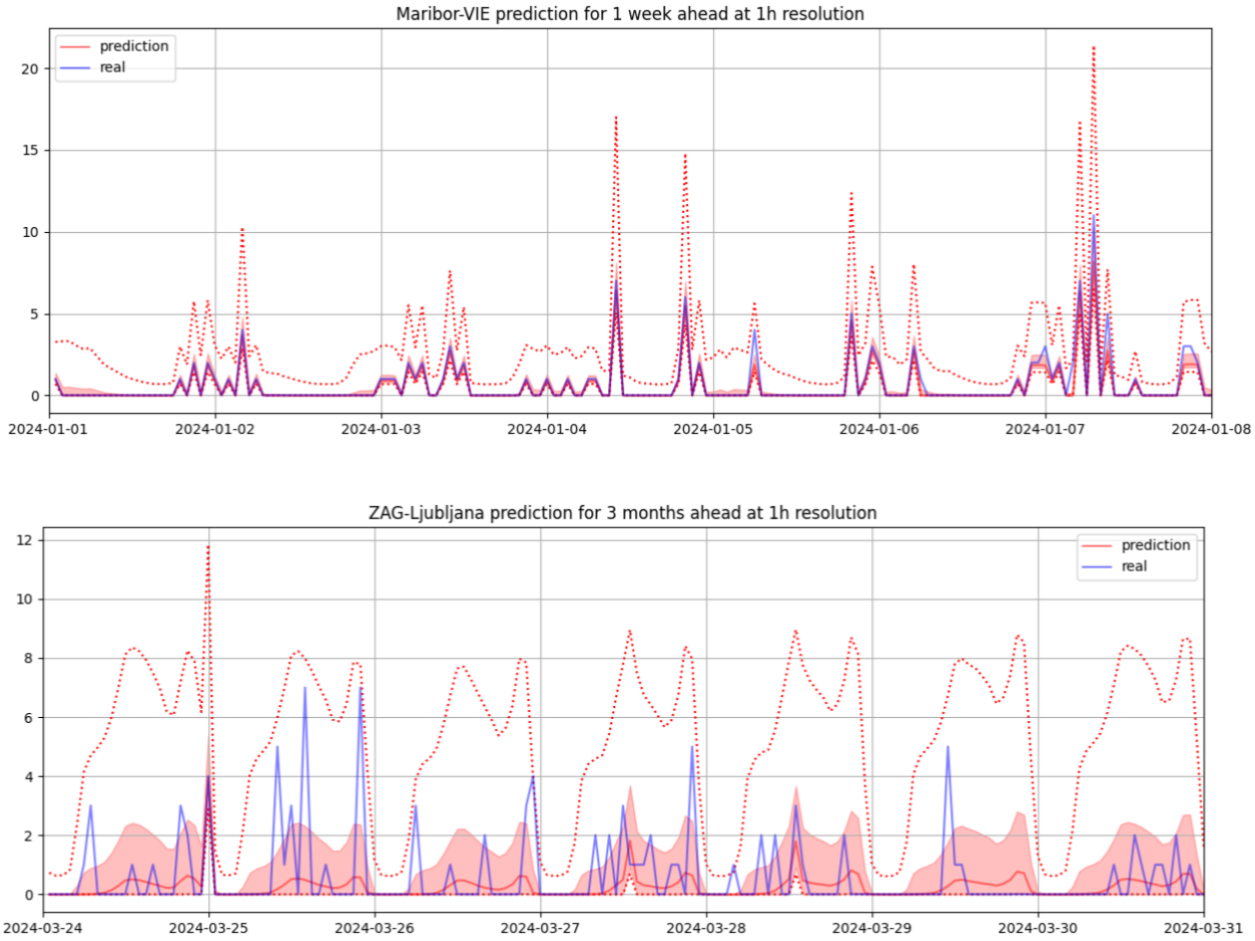

Figure 43: Examples of prediction results.

This means that the model's accuracy decreases with forecasting horizon. To further analyse this phenomenon, we require metrics to assess the model's accuracy. The first metric we use is mean absolute error (MAE) which measures the difference between median estimation of our model (q=0.5) and real number of passengers in a given time bucket. The second metric is mean absolute percentage error (MAPE). Because large proportion of time buckets in our data has zero number of passengers, we used a smoothened version of MAPE in order to avoid dividing with zero. For a smoothing time scale, we used 1 week. This means that MAPE is calculated by finding MAE in the time span of 1 week and dividing this value with total number of passengers in that week. This metric gives a different perspective of the model error compared to MAE because there are seasonal variations and trends in the number of passengers through time. MAE is, therefore, more appropriate for measuring accuracy on larger time scales while MAPE can also be used on smaller time scales. Since our model is the first system for forecasting demand in this scope and for this data, we will compare it to a baseline model that is simply a mean value of passengers in a time bucket. Figure 44 and Figure 45 report how our metrics depend on forecasting horizon for different routes and time bucket resolutions. We see that MAE of our model is substantially smaller than the MAE of the baseline until we get to forecasting horizons of approximately 1 month. With MAPE we see a similar picture, however, due to smaller averaging windows we see more variance in the results. But we can see that, for large forecasting horizons, MAPE for a model with 1h resolution is somewhere between 90% and 100% error while the model with 3h resolution is systematically better with error between 60% and 90%. This means that finer time resolution is more appropriate for short-term forecasting while a coarser time resolution works better for long-term forecasting.

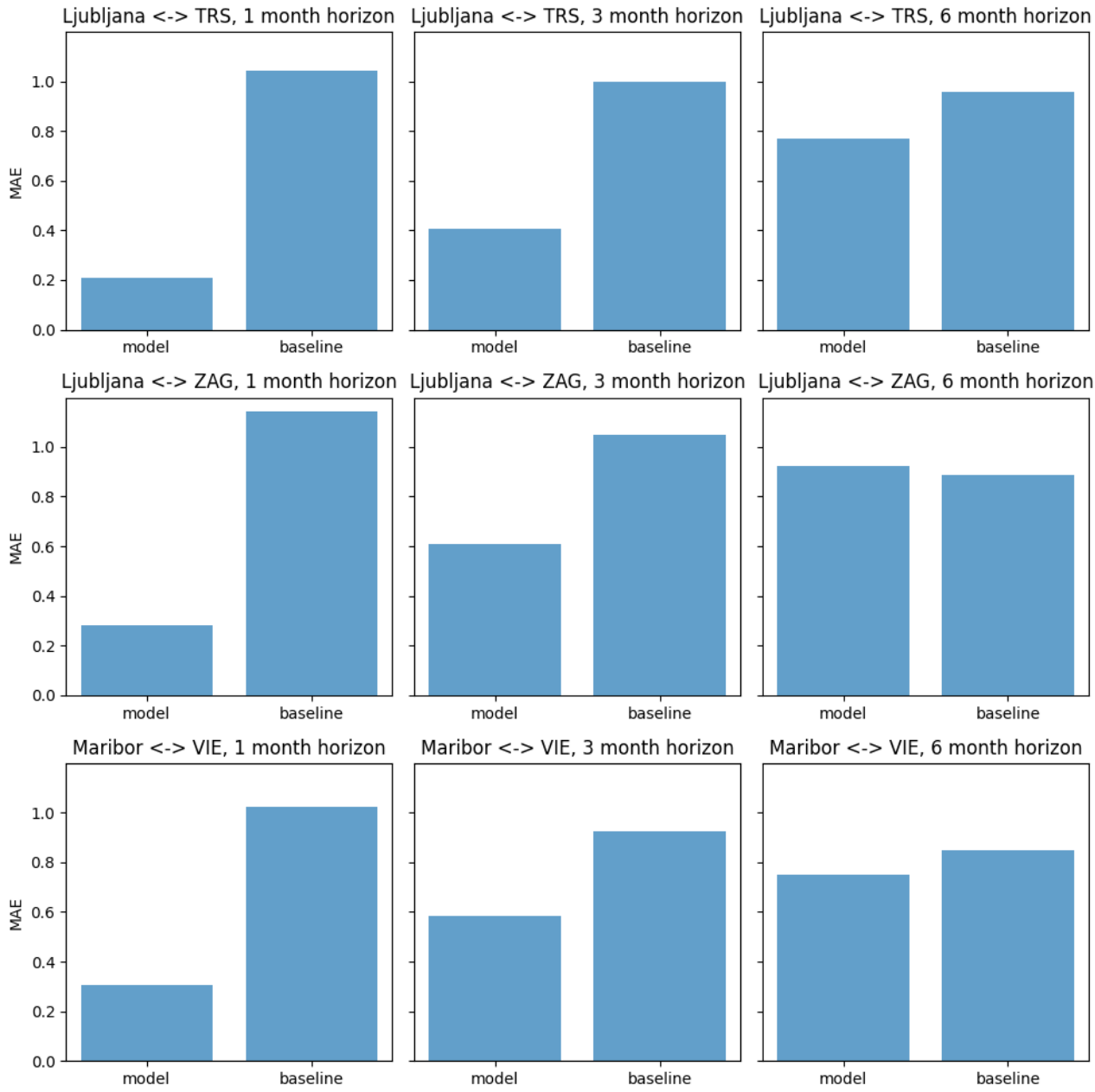

Figure 44: Mean Absolute Error (MAE) of our model compared to the baseline model for the three routes and three different forecasting horizons.

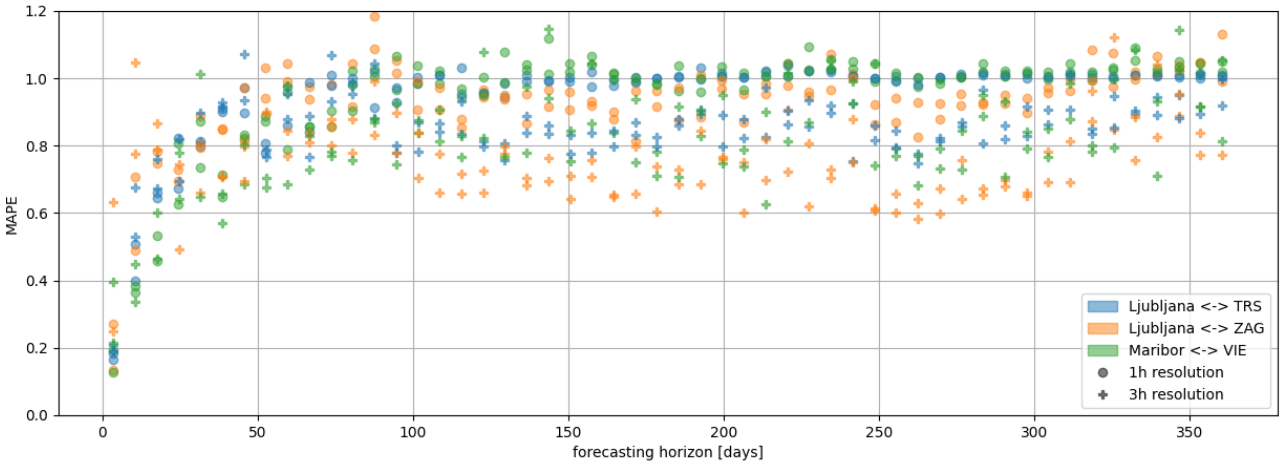

Figure 45: Mean Absolute Percentage Error (MAPE) of our model with respect to the forecasting horizon for three different routes and two different time resolutions.

Up to now, our analysis cantered around median prediction of our model (q=0.5). For other quantile estimation we could use pinball loss function to assess the accuracy of our model, however, we do not have any reference to compare those values with. But we can make a statistical analysis to see if, on average, the frequency of the real number of passengers being between the estimated positions of quantiles is close to the expected frequency. Figure 46 depicts the results of this analysis. We can see that the distributions do not align completely with the ideal case. But nonetheless our model seems to capture the real distribution of the data to a reasonable degree with the exception of slight skewness to the small values in case of the 3h resolution model and to the larger values in case of the 1h resolution model. This effect is probably due to quite complicated data distribution, with many zeros and occasional outliers with very high values. Given that the quantile regression with pinball loss, used here, was intended for normally distributed variates and we have count variates (nonnegative with outliers), these results are satisfactory.

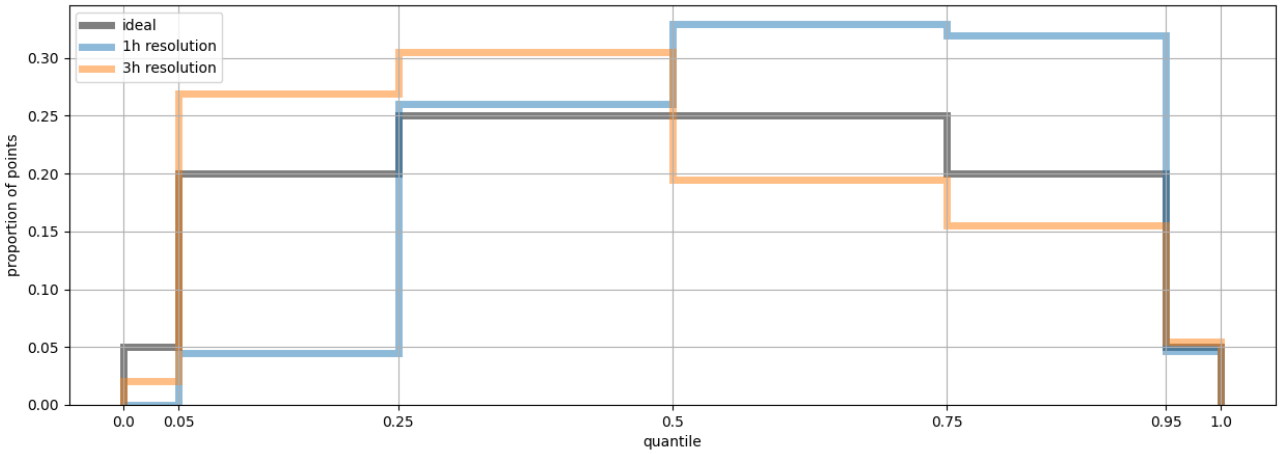

Figure 46: Statistical analysis for how often the real number of passengers falls within the bins determined by the quantiles predicted by our model.

3.3.4 Integration of the Prediction Model

Our model is currently a part of FastAPI web service which operates on JSI servers and is used by other Use Case partners to acquire predictions. The prediction is evoked by sending a query to the web service which contains a specification for the prediction (such as time resolution, forecasting horizon, and so on) in a JSON format. The web service than returns, also in a JSON format, a forecast for the specified route and time range for each of the desired quantiles. This service is used only for inference (making predictions) using an already trained model that was specifically fine-tuned for GoOpti Use Case and using GoOpti data. Because our model uses real-time features, such as the currently known number of passengers, the service regularly connects to the GoOpti data base to obtain this up-to-date data. For this reason, the web service requires an authentication (using username and password) because the model predictions could be used to partially reconstruct data from proprietary GoOpti data base.

Even though our web service was specifically designed for GoOpti Use Case its architecture can also be used for other related Use Cases. This would entail including a new trained model into our service, linking the service to a new data base containing features relevant for that Use Case, and creating a new user that is allowed access to such predictions. The structure of the prediction query for a different Use Case would also most likely differ from the one that applies in the GoOpti Use Case, possibly including other specifications for conditions under which the forecast is performed (such as the way origin and destination is specified geometrically, irregular ways to specify time spans of the forecast, other quantile values not considered in the GoOpti Use Case, and so on). On the model training side, for other Use Cases, it is also reasonable to expect that the optimal model architecture and feature relevance would not fully align with the GoOpti Use Case. This means that feature extraction and model fine-tuning procedures would need to be repeated in order to train a model of sufficient accuracy. Nonetheless, these methods were fully developed in the scope of this project and could be translated to related traffic and demand forecasting use cases.

4. DYNAMIC OPTIMIZATION

4.1 Optimisation of urban freight distribution with the DRT service for last-mile delivery

In deliverable D3.2, various studies were presented that facilitate the integration of urban freight distribution with the DRT service for last-mile delivery. Optimisation techniques based on metaheuristics have been designed and implemented to effectively solve the resulting model, ensuring both efficiency and effectiveness in the solutions obtained. These algorithms consist of hybrid metaheuristic strategies derived from operators included in JSprit, the framework used for developing both the mathematical model and the associated solution algorithms.

4.1.1 Architecture of the Proposed Solution

The solution considers flexibility in handling the physical characteristics of the problem to be solved: different types of vehicles and capacities, access restrictions, time windows, integration of people and parcels, optimization of multiple objectives, etc. On the other hand, the solution is horizontal and scalable, exposed as a service layer that interacts with the optimizer components and provides results asynchronously via an API. Figure 47 shows the different main components that we have accounted for in the development of the proposed module.

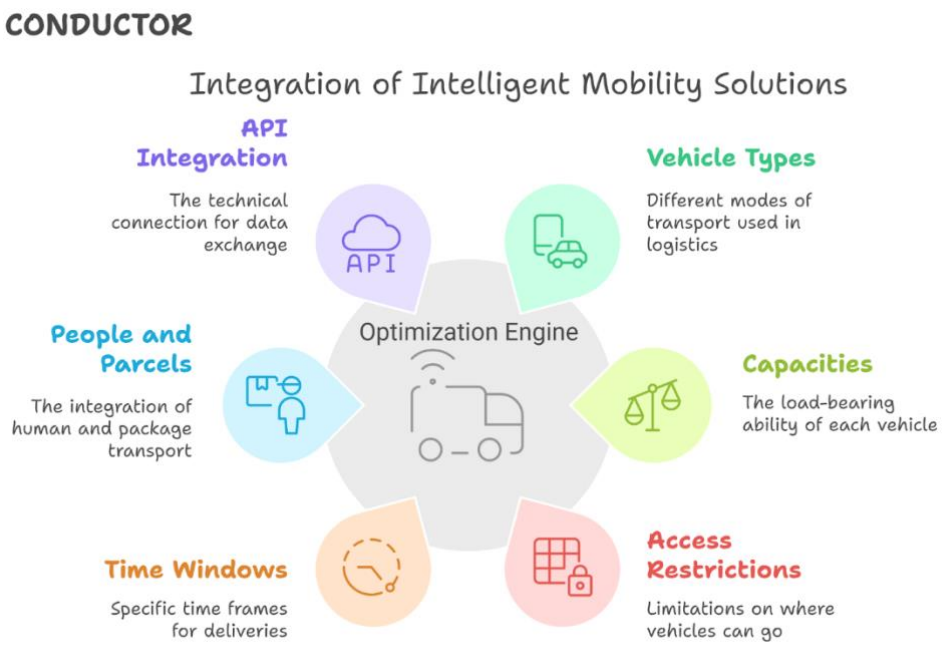

Figure 47: Main components of the solution

The conducted tests have shown that the size of the input data is too large for the optimizer to allocate tasks within a timeframe that meets the actual needs of the system. Therefore, it has been decided to implement a pre-optimizer that will facilitate the optimiser's work by clustering the tasks. This process involves dividing the data into smaller instances that can be efficiently managed by the optimizer, ultimately consolidating all outputs into a single result.

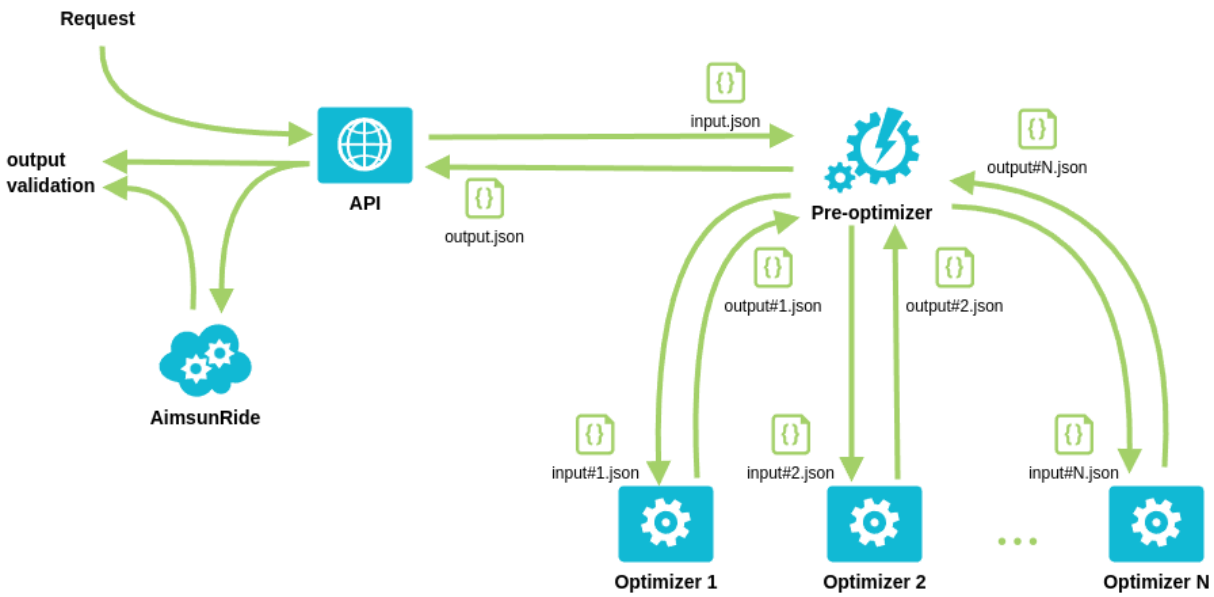

Figure 48: Architecture of the solution

Figure 48 illustrates the final architecture that has been implemented for this component. Requests are made to an API that interfaces with the pre-optimizer, which, based on the input requirements, will cluster and distribute the workload across multiple jobs running in parallel on different optimizer instances. Once the results are obtained, they are consolidated and returned in a single output.json file. This file, along with its validation conducted on Aimsun's premises simulating the scenario, will constitute the final result of the execution.

4.1.1.1 Pre-Optimizer

The pre-optimizer is a software tool developed in Python that acts as a bridge between the input data and the optimizer. Its primary function is to clean and process the data, if necessary, and to ensure that the input size is manageable by a single instance of the optimizer. If the input exceeds this capacity, the pre-optimizer will cluster the tasks, dividing them into smaller groups that can be handled by parallel instances of the optimizer. Given that the optimization problem exhibits exponential growth based on the size of the tasks to be optimized, breaking them down into smaller tasks allows for simultaneous execution, thereby ensuring that the overall execution time of the problem remains reasonable. Subsequently, the pre-optimizer will process the outputs from each instance of the optimizer and consolidate them into a single file, which will serve as the final output of the system and will be sent to Aimsun for simulations.

4.1.1.2 Meeting points

To improve the efficiency of the DRT, strategic meeting points have been defined that allow us to pick up and drop off people at the predefined points instead of doing door-to-door trips, therefore offering a balance between user convenience and operational efficiency. In this way, people in low-density population areas will have access to the service, and logistics operators will be able to organise their itineraries more efficiently and optimise the number of vehicles and drivers used to provide the transport service.

For their identification, bus stops were considered, as well as the intersections of main roads with secondary roads, secondary roads with secondary roads, secondary roads with tertiary roads, secondary roads with residential roads, and tertiary roads with residential roads.

Their identification streamlined the clustering process by reducing the number of geographic locations compared to the original individual sites.

4.1.1.3 Clustering

Since the problem involves tasks related to the distribution of both orders and people, and considering that people require pickups and drop-offs at dispersed points, clustering cannot be performed solely based on delivery points. This complexity necessitates the use of trajectories, defined as the vector connecting the origin (pickup point) to the destination of each task. Additionally, it is important to take into account specific time windows, as there may be trajectories that are close in space but far apart in time. For instance, two tasks may share a physical trajectory, but one must be completed in the morning while the other is scheduled for the afternoon, making it impossible to optimize a single trip.

To deal with the above and the high number of requests for parcel delivery and people pick-up and drop-off, a clustering model has been designed that considers the distance, direction, and time properties of the trajectories. Together with the pre-optimizer, this ensures that the data size is manageable for the optimizer, thus improving its efficiency.
For this, we generate spatial, directional and temporal distance matrices. The spatial distance matrix (distMatrix) represents the difference in geographic distance between two trajectory vectors. The directional distance matrix (directMatrix) considers the directional movements between two vectors. The temporal distance matrix (timeMatrix) represents the difference between the pick-up and delivery time intervals. Finally, these matrices are combined into a final distance matrix:

totalDistMatrix = w1*distMatrix + w2*directMatrix + w3*timeMatrix

Where w1, w2 and w3 are the weights of the distance, direction and time matrices respectively, and add up to 1 in total (w1=0.3, w2 = 0.35 and w3= 0.35).

The clustering algorithm used was agglomerative hierarchical clustering which works with distance matrices. It is also worth mentioning that to optimize the calculation of the matrices and clustering, we used the Just Another XLA[3] (JAX) library.

In Figure 49, 11 clusters can be observed, generated from a sample of 100 trajectories of people and packages, considering their direction, distance, and time interval. CC1, CC2, and CC3 represent the package consolidation centres, provided by NOMMON in the Scenario 1 dataset. The circle indicates the starting point of a trajectory, while the X represents its endpoint.

[3] Accelerated Linear Algebra

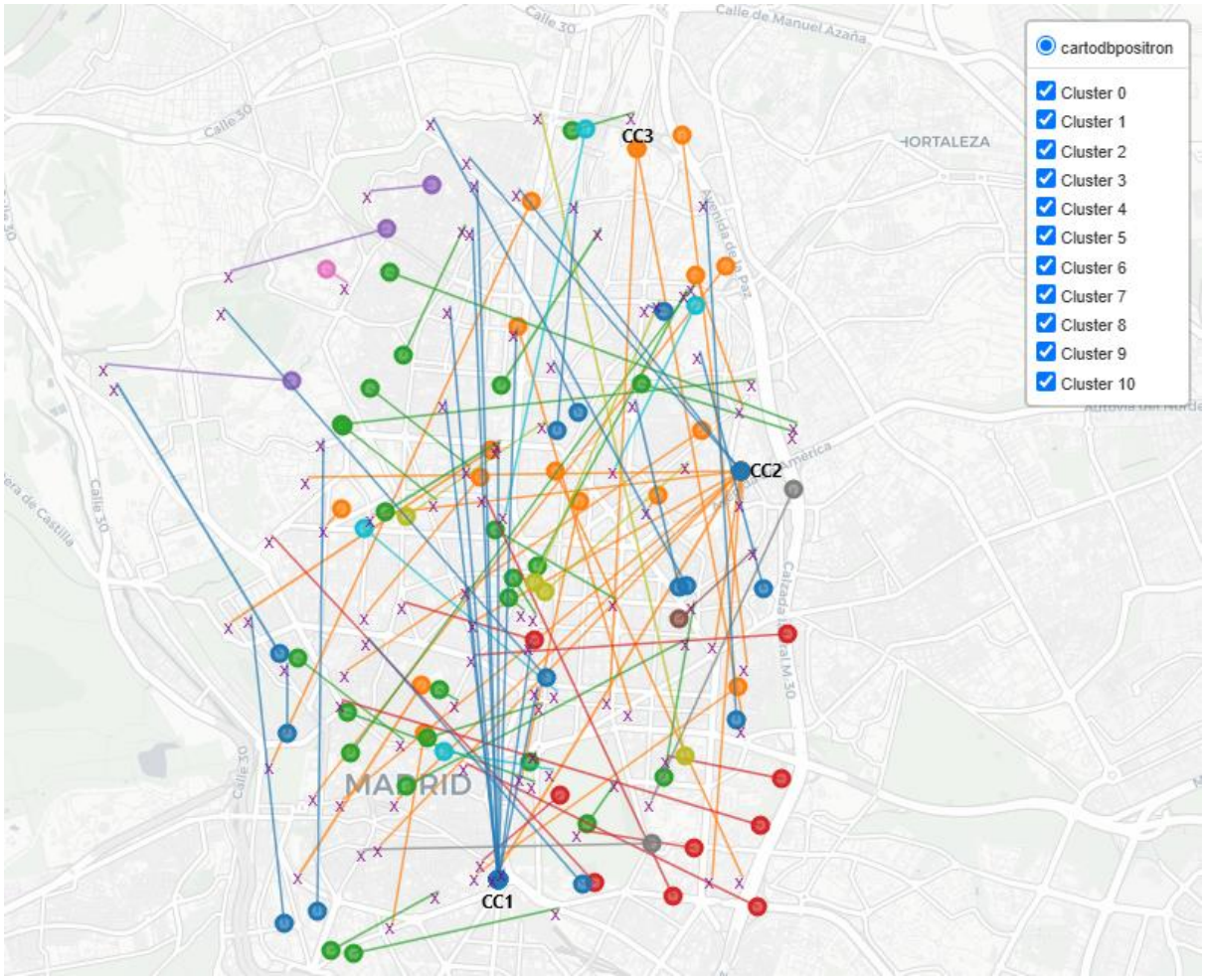

Figure 49: Example of cluster results from a sample of 100 trajectories

4.1.1.4 Optimizer

The Large Neighbourhood Search (LNS) algorithm has been implemented in the proposed solution. It is a metaheuristic where the neighbourhood of a solution is implicitly defined by destroy and repair operators. A destroy operator removes part of the current solution, while a repair operator rebuilds the removed portion. Typically, the destroy method includes some degree of randomness to modify different parts of the current solution, thereby exploring the search space more effectively.

LNS employs a broader neighbourhood exploration technique compared to other classical local search metaheuristic algorithms. It combines different destruction and reconstruction operators (ruin and recreate) along with strategies for accepting or rejecting solutions, making it a hybrid metaheuristic.

The optimization procedure used in this module follows a stochastic approach based on ruin and recreation (R&R) and can be summarized as follows:

1. Initialization: Start with an initial feasible configuration.
2. Selection of ruin and recreation mode: Choose a technique to destroy part of the neighbourhood configuration and another technique to reconstruct it.
3. Determine the number of nodes to be removed.

4. Execute Ruin & Recreate: Generate a new solution configuration using the heuristics selected in step 2.

5. Acceptance criterion: Decide whether to accept the new solution based on a decision rule (e.g., Simulated Annealing, Threshold Accepting, etc.). If the solution is accepted, proceed from step 2 using the new solution; otherwise, restart from step 2 with the previous solution.

4.1.2 Optimization of Freight Deliveries in DRT using Soft Time Windows in FleetPy

This section details the main optimization model used by FleetPy for fleet management. It also describes the optimization approach developed for the integration of logistics into DRT services. The combined services and the developed method will be investigated further for the city of Madrid UC3. The description in this section is based on the optimization model described in D3.2 including further developments since its submission.

The overall fleet management problem in DRT services falls under the category of Stochastic and Dynamic Vehicle Routing Problem (SDVRP). FleetPy uses a dynamic simulation environment, where the ride requests are revealed over time. It uses two types of control algorithms for assigning vehicles to requests. The first type includes quick response algorithms that respond to the requests as soon as they are revealed to the system. The second type accumulates the requests for a brief period and formulates an optimization problem for assigning vehicles to ride requests. The latter category is known as batch optimization.

FleetPy represents the street network as a directed graph $G_{op} = (N_{op}, E_{op})$ with nodes N_{op} and edges E_{op}. Each edge $e \in E_{op}$ is associated with a distance d_e and a travel time $\tau_e(t)$. In CONDUCTOR, the street network and the travel times on edges $\tau_e(t)$ are derived from the macroscopic simulation of Aimsun Next. In D2.1, the edge travel times were described to be derived in real time Aimsun-FleetPy bridge and Aimsun Next's microscopic simulation. However, in the final implementation of the method, macroscopic simulation in Aimsun Next is used instead of a microscopic model in order to satisfy the requirements of UC1 Madrid, as described in D2.5.

As shown Figure 50, FleetPy conducts the following major steps in each simulation loop:

1. Boarding and alighting of passengers as well as pick up or drop off events of freight requests are registered by the fleet operator.
2. New DRT customers enter the simulation and request a trip i at the time t_i by providing origin $o_i \in N_{op}$ and destination $d_i \in N_{op}$, and must be picked up with a maximum pickup delay of w_{max} starting from t_i, otherwise the customer does not take the trip and leaves the system.
3. The operator evaluates whether it can serve the request within the given time constraints. If so, an expected pick-up time t_i^{pu} and drop-off time t_i^{do} is provided.
4. Operators accommodate a subset of freight requests into vehicle schedules. The soft time windows assigned to the freight requests are also used for this purpose.
5. If the freight request cannot be served within the assigned soft time widow, the time window is extended, and the parcel receiver is notified of the modified delivery time window.
6. The operator assigns new/updated schedules to its vehicles.

In order to assign customers to vehicles, FleetPy first builds a pool of schedules. A schedule is defined as a series of stops at network nodes N_{op} where boarding and alighting processes of vehicles are conducted. In between these stops, vehicles are travelling on the fastest route in the network G_{op}. There are multiple possible permutations of stops as soon as more than one passenger is assigned to a vehicle $v \in V$, which are further increased when freight requests are also considered.

The k-th possible permutation of stops for the schedule $\psi_k(v; R_\psi, P_\psi)$ serving all passengers and freight requests in the set R_ψ and P_ψ, respectively, is considered feasible if all the following conditions are satisfied:

1. the drop-off stop succeeds the pick-up stop for each customer.
2. the number of on-board customers never exceeds the vehicle capacity (c_v).
3. each customer is (supposed to be) picked up before a maximum waiting time w_{max} elapsed.
4. if the operator offers a pooling service, the maximum additional travel time must not exceed a detour factor δ_{max} compared to a direct trip.

As described in deliverable D2.1 and D2.5, FleetPy considers three levels of logistic integration: status quo (freight and passengers served by separate fleet), moderate (both served by same fleet, however, no parcel can be collected or delivered in between a passenger trip) and full (a freight request can also be collected or delivered in-between passenger trips). For a moderate integration to be considered, the optimization problem additionally uses the following constraint:

5. while passengers are in the vehicle, no stop is allowed where only parcels are picked up or dropped off.

Schedules are rated by an objective function $\phi(\psi_k(v; R_\psi, P_\psi))$. The goal of the fleet operator is to assign schedules minimizing the aggregated objective function for all its vehicles. $\phi(\psi_k(v; R_\psi, P_\psi))$ can be modelled in multiple ways. Since the submissions of D3.2, TUM modified the objective function to also include freight delivery time windows:

$$\phi(\psi_k(v; R_\psi, P_\psi)) = d(\psi_k(v; R_\psi, P_\psi) - P\,|R_\psi| - f(P, T_{P_\psi})\,|P_\psi|) \qquad (3)$$

where $d(\psi_k(v; R_\psi, P_\psi)$ refers to the distance to drive to complete the schedule. P is a large assignment reward to prioritize serving customers and parcels over minimizing the driven distance. However, while P is fixed for customers, the reward for freight request is a function $f(P, T_{P_\psi})$,dependent on P and the difference between start of assigned time window and the actual delivery times, given by the set T_{P_ψ}.

To solve the above optimization problem for combined logistic and DRT service, FleetPy uses heuristic approaches to build a set of schedules that include freight requests. The fundamental concept of the heuristic is to only insert freight requests into vehicle schedules if the detour to pick up or drop off a freight request is small or if delivery of the freight request is urgent due to the assigned delivery time window.

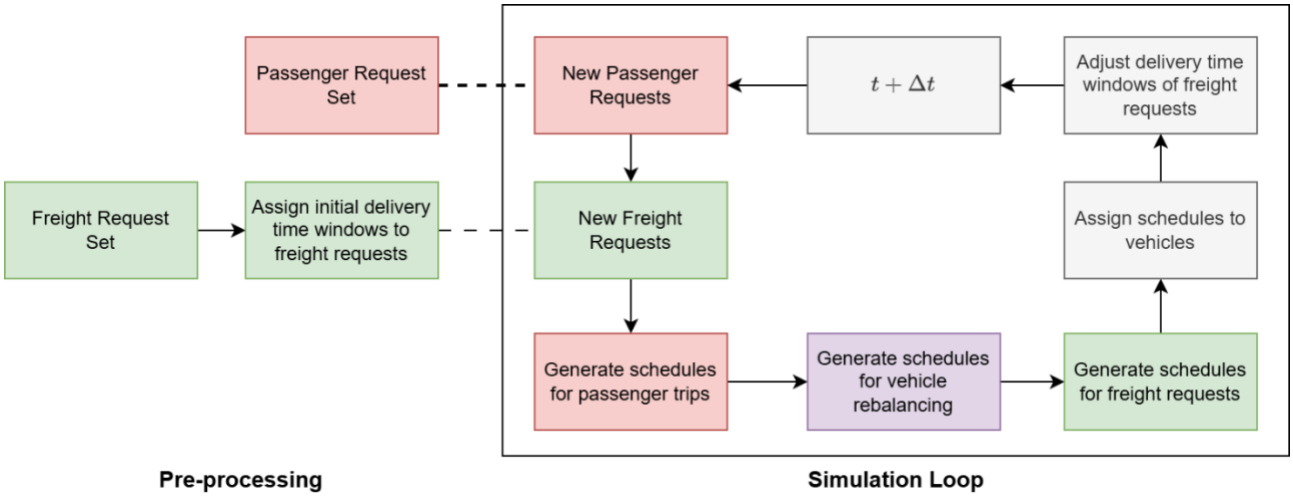

Figure 50: Main Simulation Loop of FleetPy

4.1.3 Preprocessing for Initial Delivery Time Assignment to Freight Requests

As mentioned in D3.2 and D2.5, within CONDUCTOR, TUM investigated the mechanism for providing estimated delivery time windows to the freight requests. This replicates a similar strategy to the current freight services where the delivery of freight requests is given a rough delivery time-windows ranging in hours. However, in a combined DRT service for passengers and freight, the complexity of assigning time windows to freight requests is significantly higher than dedicated freight services. The fundamental challenge is the significantly different nature of freight and passengers requests. The passenger requests are added dynamically into the system and must be picked up within w_{max}. This requires the fleet with sufficient empty seats to be close to the potential customer locations for higher service quality. In contrast, all freight requests are known beforehand at the start of the day and do not have strict time constraints for delivery. Thus, adjusting the freight requests into DRT schedules with dynamic DRT customers raises the problem complexity.

In the estimated delivery time window strategy, all freight requests are first assigned initial delivery time windows (in hours) in a preprocessing step. These time windows are used as soft time constraints while assigning freight requests to DRT fleet. The motivation here is that using these initial delivery times, the freight receiver will get a notification of the parcel arrival time, making it easier for the receiver to be available at the time of delivery. Later, according to DRT availability, if some freight requests cannot be served within the assigned time window, the time windows are slightly modified and the freight receiver is accordingly informed of the changes. Thus, the more accurate the initial time window assignments are, the less changes to the delivery time windows would be required during the actual DRT operation. Therefore, the preprocessing method for initial time window assignments play a crucial role for higher service quality of freight requests.

Looking at their importance, TUM investigates two methods for initial delivery time windows. Both methods use spatio-temporal aspects for this purpose. First, the freight and DRT customers are divided into regions according to their origin and destinations. For simplicity, the city districts are used for this purpose. Then, two different approaches are used:

1) **Region-Based Equal Delivery Time Windows:** In this method only the freight requests are used. Within each region, the number of freight requests are counted. Later, within the working hours, all freight requests are assigned a time window of 2 hours, such that, for each region, the number of freight deliveries for each 2 hour time window is equal. The downside of this strategy is that it does not consider the DRT customer demand. Thus, there are higher

chances that such a strategy may not be able to meet the initially assigned travel time window.

2) **DRT Demand based Delivery Time Windows:** Since the freight requests are served while also serving the DRT passengers, considering the forecast of DRT passengers may serve as an important factor for the assignment of delivery time windows. Thus, in this method, first the number of DRT passengers are counted for each combination of region and 2 hour time windows during the working hours. Later, the freight requests in each region are assigned 2 hour time windows such that the ratio of freight requests in each region and 2 hour time window is the same as the DRT requests. The motivation here is that the vehicles would be readily available in these regions due to DRT passengers, and thus, there is a higher chance that the initially assigned delivery time windows are fulfilled.

4.2 Risk-based Adaptive Routing in Public Transport

4.2.1 Introduction

Even though public transport (PT) operations are planned meticulously, things may play out very differently in practice. Disruptions and disturbances range from minor increases in dwell times due to an unexpected high boarding rate to vehicles breaking down. Such perturbations or events can manifest or propagate in the realized PT schedule, leading to delays which may affect passengers in many ways, e.g. by missed transfers and late arrivals. Consequently, uncertainty plays a critical role in passengers' decisions regarding their route and departure time.

Since uncertainty is only revealed over time, passengers need to hedge against it when making decisions. Yet, they oftentimes have the opportunity to respond to changing information by postponing some of their decisions or adjusting their initial decisions over time. As such, passenger decisions are not fixed a priori but depend on what is happening in the network and the available information, cf. (Hall, 1986).

When making decisions, travellers are not only concerned with expected or scheduled travel and arrival times but also assess and anticipate potential deviations. They desire a trip that is reliable, i.e. one for which the arrival or travel time is consistent with what was communicated. In fact, travellers are in general strongly averse to scheduling mismatches (Small et al., 1999) and commuters especially consider travel time reliability to be one of the two most important factors for route choice (Abdel-Aty et al., 1995). Consequently, risk-averseness in the face of the uncertainty needs to be captured when modelling passenger decision making in order to accurately estimate or predict passenger flows.

In assignment studies, passenger decision-making is typically modelled as a one-shot deterministic optimization problem, which in this setting relates to having full information, and selecting a priori a departure time and route with minimum *cost* (or maximum utility). Such an approach does not suffice when trying to capture the complexity of passenger decisions under limited and evolving information. First, the deterministic approach does not account for the possibly infinite supply scenarios that may occur. Second, in PT, it is inevitable for passengers to make recourse decisions in response to new information that becomes available during the trip, because they either have to (e.g. if they missed a transfer) or they want to. Although stochastic, multi-stage extensions exist, most studies assume decision-makers to be risk-neutral (Gao & Chabini, 2006), include an objective function that captures non-constancy in travel times but not necessarily the potential impact, or focus on (car) traffic networks where travel time uncertainties typically only impact costs, but not the available routes (Gao et al., 2010).

In this paper, we propose a novel framework for modelling passenger route choice under uncertainty, which both captures the adaptive nature of passenger decisions in PT systems and includes risk as a relevant factor for decision-making. Rather than considering a single study-specific reliability metric, we integrate general risk measures into a stochastic route choice and departure time problem, where passengers make and adapt decisions based on a summary and quantification of the uncertainty in PT travel times.

4.2.2 Route Choice under Uncertainty and Risk

Transportation networks are by their very nature uncertain, hence travel times cannot be predicted with total accuracy in advance. This holds true both for traffic networks and PT systems and requires travellers to come up with more elaborate routing strategies than what is modelled by the classical deterministic shortest path problem. Instead of fixing a path in the beginning, which is followed through regardless of the situation, it is typically assumed that travellers adapt their route along the way depending on the realization of the network and updated information about the past, current and future situation. An approach to model this is in a stochastic time-dependent (STD) network, where the travel time distributions of the links might change over time. Considering adaptive decision-making, Optimal Routing Problem (ORP) in an STD network was first studied by (Hall, 1986), which showed that in such a network this was indeed more effective than simple paths. Similar problems and their solution approaches have been studied, including e.g. dynamic programming approaches as well as various algorithms and heuristics (Chabini, 1999; Gao et al., 2010; Gao & Chabini, 2006; Miller-Hooks, 2001; Miller-Hooks & Mahmassani, 2000).

However, these studies usually solely focus on traffic networks. Even though these considerations are also clearly relevant for the PT setting, the uncertainty in PT networks is oftentimes more difficult to model. When one is traveling in a traffic network, in most cases travel time is the only uncertainty of importance for route choice. Even though incidents, congestion or construction might occur and influence the trip duration, the probability that they cause failure or non-existence of a route is usually negligible. This is not the case for PT. Here, a link to travel on is not a street, which is always available regardless of the time and circumstances. Instead, a link is a trip (e.g. by bus or train) between two stations offered by the PT system operator. This service operates at a schedule and thus is only available at certain times (which might also vary due to the circumstances in the network), or which might even be cancelled and thus not exist at all. Routing in stochastic networks in which the existence of links is random has been studied e.g. by (Andreatta & Romeo, 1988; Croucher, 1978). However, their focus lied solely on the uncertain existence of links, while travel times were deterministic. Thus, there is a need to study stochastic networks which model both: the uncertainty of travel times as well as the uncertainty of the existence of the routes in general.

Regardless of the design of the underlying stochastic network, all approaches on routing under uncertainty from above agree that the route choice decision is based on the total travel time of the trip. The optimal route is determined by minimizing the expected travel time and it may be adapted by recourse decisions along the way. However, passengers are actually highly concerned with the uncertainty of travel times as they often cause missed transfer and cancelled trips in PT networks. In fact, there are various ways for passengers to evaluate and deal with the variability of travel times in PT, (travel time) reliability being one of the most commonly studied approaches in transport studies. It relates to the evaluation of travel time variations according to the preferences and interpretations of the different participants in traffic and PT. The formal definition is usually dependent on the respective setting, reflected by the numerous reliability measures, cf. (Zang et al., 2022). Many of the proposed metrics are easy to compute and have an intuitive interpretation, with travel time standard deviation (or variance) (Carrion-Madera & Levinson, 2010; Frittelli & Maggis, 2011) probably the most well-known. A major drawback of these metrics is that many measure non-

constancy, but not necessarily impact. The consequences resulting from a deviation are often not accounted for: even a very reliable trip might be extremely late on occasion, or might even be always late for that matter, and hence it might be considered a risky trip anyhow. Thus, reliability may not adequately describe passenger's risk preferences entirely if they neglect the `value of unreliability' (Watling, 2006). Indeed, the risk of deviating from the initially planned route due to travel time variations has a direct impact on how passengers make decisions regarding their routes and departure times, as they adapt these specifically for trips that are considered unreliable, cf. (Small, 1982; Tirachini et al., 2014).

In this study, we argue that risk rather than reliability should be adopted as metric to aggregate and quantify both the likelihood and consequences of travel time variations and could be consequently adopted as an objective function in a passenger path choice model. The concept of risk stems from finance, where the risk of a financial position like a stock or asset is quantified by functions, the so-called risk measures (Föllmer & Schied, 2016). Here, risk is oftentimes interpreted as the amount of cash one should hold to be safe of possible future monetary losses and is used to make investment decisions, but also to prevent bankruptcy of firms. An axiomatic approach was developed by (Artzner et al., 1999) to assess the ability of a risk measure to reflect the characteristics of real-life financial applications and differing risk attitudes of multiple involved parties, allowing for a wide-spread interest in and use of risk measures both within and outside the financial applications (Egidio dos Reis et al., 2009; Filippi et al., 2020; Insurance, 2006; Szegö, 2002). Previous risk analyses in a PT context usually focused on evaluating random travel times. Similar to the approaches on reliability, risk is typically assessed by being compared to a reference, which in PT is usually the timetable, prompting the idea of the risk of not being on-time, missing a transfer or a trip generally not going as planned. Previous studies include two prominent examples of monetary risk measures (Chen & Zhou, 2010): The Value at Risk (V@R) and especially the Conditional Value at Risk (CV@R) acknowledge the often heavily right-tailed probability distribution of travel times and delays (Philippe, 2006; Rockafellar & Uryasev, 2000). This is also in line with the risk aversion towards delays which is congruent with the economical concepts described in Cumulative Prospect Theory (Kahneman & Tversky, 1979) and has been adopted in risk-based routing decisions before (Gao et al., 2010). To allow for different risk behaviour, a risk acceptance parameter can be set accordingly.

Inspired by travel time reliability and the risk theory from financial mathematics, we develop a risk theory that fits into the stochastic decision-making process regarding route and departure choice of passengers in PT. By covering existing examples of risk measure like V@R and CV@R, but also e.g. penalties for early- and late-arrival, this extends existing approaches by accounting for the heterogeneity of risk attitudes within the population.

4.2.2.1 Problem Illustration

This research studies adaptive routing under uncertainty in PT and how risk can be included into these considerations. There are three aspects we aim to address specifically:

1. Define a model for adaptive routing of passengers in a stochastic network, accounting for the uncertainty of both travel times and the existence of the routes in general.
2. Define the risk of a route as a metric to assess the impact of uncertain realizations of timetables.
3. Use the route/departure choice model to analyse different risk operator in terms of their suitability to reflect passenger preferences.

As neither of these points is trivial, we first want to illustrate the difficulties arising with these topics. For this, we introduce a simple example.

Consider a PT passenger, traveling from station s_0 to s_2. The timetable, illustrated in the space-time graph in Figure 51, provides three possible routes $R = \{r_1, r_2, r_3\}$. The first route r_1 (orange) is direct but also stops at s_1. Alternatively, passengers can leave the orange line at s_1 and transfer into the pink one for route r_2. Or one could take the green line and then transfer into the orange line, yielding route r_3. While these are simply the scheduled routes according to the timetable, travel times may vary in reality and adapting the route may become necessary. The pink line might be so delayed that it makes no sense to transfer for r_2. Even worse is the case if the transfer of r_3 is not successful due to delays of the green line. Then one is either stuck at s_1 or might get lucky and catch the delayed pink line to successfully finish the trip.

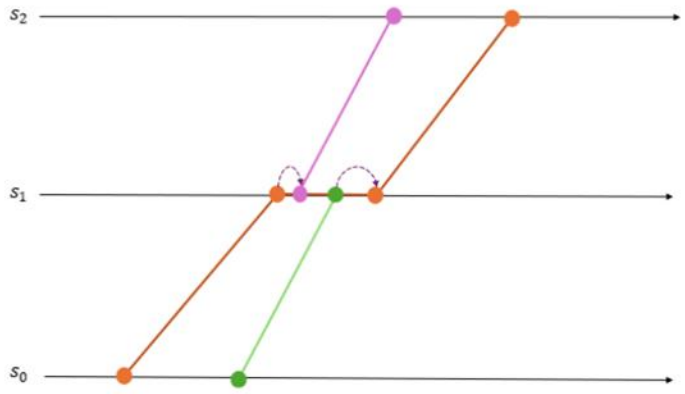

Figure 51: Available connections from s_0 to s_2 as scheduled in the timetable

So while R is the set of paths the timetable suggests to travel from s_0 to s_2, reality might look very different from this. In practice, one does not know the actual realization of the timetable, but may learn or be informed about the delay distributions. Using an information system, passengers may be able and willing to take recourse action at a later time. Hence, choosing a route out of R equals an initial boarding decision with a path in mind, which is either followed through with or corrected according to certain strategies, in case the realization of the path includes e.g. a missed transfer. Possible strategies to move through the network are called **routing policies**, and the ones we will consider here are described in Table 7.

Table 7: Possible routing policies to travel from s_0 to s_2 in above's PT system

Policy	Name	Routing Decisions
μ_1	Safe & direct	Orange line until
μ_2	Safe & quick	Orange line to, then transfer into the pink line if it is on-time Alternatively orange line until
μ_3	Risky & quick	Green line to, then transfer to into orange line if possible If it is not possible, but pink line delayed, transfer into pink line If no transfer possible, finishing trip is not possible

The question remaining is which of these routing policies to choose from. This choice clearly depends on the preferences of the passenger, whether a short travel time, a certain arrival or departure time, or a more reliable or less risky one is asked for. In the following, we assume for the

sake of our example that passengers are solely averse to travel time variability and its consequences. That is, their objective is either to maximize travel time reliability of the route or minimize risk. With this focus, we are only interested in the deviation from the scheduled arrival. For simplicity, we assume that the actual arrival time of a line can only be equal or later than the scheduled one, causing delays. These delays are realized upon arrival at station s_1 and are the same on arrival at s_2. The probability distributions of these possible delays D (in minutes) for each line $\{o, g, p\}$ (orange, green, pink) are

$$P(D_o = 0) = 0.95,\ P(D_g = 0) = 0.9,\ P(D_p = 0) = 0.5,$$

$$P(D_o = 5) = 0.05,\ P(D_g = 10) = 0.1,\ P(D_p = 30) = 0.5.$$

Here, $D_o = 0$ means the delay of the orange time equals 0, i.e. it is on-time, while $D_o = 5$ refers to a delay of 5 minutes.

Based on this, we derive the probability distributions of the delay D_i of the policies $\mu_i, i = 1,2,3$. As the originally planned arrival time, we choose the one which corresponds to the realization of a policy where nothing goes wrong. Then it is $D_1 \sim D_o$, as the policy simply sticks with the orange line. A trip following policy μ_2 is considered on-time if the transfer from the orange to the pink line is chosen, which only happens if both the pink and the orange line are on-time. If one line is however delayed, μ_2 suggests to stick with the orange line, resulting in a delay of 15 minutes (if pink is delayed, but orange is on-time) or 20 minutes (if orange is delayed) as the orange line is scheduled to arrive 15 minutes after the pink one. For policy μ_3, it is considered on-time, if the transfer from the green to the orange line is successful and the orange line is on-time. If the orange line is delayed, the transfer is successful both times, for the green line being either on-time or delayed. In this case, the delay of μ_3 is 5 minutes. If the green line is delayed, but the orange line is on-time, there are two possibilities. Either the pink line is also delayed and one transfers into this one, resulting in a delay of 15 minutes. If this is not possible, one is stuck at station s_1 and cannot finish the trip in time. Based on a periodic timetable of 60 minutes, we denote the delay in this case by 60 minutes. That is, we result in the following probability distributions of the delays of μ_2 and μ_3:

$$P(D_2 = 0) = 0.475,\ P(D_3 = 0) = 0.855, \tag{4}$$

$$P(D_2 = 15) = 0.475,\ P(D_3 = 5) = 0.05,$$

$$P(D_2 = 20) = 0.05,\ P(D_3 = 15) = 0.0475,$$

$$P(D_3 = 60) = 0.0475.$$

As the passenger does not know the realization of the PT system and therefore also does not know the delay which awaits them, they have to base their route choice solely on the available information before departure. That is, at station s_0 they choose their strategy (and with this their initial boarding decision) based on these probability distributions.

While this notation is sufficient for this simplified example, it lacks a lot of information which is needed to model realistic PT systems. In reality, it is necessary to include the passing of time and update the available information and probability distributions, especially since there usually are more than only one stage where passengers can take recourse action. Real PT systems are also much larger, making it necessary to keep track of it in a designated network as well as a more specific timetable. While keeping track of the different routes and policies in this example already needs a lot of consideration, modelling adaptive routing in larger networks becomes instantly more complex.

To find the optimal routing policy in the presented example, the delay distributions of the policies need to be evaluated according to the preferences for reliability and risk of a passenger. Reliability in PT is often strongly related to punctuality. So we are looking for the connection which is most likely on-time, i.e. $\max \quad P(D_i = 0)$ over policies $i = 1,2,3$. According to this definition, μ_1 is considered the most reliable, as no delays can occur due to a missed transfer. Then, μ_3 is also quite reliable and might be a valid alternative if passengers might additionally be looking for a shorter travel time. However, the possibility that this journey cannot be completed if the transfer is missed (which is substantial), is disregarded by focusing only on punctuality. Thus, a reliable option in this sense may be risky.

Instead, we need to formally define risk. A common choice is the Value at Risk at level $\alpha \in [0,1]$, which returns the delay that will not be exceeded with probability $1 - \alpha$ (Philippe, 2006). Assuming confidence level $\alpha = 0.05$, we observe for above's routes that the delay in 95% of the time does not exceed the following thresholds:

$$V@R_{0.95}(D_1) = 0, V@R_{0.95}(D_2) = V@R_{0.95}(D_3) = 15. \quad (5)$$

We observe that the direct connection is considered less risky than the policies including a transfer. But the two adaptive policies are considered equally risky, even though they vary greatly in their design. This results from the fact that the worst-case scenarios which occur with probability lower than 5% are not taken into account. For μ_3 there exists the rare probability to miss every transfer and be 60 minutes delayed, hence its worst case scenario is much more unpleasant than the one from μ_2. This may be contradictory to being considered equally risky according to (6.2).

An alternative risk measure is the Conditional Value at Risk, where risk is computed as the conditional expectation over all worst case events which occur with probability less than α (Philippe, 2006). For the given routes, that yields

$$CV@R_{0.95}(D_1) = 0.25,\ CV@R_{0.95}(D_2) \approx 15.48, CV@R_{0.95}(D_3) = 37.5. \quad (6)$$

We observe that this is a more conservative risk measure, valuing tail events more and accounting for the possibility of complete failure of route μ_3 by correcting the risk upwards significantly. This is congruent on how one may feel about this option, making $CV@R(D_i)$ a possible risk measure to be adopted in a stochastic optimization model. Although mathematically and conceptually appealing, CV@R may be difficult to interpret or explain compared to the measures introduced, hindering the potential use of it in practice.

It becomes apparent that the choice of risk measure is relevant for accurately modelling passenger route choice. To define it properly and study their effects, an adequate route choice model is needed.

While the PT system in real-life might be more complex, many of the struggles in the example remain. Passengers have to make boarding decisions without foreknowledge on the realization of the trip. Even though they can adapt their decisions later on, they also typically have a route in mind which can be considered an optimal solution of a least cost path choice model, based on a probability distribution learned based on experience, where the existence of paths and their cost is based on their random travel times.

4.2.2.2 Route Choice Model

While V@R and CV@R are intuitive and common ways to describe the risk perception of passengers, there are many more possibilities to choose from. In order to find the best among all of

these options, it is necessary to understand how passengers move through the PT system in general, that is their route choice and decision-making process. Due to the uncertainty of depending on information and a multitude of scenarios a passenger can face during their trip, they usually do not make a one-off path decision at the beginning of their trip and stick to it regardless of the course of the journey. Instead, passengers are assumed to be more flexible and to take recourse action when real-time information becomes available - either because the information suggests a preferable route the passenger might switch to voluntarily, or because their original route is no longer available and they are forced to reconsider. That is, instead of a fixed path passengers choose a strategy for making decisions based on the current state of the network when traveling.

In the following, we will describe a model of adaptive route choice in an STD network which accounts for this travel behaviour and which will serve as the basis for studying the suitable risk measure. This is based on the model described by (Gao & Chabini, 2006) which was designed for traffic networks. However, we will adapt it to also account for a crucial difference of traffic and PT networks: the existence of certain links (and with this, routes through the network) is also stochastic and time-dependent in PT. Hence, the underlying STD network will be adapted to fit this context. Based on this, we will be able to discuss the importance of risk in passenger route choice and the differences and impact of the various risk measures.

4.2.2.3 The Network

Consider a stochastic time-dependent network $G = (N, A, T, P)$. The stations are captured in the set of nodes N, which is fixed and deterministic over time. The links (or arcs) A between the stations are trips between stations provided by the PT service. The travel times of the links underlie the probability distribution P, which possibly depends on the time period $t \in T = \{0,1, \dots, K-1\}$ (as for e.g. during peak hour travel times are generally longer). Trips are scheduled by the timetable and depend on the time: Whether a link between two nodes exists in time period t depends on whether a trip is scheduled at this time. However, the existence of trips is also stochastic, as they might be cancelled or delayed (and thus exist not in time period t but in $t+1$). This is also captured in the probability distribution P.

Following the notation from (Gao & Chabini, 2006), we furthermore denote by $C_{ij,t}^{\sim}$ the random travel time on the arc connecting the nodes i and j in time period t (where the randomness is made apparent by ~). The set of support points of the probability distribution (i.e. the scenarios with non-empty probability) is assumed to be finite and denoted by $P = \{v_1, \dots v_R\}$. For a given support point $v_r \in P$, the probability is $p_r = P(v_r)$ and the realized travel time of an arc is accordingly $C_{ij,t}^{r}$. Finally, we denote the single destination node by d.

4.2.2.4 The Decision Process

A **decision** of a passenger describes the choice of which link to take next. We assume that passengers can make decisions only at nodes (i.e. stations) since these are the only places where they can actively follow up on their decisions. Decisions are furthermore made based on the **current state** $x = (j, t, I)$, which consists of the **current node** j of the respective passenger, the **current time period** t and the **current information** available to the passenger i. The current information consists of a set of available realized links and their travel times that are useful for making inferences about future link travel times. A discussion of what this means in detail and how different kinds of information influence decision making is provided by (Gao & Chabini, 2006). As it depends on time and location, current information will be denoted by $I = I(j, t)$ whenever node and time are not clear from the context. Regardless of whether the available information includes no, partial or total knowledge about the system state, we assume that passengers know the probability distribution of travel times a priori.

A **routing policy** is a decision rule that specifies which node to take next at each decision node based on the current time and information, i.e. a mapping from a state x to a decision for every state. A path is a special case of a routing policy, where the decision only depends on the node but is the same regardless the time and available information. For any realized scenario of the STD network (the PT system), a policy manifests itself as a path.

Next, we aim to determine **optimal routing policies**, which are routing policies that move a passenger through the PT network from their origin to the destination station while optimizing some objective. In this model, future adaptive choices of the routing policies are fully considered. In (Gao & Chabini, 2006), an optimal routing policy minimizes the total travel time of the trip, however, we will discuss alternatives to this objective focusing on the risk of the trip.

During a trip, a passenger will experience a series of states $\{x_0, x_1, \dots, x_S\}$, which we will call a **state chain**. The initial state is x_0, while x_S denotes the state where the destination node d is reached, i.e. where d is the current node. Current nodes of a state chain form a path and S is the number of links in that path. Given initial state x_0 and routing policy μ, one can experience multiple state chains depending on the realization of the STD and the set of possible state chains is denoted by $M(x_0, \mu)$.

While (Gao & Chabini, 2006) assumed that there was always at least one path from any node to the destination node under any possible realization of link travel times, it is of the very nature of PT systems that this is not the case. Links may be cancelled or a path may not be completed in the time frame T due to delays or other complications. However, we assume that in the deterministic network provided by the timetable of the PT system, there is at least one path available from any node to the destination node which is completed within the given time frame T. To evaluate the probability of not arriving in-time or at all is a crucial aspect of the subsequent analysis.

4.2.2.5 The Minimization Problem

The Optimal Routing Policy (ORP) Problem introduced by (Gao & Chabini, 2006) determined an optimal routing policy by minimizing the expected total travel time as the focus of their studies of routing in a traffic network. This is a common and reasonable approach, as the length of a journey is an important factor in routing decisions. Furthermore, the linearity of the expectation operator allowed for solving this problem dynamically and also provided optimality conditions for the solutions of the ORP problem. This can analogously be done for this setting, where the difference mainly lies in the probability of non-existence of links. However, we assume that this is accounted for in the probability distribution P, thus the very general formulation of (Gao & Chabini, 2006) still holds. In practical application and computations, this will however be more difficult.

Instead of minimizing total travel time, we instead tackle the problem of minimizing risk. Recall that risk is always evaluated in comparison to a reference value, which in PT usually is a path planned ahead and associated with travel and arrival times provided by the timetable. In the following, we assume this deterministic reference path p to be chosen already and denote its scheduled, deterministic arrival time by t_p. Note that the choice of this path is in itself not trivial and can also be described as an optimization problem (possibly in a non-stochastic network). What is considered the best path again highly depends on the preferences of the respective passenger.

Moreover, the risk of a policy depends on the probability distribution of the link realizations, travel times and the knowledge a passenger has about these distributions. That is, the risk depends on the current state.

Finding an optimal routing policy which minimizes the risk with respect to a previously planned path P thus amounts to solving

$$\mu^* = \arg\min_{\mu} \quad \rho(\mu, p, x_0)\ \forall x_0. \qquad (7)$$

The question remains what a reasonable choice of risk measure is, as well as whether and how this optimization problem can be solved.

4.2.2.6 Example Discussion

The formulation of (Eq. 7) to derive the optimal policy is very general. This is advantageous as it can be tailored to various different settings and problems and allows a more abstract discussion about the functionalities and properties of different risk measures as a way of calculating the optimal routing strategy. However, (Eq. 7) is in itself far from concrete. To make it more approachable, we show its use and also why the abstract formulation is useful, hence applying (Eq. 7) to the basic routing problem introduced in Section 4.2.2.5. For this simple setting, we explain notation and illustrate that capturing risk-based adaptive decisions is a complex task. We will also look again at different risk measures and discuss their advantages and disadvantages.

The example network consists of the nodes $N = \{s_0, s_1, s_2\}$ and the arcs A provided by the orange, green and pink line of the PT system as shown in Figure 51.

The probability distribution P describes the probability of on-time or delayed arrival of each of the pairwise independent lines and is assumed to be known in advance. It is provided in (Eq. 4). Each line has two possible realizations: it can either be punctual or delayed, which in the following will be denoted by p or d, respectively. The respective delay D depends however on the line and is given in (Eq. 4). Derived from the different combinations of these realizations for the three lines, there is a total of 8 possible scenarios of how the whole PT system will be realized. In line with the notation from (Gao & Chabini, 2006), we denote these by $P = \{v_1, \dots, v_8\}$ and provide a list in Table 8.

Table 8: List of possible scenarios including the state of each line, the respective delay in minutes and the probability of the scenario

Scenario	Realization of line			Delay (in minutes)			Probability of Scenario
	Orange	Green	Pink	D_o	D_g	D_p	
	P	P	P	0	0	0	0,4275
	P	D	P	0	10	0	0,0475
	P	P	D	0	0	30	0,4275
	D	P	P	5	0	0	0,0225
	P	D	D	0	10	30	0,0475
	D	P	D	5	0	30	0,0225
	D	D	P	5	10	0	0,0025
	D	D	D	5	10	30	0,0025

Depending on the realization of the individual lines, transfers between different lines might be possible/desirable or not. Based on this, passengers might choose different paths according to their preference. Thus, the policies, which are functions $\mu_i: P \to Paths$, will have different realized paths depending on the scenario. The different outcomes of the policies μ_1, μ_2, μ_3 for the scenarios are described in Table 9. The notation (l_1, l_2) consists of the two parts of the trip, where $l_1, l_2 \in \{o, g, p\}$ denote how one travels to node s_1 or s_2, respectively.

Finally, we need to introduce a notation for time. As we are not interested in specific departure or travel times, but simply in the deviation from the originally planned arrival, we reduce to the following notation. At station s_0, we only consider a default time t_0. At stations s_1 and s_2, we denote the time by l_k, where $l \in \{o, p, g\}$ and $k \in \{p, d\}$. Thus, the time l_k at a station is denoted as the time of arrival by a specific line and whether it is on-time or delayed. At station s_1, the possible times are in subsequent order $\{o_p, o_d, g_p, g_d\}$ and at s_2 it is $\{p_p, o_p, o_d, p_d\}$.

Table 9: List of possible scenarios, the realized paths for each policy, the arrival time at the destination nodes and the respective delay (in comparison to respective planned path r_i)

Scenario	Realized path			Arrival time at s_2			Delay (in minutes)		
	μ_1	μ_2	μ_3	μ_1	μ_2	μ_3	D_1	D_2	D_3
v_1	oo	op	go	o_p	p_p	o_p	0	0	0
v_2	oo	op	ox	o_p	p_p	x	0	0	60
v_3	oo	oo	go	o_p	o_p	o_p	0	15	0
v_4	oo	oo	go	o_d	o_d	o_d	5	20	5
v_5	oo	oo	op	o_p	o_p	p_d	0	15	15
v_6	oo	oo	go	o_d	o_d	o_d	5	20	5
v_7	oo	oo	go	o_d	o_d	o_d	5	20	5
v_8	oo	oo	go	o_d	o_d	o_d	5	20	5

The single destination node in this example network is s_2 and on their way to the destination, a passenger encounters different states on their trip, depending on the network realization and the decisions of the passenger. In general, the current state of a passenger $x = (s, t, I)$ consists of the the node, time and available information. Thus, in this case there is a single initial state $x_0 = (s_0, t_0, P)$. The initial state is composed of the origin node s_0, the default starting time t_0 and the available information, which consists of the knowledge about the probability distributions about the delays of all lines P. We assume that all departures at s_0 are on-time and the realization of the PT system happens upon arrival at station s_1, hence from that point onward the passenger knows the scenario $v \in P$. Thus, for stations s_1 and s_2, the current state of a passenger is of the form (s, l_k, v), i.e. consisting of the station s, the time of arrival l_k and the complete information about the PT system in the form of the scenario v. A state chain (x_0, x_1, x_2) is a sequence of states a passenger encounters during their trip and it depends on the chosen policy and the realized scenario. The set of possible state chains for a policy μ is denoted by $M(\mu)$ (and does not depend on the choice of the initial state x_0 as there is only one).

A passenger has to make a boarding decision at station s_0 based on the current and only initial state x_0. As the realization of the PT system happens upon arrival at station s_1 and at this point the delay of each line and possible transfers are known completely, the passenger has the option to take recourse action based on their preferences at station s_1.

The passenger decides for one of the three policies μ_1, μ_2, μ_3, which provide available strategies to move through the network from stations s_0 to s_2. As we assume that the passenger is only interested in the risk of a policy, i.e. evaluating the deviation from schedule, a reference path and arrival time is needed to compare the actual realization to. For each policy, this reference path p_i was implicitly chosen as the path with optimal outcome of the respective policy, corresponding to the scheduled routes R . That is, $p_i = r_i$ for each policy μ_i, $i = 1,2,3$ (cf. Section "Example Discussion). Since there is only one initial state x_0, for the optimization problem (Eq.7), the passenger solves to find the policy that minimized the risk, which reduces to

$$\mu^* = \arg\min_{i \in \{1,2,3\}} \quad \rho(D_i, r_i) \qquad (8)$$

Therefore, the optimal policy indeed only depends on how the risk measure ρ evaluate the risk of the policy μ_i in comparison to the planned path r_i. As we aim to evaluate only delays, we can reduce the notation even further. Recall that the delay of policy μ_i is given by $D_i = t_2 - t_{r_i}$, where t_2 denotes the time of the final state x_2, i.e. the actual time of arrival, and t_{r_i} is the scheduled time of arrival of the reference path r_i. While t_{r_i} is deterministic, t_2 (and therefore also D_i) depends on the state chain $(x_0, x_1, x_2) \in M(\mu_i)$, and is therefore random. Hence, determining the optimal policy based on minimizing the risk of delay is done by

$$\mu^* = \arg\min_{i \in \{1,2,3\}} \quad \rho(D_i) \qquad (9)$$

Based on this, we are now able to discuss the choice of risk measure ρ that is suitable in this example and for the preferences of different passengers. For this, we look at some common and simple examples of risk measures: the expected delay, the Value at Risk and the Conditional Value at Risk.

4.2.2.7 Expected Delay

The most basic choice of how to evaluate the deviation from plan is to determine the expected delay. By doing this, the optimal policy is derived by solving

$$\mu^* = \arg\min_{i \in \{1,2,3\}} \quad E_{(x_0,x_1,x_2) \in M(\mu_i)}[D_i] \qquad (10)$$

Considering that the time of arrival depends on the scenario v and policy μ_i, i.e. $t_2(\mu_i, v)$, we can rewrite the expected delay as follows

$$E_{\{(x_0,x_1,x_2) \in M(\mu_i)\}}[D_i] \qquad (11)$$

$$= \sum_{\{(x_0,x_1,x_2) \in M(\mu_i)} \left(t_2 - t_{r_i}\right) P(x_0, x_1, x_2)$$

$$= \sum_{\{v \in P} \left(t_2(\mu_i, v) - t_{r_i}\right) P(v)$$

The expected delay for the three given policies is then computed as

$E[D_1] = 0.25,\ E[D_2] = 8.125,\ E[D_3] = 3.8125.$

Thus, the deterministic policy μ_1 is considered least risky in terms of expected delay. The adaptive policies have higher risk as they include the possibility of missing a transfer. While this increases the risk for μ_2 significantly, the fact that with μ_3 one might not be able to successfully conclude their trip at station s_2 in time (for scenario v_2) does not seem to impact the risk properly.

With this example, it becomes also apparent that the choice of reference path p is important. In reality, policy μ_2 might be considered an improved version of policy μ_1, as the optimal outcome of μ_2 is shorter travel time and earlier arrival and the worse outcomes are just equal to $\mu_1 = r_1$. However, since the reference path of μ_2 is not r_1, the risk of the two policies is not compared directly and cannot be compared intuitively without further explanation. This shows that the choice of reference path could also be improved in this example.

Even though expected delay might not reflect all aspects of passengers' risk preferences, its simplicity has computational advantages. Due to the linearity of the mean one can utilize dynamic programming to solve (Eq. 9). While this is clearly not necessary in this simplified one-stage stochastic programming problem, it is useful if more stages of decision making are involved. We refer to the optimality conditions discussed in (Gao & Chabini, 2006) for this, as (Eq. 9) can be rewritten in the corresponding form.

4.2.2.8 (Conditional) Value at Risk

Another intuitive choice of measuring risk is by determining the delay that will not be exceed by a certain probability. The formal definition of the Value at Risk at level $\alpha \in [0,1]$ is

$$V@R_{1-\alpha}(D_i) = \inf\{m \in R | P(D_i \leq m) \geq 1 - \alpha.\} \qquad (12)$$

Recalling the results from (Eq. 5), we again observe that V@R has a similar flaw as the expected delay, as it assigns the same risk to μ_2 as to μ_3, which has the chance of not being completed, might not reflect passenger's risk preference. The common alternative is the Conditional Value at Risk, formally defined as

$$CV@R_{\alpha}(D_i) = E[\, D_i \mid D_i \geq VR_{\alpha}(D_i)\,]. \qquad (13)$$

As computed in (6.3), μ_3 has indeed a significantly higher risk, as CV@R emphasized tail-events. In addition to serving this common risk preference, CV@R is also a convex risk measure, which might be a computational advantage for solving the optimization problem, especially in comparison to V@R which is not convex.

This might indeed become necessary because solving (6.6) with dynamic programming as for e.g. done for the expected delay is only possible if the risk measure ρ is additive. In reality, this is rarely the case as people tend to overestimate small risks and underestimate big ones (Kahneman & Tversky, 1979). Moreover, this is also congruent with the consequences of travel time variations in PT: Even small delay with one line may cause missed transfer and thus a large delay for a passenger. Therefore, the overall risk-minimizing route might not be optimal at every stage. This inherent nature of risk thus calls for alternative approaches in solving (Eq. 9).

4.2.3 Results and Outlook

In this research we tackle the problem of adaptive routing based on minimizing the risk to provide a more accurate description of passenger behaviour in a PT network. The two main difficulties we take into account for approaching this problem are

1. Modelling adaptive route choice in a stochastic time-dependent PT system,
2. Defining risk and the choice of a suitable risk measure.

In this work, we provided a model that addresses the first point. We defined a STD network and routing policies as an adaptive extension for deterministic path choice. Based on this, (Eq. 7) described the optimization problem to derive the optimal routing policy to minimize risk. This model is kept very general, which allows it to be applied to a variety of different settings. But due to the generality, it is less intuitive and applicable for concrete examples. As the second goal was however to analyse risk measures, the generality of the model allows it to take a broad view at them, define risk in a general manner and study the properties and suitability of different examples without restricting ourselves to a specific setting.

We applied the model to a toy network to show how it can be applied. Within this example, we applied some examples of risk measures to show their differences and discuss their computational and interpretational advantages. This discussion needs to be extended with looking into the properties of risk measures in more detail. It will also be helpful to consider a more advanced example to really highlight the differences of adaptive routing in comparison to other ways of path choice, and its effect also on the risk of the route.

4.3 Optimisation of Demand Responsive Transport

4.3.1 Intent of demand responsive optimisation

This component addresses the problem of precisely estimating the number and type of required vehicles for pickup and delivery of passengers for some time in the future. It builds upon the Prediction Models for Demand Responsive Transport described in the section above, using its predictions as input data. In addition to demand predictions, we are also utilizing historical reservation data. The output consists of the total number of required vehicles across all routes, along with detailed information for specific routes of interest.

To generate the desired output, we first take the prediction data and simulate reservation data. Each reservation includes attributes such as start location, end location, and pickup time. However, certain features cannot be directly inferred from demand predictions, so we derive them from historical data. Specifically, we use historical data to create three distributions related to:

- Number of passengers per reservation – The typical group size within a single booking.
- Reservation type – Travel preferences that dictate whether a passenger prefers a private vehicle or is willing to share a ride.
- Pickup/delivery time window – Defines the flexibility of pickup and drop-off times, whether we are looking at a fixed time, or a more variable interval, which impacts pricing and passenger experience.

Demand predictions provide hourly passenger estimates for different days. Using these predictions, our component simulates reservation data by sampling values from the historical

distributions of the previously mentioned parameters. The generated reservation data is structured according to the required format for the Route Optimization Engine.

To ensure robust route planning, we generate 100 independent reservation samples for each day and we submit each sample as an independent job in the Route Optimization Engine. This engine is engaged at two key stages:

1) Preprocessing – Used to determine travel times between locations based on specific departure times. This helps define feasible delivery intervals for each pickup window.

2) Optimization – Each reservation sample is processed as an independent job, where passengers are assigned to vehicles, and routes are optimized to ensure efficiency.

As a result, the system calculates the total number of vehicles required per day, as well as for each specific route for each sample. Once the optimization process is complete, the final step involves aggregating and analysing the results obtained from all 100 independent samples. This analysis focuses on identifying patterns and trends in fleet requirements by examining statistical measures such as mean, standard deviation, and overall distribution. By assessing these variations, we gain a deeper understanding of how demand fluctuates across different days and routes. Additionally, the results are validated against historical data to ensure consistency, accuracy, and reliability in the estimated vehicle requirements. This validation process helps refine the approach and enhances confidence in the system's predictive and optimization capabilities.

Regarding the results, we conducted both a global analysis and a more detailed, route-level analysis to uncover patterns within individual routes.

We began by examining the distribution of the number of passengers and vehicles, as well as the correlation between these two variables. It's important to note that passengers are first grouped into visits using statistical sampling methods, and then further grouped into vehicles by the routing engine.

The visualisations reveal a strong linear correlation between the number of passengers and the number of vehicles, which is expected. However, there is some variance, suggesting that fluctuations in passenger count do not always directly translate to proportional changes in the number of vehicles. This can be attributed to differences in reservation types and the number of people per reservation, which influence how passengers are grouped into vehicles.

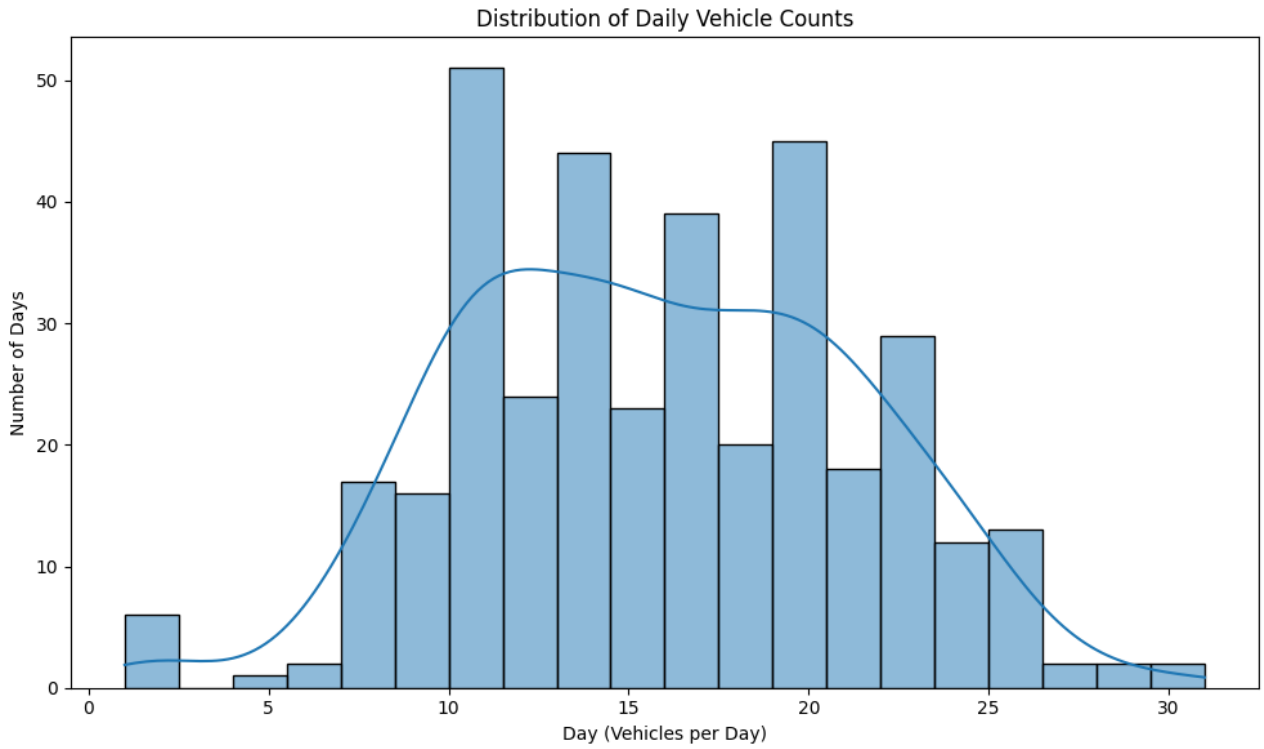

Figure 52: Daily vehicles distribution

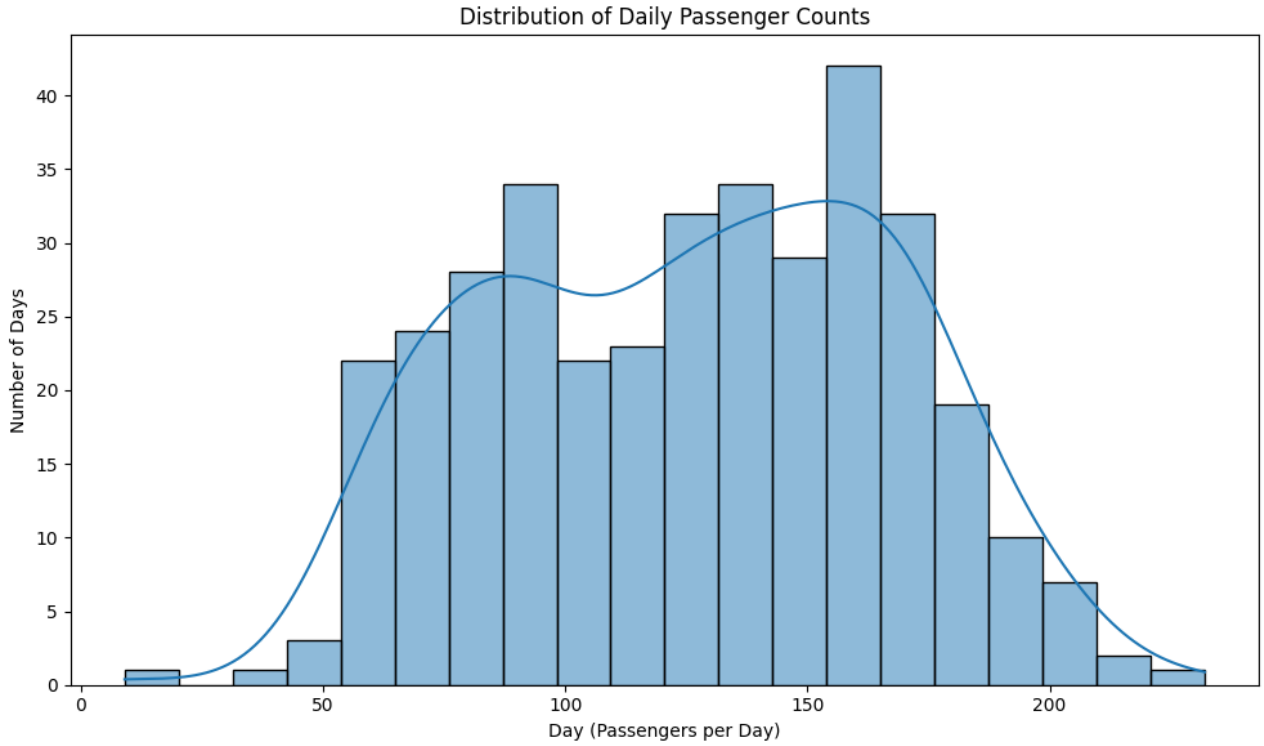

Figure 53: Daily passengers distribution

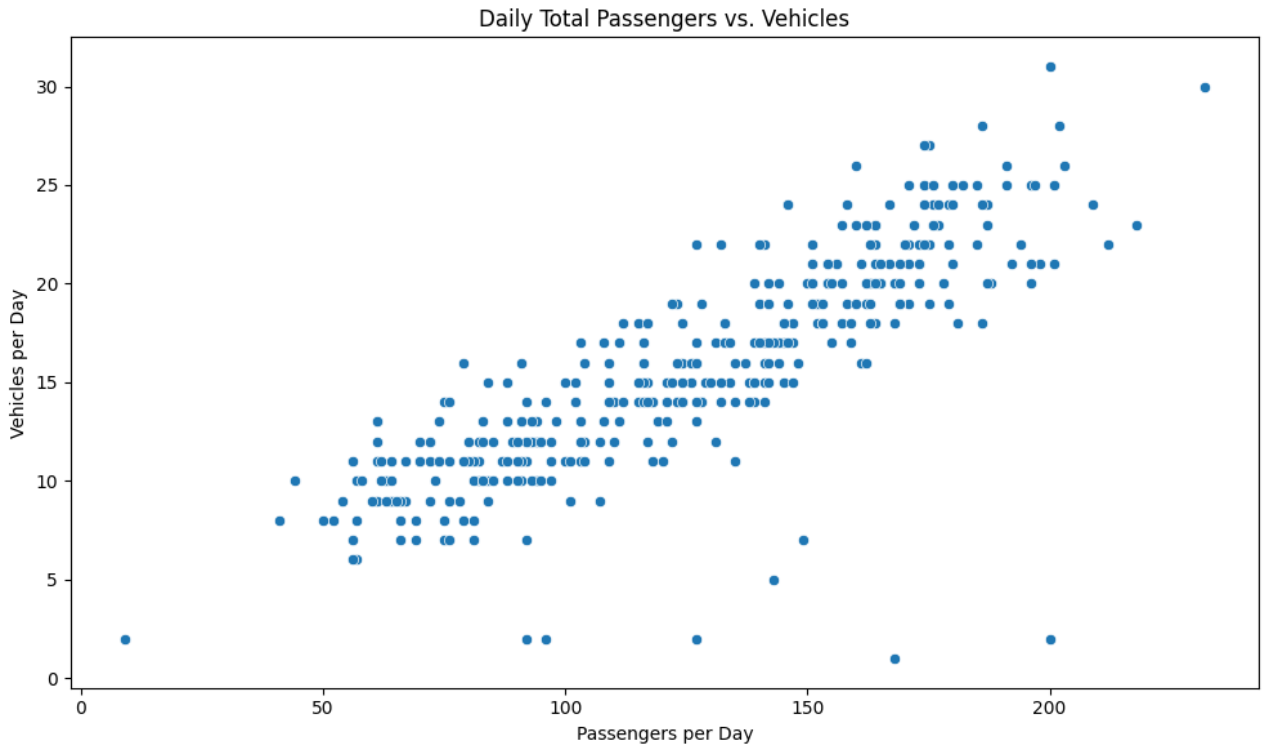

Figure 54: Total passengers vs vehicles

Additionally, we analysed the correlation between different routes and examined the distribution of vehicles and passengers on a per-route basis. This allowed us to assess how consistently the number of passengers aligns with the number of vehicles across individual routes, revealing route-specific patterns and variations in demand and vehicle allocation.

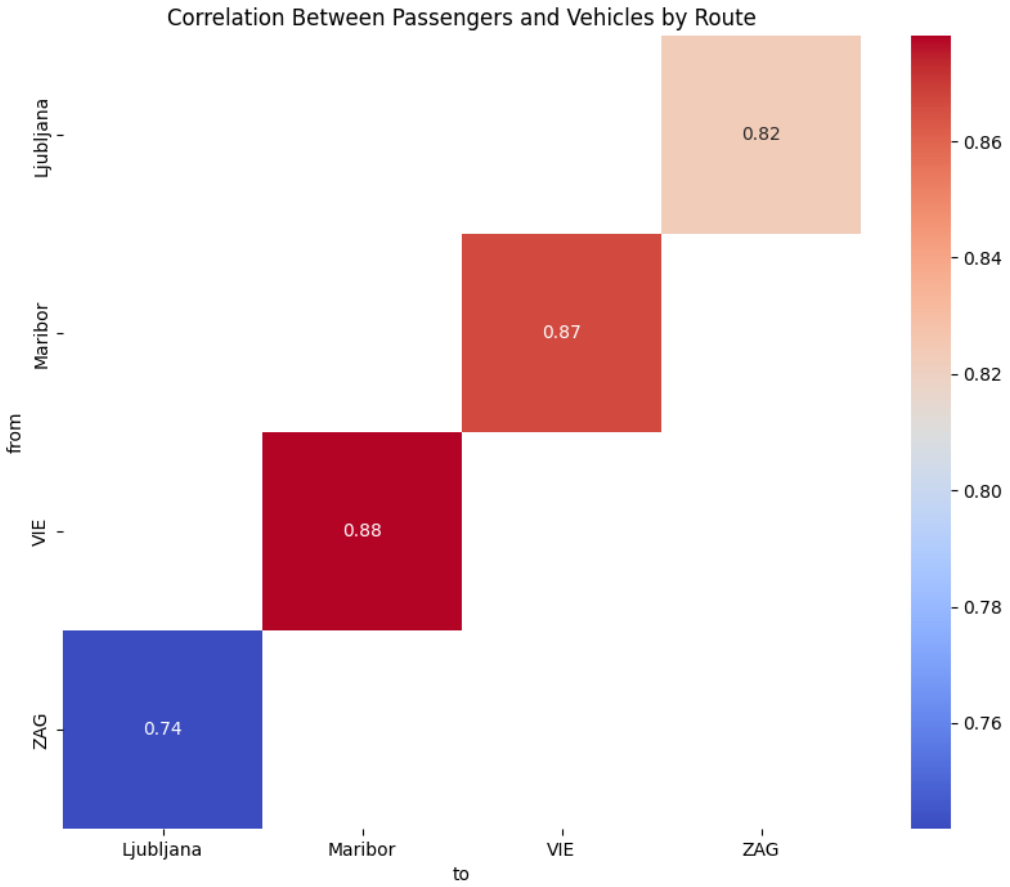

Figure 55: Correlation heatmap between passengers and vehicles by route

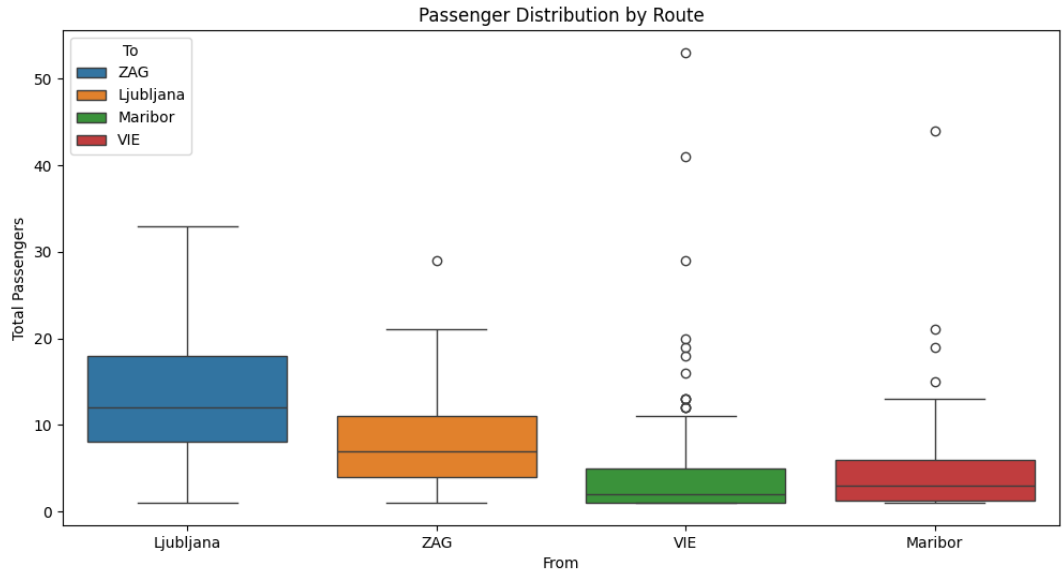

Figure 56: Boxplot of passenger distribution by route

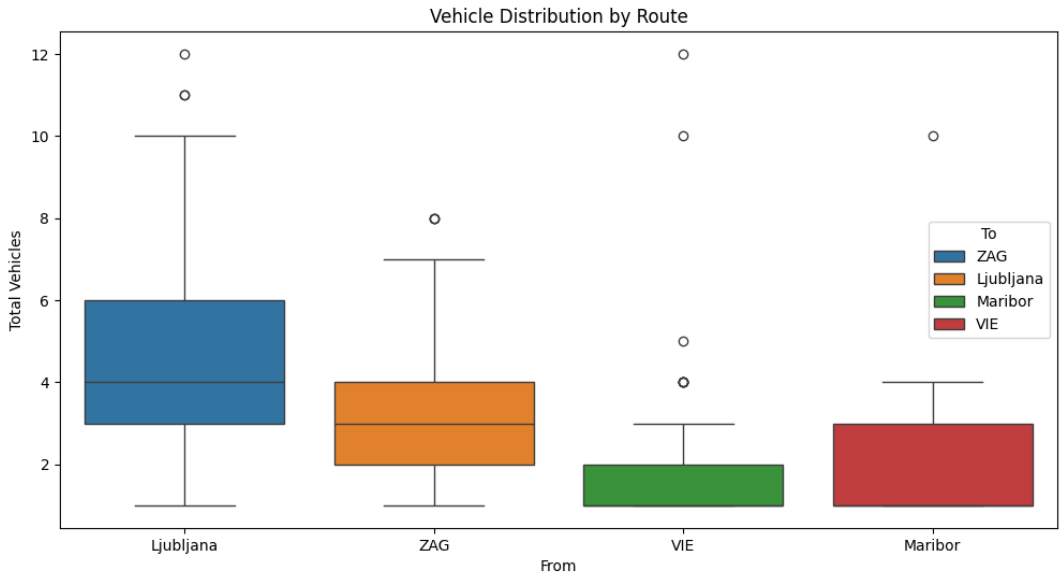

Figure 57: Boxplot of vehicle distribution by route

The methodology of the component is represented in the image below.

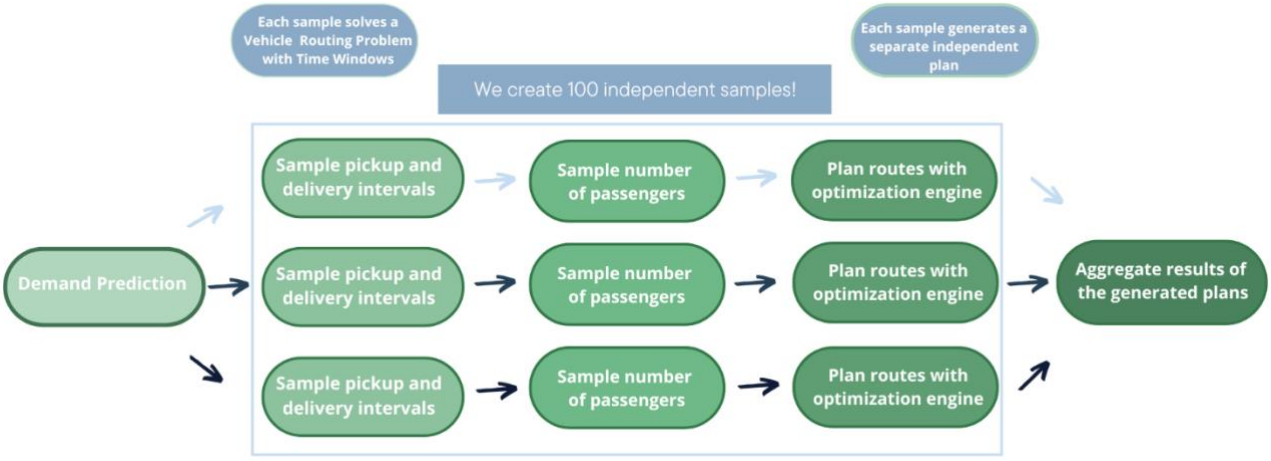

Figure 58: Continuous Planning Methodology

5. ANOMALY DETECTION

5.1 Anomaly detection in traffic patterns

The Anomaly Detection component within the CONDUCTOR ecosystem is designed to identify irregularities in traffic behaviour, offering timely insights into deviations from expected operational norms. This functionality plays a pivotal role in the broader context of adaptive traffic and fleet management, contributing to the system's responsiveness, resilience, and overall efficiency.

As outlined in Deliverable D3.3, anomalies in traffic data refer to patterns that deviate significantly from historical or contextually expected behaviour. These anomalies can be the result of unplanned events such as traffic incidents, weather disturbances, or atypical demand surges, as well as scheduled disruptions like roadworks or major public gatherings. A comprehensive conceptual overview and classification of anomaly types—including point, contextual, and collective anomalies—was presented in Section 3.1.1 of D3.3, laying the theoretical foundation for the detection approaches that follow.

In this final iteration, the work conducted under Task 3.5 has progressed from preliminary experimentation to a more mature and operational version of the anomaly detection module. Building upon the methodologies introduced in the initial implementation (D3.3), the current chapter describes the finalized specifications, implementation updates, and methodological refinements based on simulation insights and technical validation activities conducted throughout the project timeline.

The anomaly detection functionality remains centred on a macroscopic approach, with the primary objective of identifying periods where the collective performance of the monitored network exhibits unusual behaviour. Rather than detecting micro-level events such as isolated vehicle breakdowns or individual lane closures, the system aims to uncover larger-scale deviations that signify network-wide inefficiencies or disruptions.

In the sections that follow, we detail the enhancements introduced in the anomaly detection workflow, the enriched datasets used for training and evaluation, the refined modelling techniques, and the integration pathway towards the overall CONDUCTOR architecture. These updates reflect both the lessons learned from the initial deployment and the progressive alignment with real-world deployment scenarios provided by the Athens use case.

5.1.1 Architecture of the Proposed Solution

The finalized architecture of the Anomaly Detection component has been structured to ensure operational efficiency, scalability, and seamless integration within the broader CONDUCTOR framework, with specific tailoring for the Slovenian pilot. The design reflects the modular nature of the service, enabling flexibility across diverse deployment environments while ensuring robustness in real-time operation. Data ingestion is initiated through the interface provided by Jozef Stefan Institute (JSI), which acts as a proxy service to the historical traffic datasets maintained by National Access Point (NAP) Slovenia. This layer facilitates structured and secure access to the required traffic records, forming the foundation of the anomaly detection pipeline. Upon retrieval, the incoming data undergo a comprehensive preprocessing workflow, which includes systematic data cleaning procedures, Exploratory Data Analysis (EDA), and imputation strategies. These steps, as elaborated in earlier stages of the project, are critical in ensuring data reliability and consistency, both of which are essential for the effective training and deployment of anomaly detection models.

Following preprocessing, the dataset is partitioned into training, validation, and test subsets to support methodical model development. A range of machine learning models is subsequently trained and validated using these splits. Hyperparameter tuning and iterative refinement are carried out

based on key performance metrics, with the highest-performing configurations evaluated on the holdout test set to ensure robustness and generalizability to unseen data. Upon final selection, the optimized model is serialized and embedded within a dedicated deployment script. This script is integrated into a FastAPI application that exposes an HTTP endpoint, enabling real-time inference capabilities. The deployed service is designed for continuous operation, facilitated by an automated scheduling mechanism (Command Run On Notice - CRON job) that periodically fetches current traffic data at fixed intervals for analysis. The system's response comprises of the original dataset enriched with an additional output column, denoting the anomaly likelihood score for each record. In instances where anomalies are detected, the service also generates a targeted notification message, explicitly flagging the presence of abnormal traffic conditions. This structured output is intended to inform downstream traffic management components, thereby enhancing situational awareness and enabling prompt operational responses. Overall, the proposed architecture ensures that the anomaly detection service remains technically sound, easily extensible, and deployment-ready for real-world scenarios. It also retains the flexibility required to accommodate evolving pilot-specific needs and potential future integrations with alternative data providers.

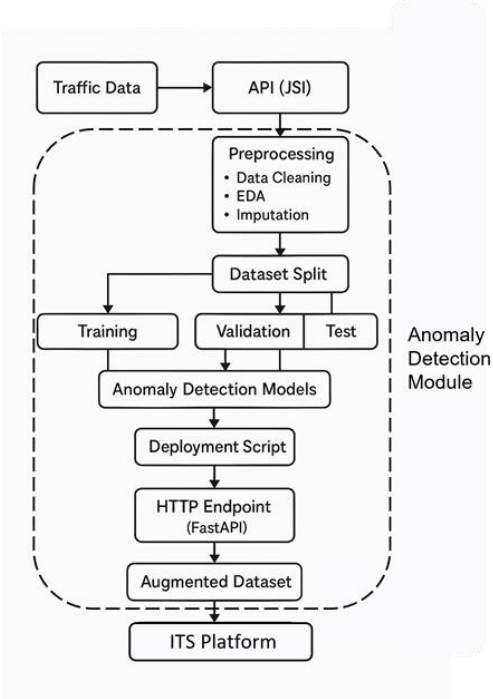

Figure 59: Continuous Planning Methodology

5.1.2 Data Enhancements and Preprocessing Workflow

In alignment with the evolving focus of the CONDUCTOR project and the refinement of pilot-specific requirements, the anomaly detection service was methodologically tailored to accommodate the data landscape of the Slovenian use case. The primary source of traffic data originates from NAP Slovenia; however, due to the platform's inherent limitations in terms of long-term data retention, an alternative data access strategy was employed. Specifically, historical data were retrieved through an API maintained by the JSI, which offers persistent archival access to traffic records of the region. This integration ensured the availability of adequate historical context necessary for model training and validation, albeit within a framework constrained by technical considerations. In particular, API rate-limiting policies necessitated a segmented data retrieval strategy, wherein data were ingested incrementally across multiple batches. Through this process, a representative training dataset was compiled, encompassing a period of approximately three months. Upon ingestion, the dataset was

received in its raw form, characterized by inconsistencies, missing values, and noise—a common scenario in traffic-related datasets. To address these challenges, a structured Exploratory Data Analysis (EDA) phase was executed as the initial stage in the processing pipeline. This phase involved the temporal alignment of timestamped records, detection and flagging of statistical outliers, and the application of a comprehensive suite of data cleaning procedures.

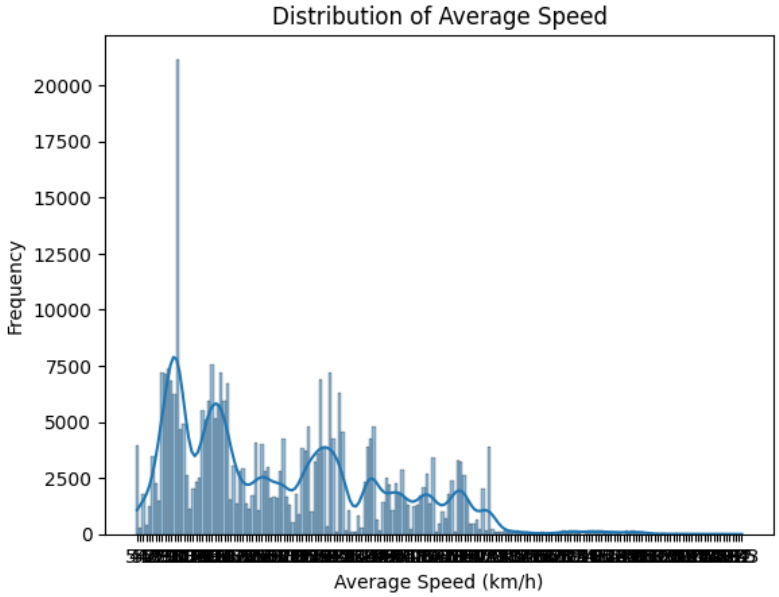

Figure 60: Analysis on the frequency distribution of average speeds (EDA)

In the context of missing data, a tiered imputation strategy was adopted, informed by the specific nature and duration of data gaps. Short-term discontinuities were addressed using forward and backward fill techniques, while longer or periodic sequences of null values were imputed using rolling time-window medians. In cases where preserving the inherent temporal seasonality of the traffic data was deemed critical, decomposition-based approaches were also incorporated to retain daily and weekly periodicity patterns. Concurrently, the dataset was augmented with a set of contextual features aimed at enriching the input space of the anomaly detection models. These included indicators such as the day of the week, and time-of-day variables, enabling the models to capture behavioural aspects associated with typical versus atypical traffic conditions. Standard normalization procedures were applied across numerical attributes to ensure comparability and model convergence. This preprocessing workflow established a robust foundation for the subsequent modelling phases, ensuring data quality, consistency, and contextual integrity. By doing so, it strengthened the capacity of the anomaly detection framework to generalize effectively and respond to the dynamic conditions present in real-world traffic systems.

```
{
    "id": "0598-21",
    "dateUpdated": "2024-03-05T23:55:00",
    "countersLocation": "0598",
    "countersLocationDesc": "Dobrova",
    "countersRoadDesc": "R3-641",
    "countersSection": "1369",
    "countersDirection": "21",
    "countersDirectionDesc": "Ljubljana - Dobrova",
    "countersLaneDesc": "",
    "countersGeolocationX": "455557",
    "countersGeolocationY": "101058",
    "countersSpeedLimit": "50",
    "countersDate": "2024-03-06",
    "countersTime": "0001-01-01 00:55:00+00 BC",
    "countersNumberVehicles": "0",
    "countersAverageSpeed": "0",
    "countersGapBetweenVehicles": "999.9",
    "countersStatus": "6",
    "countersStatusDesc": "No traffic"
},
{
    "id": "0692-11",
    "dateUpdated": "2024-03-05T23:55:00",
    "countersLocation": "0692",
    "countersLocationDesc": "Zadvor",
    "countersRoadDesc": "R3-645",
    "countersSection": "1188",
    "countersDirection": "11",
    "countersDirectionDesc": "Ljubljana - Zadvor",
    "countersLaneDesc": "",
    "countersGeolocationX": "468775",
    "countersGeolocationY": "99786",
    "countersSpeedLimit": "50",
    "countersDate": "2024-03-06",
    "countersTime": "0001-01-01 00:55:00+00 BC",
    "countersNumberVehicles": "12",
    "countersAverageSpeed": "51",
    "countersGapBetweenVehicles": "674.4",
    "countersStatus": "1",
    "countersStatusDesc": "Normal traffic"
```


Figure 4 Slovenian Traffic Data from JSI API

5.1.3 Model Refinement and Ensemble Approaches

As presented in Deliverable D3.3, the initial implementation of the anomaly detection module was grounded in a series of foundational models, including One-Class Support Vector Machines (**OneClassSVM**), **Isolation Forest**, Local Outlier Factor (**LOF**), and Long Short-Term Memory (**LSTM**) neural networks. These models served as a valuable starting point, enabling the team to establish baseline performance metrics and to explore the feasibility of different detection paradigms under constrained, simulation-based scenarios. While the preliminary results demonstrated notable potential, especially within controlled environments, several key limitations emerged during subsequent phases of testing and validation. Among the most prominent challenges were those associated with class imbalance, overfitting to sparsely distributed anomalies, and a lack of consistent generalization across varied temporal and spatial contexts.

These constraints underscored the necessity of adopting a more resilient modelling strategy capable of mitigating individual model weaknesses and increasing overall robustness. In response to these findings, the final implementation introduces an ensemble-based [see Figure 61] anomaly detection

architecture. The ensemble framework integrates the outputs of multiple heterogeneous base models through both majority voting and weighted averaging schemes. These combination strategies were selected for their capacity to improve detection consistency while reducing the sensitivity to noise, model-specific errors, and overfitting tendencies. The adoption of ensemble learning represents a significant advancement over the initial standalone model architecture, allowing for more stable performance across varying traffic conditions. Moreover, it enhances the interpretability and operational reliability of the system by aggregating evidence from multiple independent perspectives, thereby reducing the likelihood of false alarms and increasing confidence in anomaly identification. This shift toward ensemble approaches reflects the project's iterative, evidence-driven methodology, and aligns with the overarching goal of deploying a technically sound, adaptable, and high-performing anomaly detection service within the CONDUCTOR framework.

5.1.4 Majority Voting

The first ensemble strategy employed in the final anomaly detection framework is a majority voting mechanism, aimed at enhancing classification robustness by enforcing consensus among multiple base detectors. Let $M_i \in \mathcal{M}$, where $\mathcal{M}$ denotes the set of participating models in the ensemble. Each model issues a binary decision $v_i \in \{0,1\}$ for a given sample x, with $v_i = 1$ indicating the presence of an anomaly and $v_i = 0$ otherwise.

The collective decision function $V(x)$ is then defined as:

$$V(x) = \begin{cases} 1, & \text{if} \sum_{i=1}^{N} v_i \geq \left\lceil \frac{N}{2} \right\rceil \\ 0, & \text{otherwise} \end{cases}$$

where $N = |\mathcal{M}|$ is the total number of models in the ensemble.

This scheme requires a strict majority in favour of the anomaly label in order for a sample to be classified as such. By design, this approach mitigates the impact of individual false positives and promotes resilience in the presence of occasional misclassifications by isolated models. As such, majority voting enhances overall decision reliability, particularly in high-noise environments.

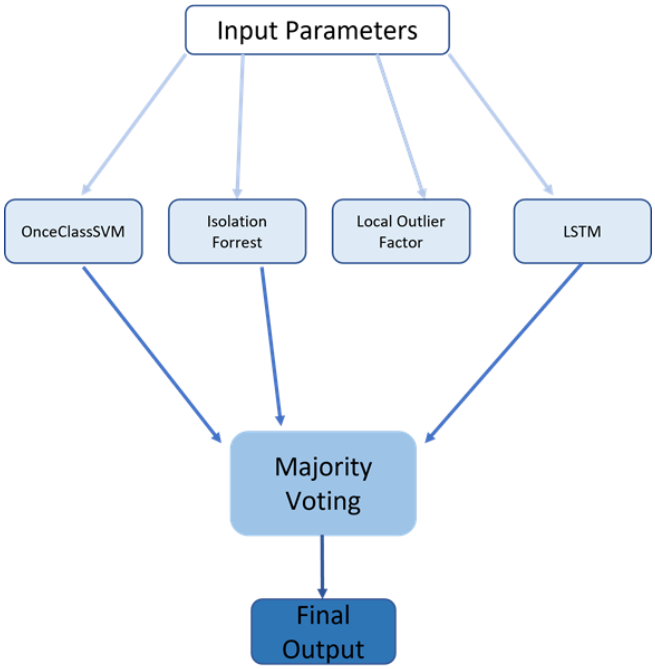

Figure 61: Majority Voting Schema of Anomaly Detection module

Table 10: Evaluation metrics of base models and the majority voting ensemble for the anomaly class

Model	Precision	Recall	F1-Score
OneClassSVM	0.74	0.63	0.69
Isolation Forest	0.52	0.93	0.59
Local Outlier Factor (LOF)	0.7	0.3	0.46
LSTM	0.75	0.61	0.57
Majority Voting Ensemble	**0.78**	**0.76**	**0.81**

The majority voting mechanism outperforms the individual base models by balancing recall and precision. The ensemble achieves a higher F1-score by mitigating extreme behaviours (e.g., Isolation Forest's high recall but low precision) and improving overall detection consistency.

5.1.4.1 Weighted Averaging

To complement the binary nature of majority voting and enable more nuanced decision-making, a weighted averaging scheme was also incorporated into the ensemble pipeline. This technique leverages the probabilistic outputs of individual models, thereby integrating model-specific confidence levels into the anomaly detection process.

Each model M_i produces a continuous anomaly score $s_i(x) \in [0,1]$, interpreted as the estimated probability of an anomaly for a given sample x. These scores are combined into a single aggregated value $S(x)$, defined as:

$$S(x) = \sum_{i=1}^{N} w_i \cdot s_i(x)$$

subject to the constraints:

$$\sum_{i=1}^{N} w_i = 1, \quad w_i \geq 0$$

The model weights w_i are determined via cross-validation on a held-out validation set, optimized to maximize the F1-score while simultaneously minimizing the rate of false positives.

Once aggregated, the final score $S(x)$ is subjected to a classification threshold θ, resulting in the predicted label $\hat{y}(x)$ as follows:

$$\hat{y}(x) = \begin{cases} 1, & \text{if } S(x) \geq \theta \\ 0, & \text{otherwise} \end{cases}$$

This fusion method allows the system to assign greater influence to models that exhibit stronger discriminative capacity, while still preserving ensemble diversity. Furthermore, the use of continuous scoring provides a transparent basis for threshold tuning and interpretability, which are critical in operational settings.Together, these ensemble strategies deliver complementary advantages. The majority voting scheme ensures robustness and stability in binary classification scenarios, while weighted averaging introduces probabilistic reasoning and performance-aware integration of model outputs. Collectively, they contribute to a more reliable, adaptable, and explainable anomaly detection framework, capable of managing the heterogeneity and temporal variability inherent in real-world traffic data environments.

Table 11: Weighted averaging ensemble performance with weights derived via F1-optimized cross-validation

Model	Weight	Precision	Recall	F1-Score
OneClassSVM	0.35	0.74	0.63	0.69
Isolation Forest	0.15	0.52	0.93	0.59
LOF	0.2	0.7	0.3	0.46
LSTM	0.3	0.75	0.61	0.57
Weighted Averaging Ensemble	—	**0.83**	**0.78**	**0.84**

By assigning greater influence on models with stronger discriminative capacity (e.g., LSTM, OneClassSVM), the weighted ensemble achieves improved balance and slightly outperforms the majority voting scheme in terms of F1-score. This method also allows threshold flexibility through score tuning.

5.1.5 Deployment and Evaluation

A major enhancement in this final version of the anomaly detection module is its technical readiness for real-time deployment. The codebase was modularized to support maintainability and integration within the broader CONDUCTOR system, while targeted optimizations improved inference speed. By incorporating TensorFlow Lite and applying model quantization techniques, the module now

achieves low-latency execution, making it suitable for continuous monitoring scenarios where response time is critical. For evaluation purposes, both historical traffic data and synthetically labelled datasets were employed to benchmark performance. The models developed during this iteration consistently outperformed the baselines reported in D3.3, achieving F1-scores above 0.1 across multiple test sets. Among the most promising results was the performance of the LSTM-based ensemble, which demonstrated high accuracy—up to 88%—in detecting reduced-speed anomalies during off-peak hours, a scenario often difficult to capture due to limited variation in input signals.

It is important to highlight that anomalies represent a very small portion of the dataset, which presents a challenge typical of imbalanced classification problems. In such cases, common metrics like overall accuracy can give a false impression of model effectiveness. Therefore, the confusion matrix plays a central role in the evaluation, as it clearly illustrates how well the model performs in identifying true anomalies while avoiding false negatives. A correct classification of rare events is far more valuable than achieving high accuracy through consistent prediction of the majority (normal) class.

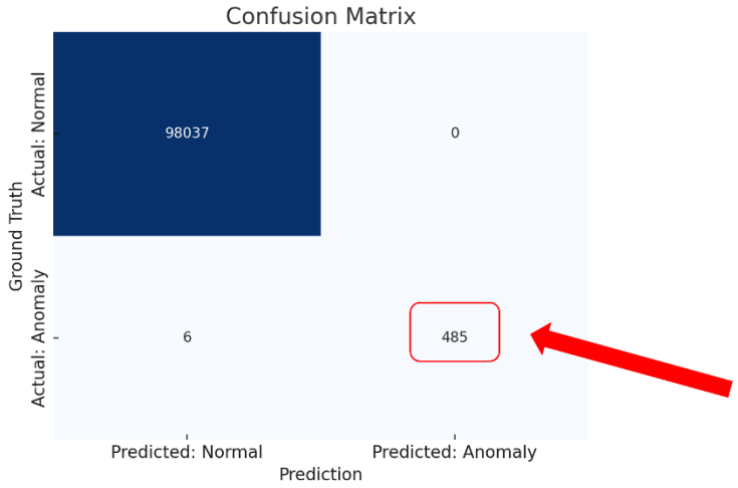

Figure 62: Confusion Matrix of the best performing combination

The confusion matrix presented above provides a detailed view of the anomaly detection model's final performance during evaluation. Out of a total of 98,528 observations, 98,037 were correctly classified as normal, with zero false positives, highlighting the system's ability to avoid unnecessary alerts under stable traffic conditions. More importantly, the matrix reveals that 485 true anomalies were successfully detected, while only 6 actual anomalies were missed. This result yields a true positive rate (recall) of 98.78% for the anomaly class, confirming the model's capacity to identify rare but critical deviations with high reliability. The absence of false positives also underscores the effectiveness of the ensemble approach in suppressing noise and limiting over-sensitivity—an aspect particularly relevant for real-time deployments, where false alarms can trigger unnecessary downstream reactions.

Of particular note is the high precision achieved for the anomaly class, as all predicted anomalies correspond to actual events. This balance between high precision and high recall demonstrates that the anomaly detection module is not only sensitive to abnormal patterns but also selective in raising alerts, a dual requirement for operational trustworthiness. The cell representing true positives (actual anomaly / predicted anomaly) is deliberately highlighted in red to emphasize its significance within the context of rare event detection. It represents the core utility of the module—its ability to correctly detect deviations in real-world traffic flow with minimal misclassification. In conclusion, the confusion matrix validates the module's readiness for integration into the

CONDUCTOR system, offering both accuracy and stability, and demonstrating its capacity to deliver high-confidence detections even in the presence of extreme class imbalance.

Moreover, during testing, it became evident that providing only a binary output—labelled as 0 (normal) or 1 (anomalous)—was not sufficient to support real-time decision-making processes. To improve the interpretability and control of the system, the binary label was extended with a probability score, derived from the underlying model confidence. This score, ranging from 0 to 1 (or equivalently, 0% to 100%), reflects the system's estimated likelihood that a given observation represents an anomaly. By incorporating this probabilistic output, the anomaly detection framework allows greater flexibility in threshold selection, depending on the operational context and tolerance for false alarms. This approach also supports more informed responses, as stakeholders can prioritize or defer action based on the assigned likelihood of abnormal behaviour.

With these enhancements, the anomaly detection module is now positioned as a deployable, high-performance component of the CONDUCTOR traffic management system. It is capable of operating under real-world constraints, while offering high sensitivity to traffic deviations and adaptability to the specific needs of pilot deployments.

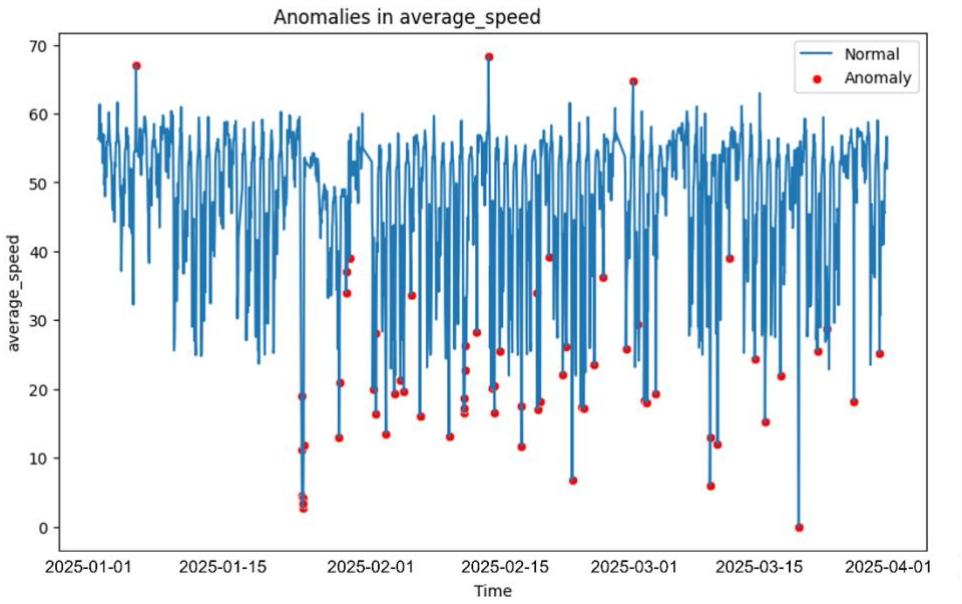

Figure 63: Sample Visualization of Anomalies on Ljubljana section

```
{
  "id": "0619-21",
  "dateUpdated": "2025-04-14T17:40:00",
  "countersLocation": "0619",
  "countersLocationDesc": "Stara Cerkev",
  "countersRoadDesc": "G2-106",
  "countersSection": "0263",
  "countersDirection": "21",
  "countersDirectionDesc": "Kočevje - Ljubljana",
  "countersLaneDesc": "",
  "countersGeolocationX": "487666",
  "countersGeolocationY": "58347",
  "countersSpeedLimit": "50",
  "countersDate": "2025-04-14",
  "countersTime": "0001-01-01 17:40:00+00 BC",
  "countersNumberVehicles": "41",
  "countersAverageSpeed": "12",
  "countersGapBetweenVehicles": "52.6",
  "countersStatus": "1",
  "anomalyScorePercentage": 84.7,
  "anomalyStatusDesc": "Anomaly suspected"
}
```


Figure 64: Sample Output of the Enhanced dataset

5.1.6 Technology Stack

To support the development, deployment, and operation of the anomaly detection service, a modern and modular technology stack was adopted. The core implementation is based on Python, leveraging a combination of machine learning libraries, time series analysis toolkits, and lightweight deployment frameworks.

Model development and data preprocessing tasks were performed using well-established Python libraries such as Scikit-learn, Pandas, and NumPy. To enhance the diversity and performance of the detection algorithms, specialized anomaly detection libraries like PyOD, STUMPY, and Kats were integrated into the pipeline. These tools provided advanced techniques for outlier detection, pattern discovery, and change point analysis in time series data.

For deployment, the trained models were encapsulated in a FastAPI application served by Uvicorn, offering high-performance and asynchronous request handling. Postman was employed during development for testing and verifying API endpoints. The service was containerized using Docker, ensuring reproducibility and compatibility across environments. A cron job mechanism was introduced to automate periodic inference calls with incoming traffic data.

This comprehensive and lightweight stack ensures the solution remains portable, maintainable, and easy to integrate within the broader CONDUCTOR ecosystem.

Table 12: Technology stack for anomaly detection in transport supply

Technology	Purpose
Python	Core programming language used for model development and data processing
Scikit-learn	Implementation of baseline ML models and preprocessing utilities

PyOD	Access to a broad suite of anomaly detection algorithms
STUMPY	Pattern discovery and anomaly detection in time series via Matrix Profile
Kats	Change point and anomaly detection in time series
FastAPI	Lightweight web framework for deploying the trained model as an API
Uvicorn	ASGI server used to serve the FastAPI application
Postman	Manual testing of API endpoints during development
Docker	Containerization of the service for reproducible deployment
Cron	Scheduled task automation for periodic inference requests
Pandas / NumPy	Data handling, manipulation, and numerical computations
Matplotlib / Seaborn	EDA and anomaly visualization tools

5.1.7 Lessons Learned and Outlook

One of the key insights emerging from this development cycle is the critical role of combining temporal depth with spatial granularity in anomaly detection for traffic monitoring. Initial experiments confirmed that models trained exclusively on data from single sensors often suffer from limited generalizability, failing to adapt across different spatial contexts or roadway segments. On the other hand, broader models developed at the road-segment level, while effective in identifying large-scale flow irregularities, occasionally overlook localised anomalies that manifest at a finer resolution. To address this, a hybrid modelling strategy was adopted, integrating both sensor-level precision and segment-level context. This approach proved effective in capturing not only routine fluctuations in traffic behaviour but also sudden disruptions, such as congestion spikes and abnormal flow reductions—thus offering a practical compromise between model sensitivity and stability.

Another key consideration encountered during the deployment process was the uneven availability of data across the Ljubljana–Trieste route. While the anomaly detection module performs reliably on the Slovenian side, benefiting from structured and accessible data sources, the Italian segment of the corridor remains unsupported due to the unavailability of traffic data in compatible formats. As a result, the current implementation remains geographically constrained to the Slovenian territory. Efforts are ongoing to explore potential solutions to this limitation, including collaborations for improved data access or the design of proxy estimation models based on adjacent cross-border segments. Looking forward, future development efforts will focus on expanding the contextual awareness of the module by incorporating exogenous variables such as weather conditions, real-time incident reports, and scheduled public events. The inclusion of such variables is expected to enhance the explanatory power of the models, enabling the system to distinguish between anomalies caused by external disruptions and those stemming from internal network dynamics.

In parallel, exploratory work has commenced on the development of a recovery time estimation framework, designed to infer the expected duration of abnormal traffic conditions based on historical anomaly patterns (but since it was optional and has not yielded final results it is not reported in this deliverable). By estimating section-specific recovery times, this feature could provide an added layer of decision support for operators, particularly in dynamic routing and resource allocation contexts. Finally, as the system matures and its predictive performance stabilizes, there is a clear trajectory toward extending its role beyond anomaly detection and into anomaly forecasting. Such a transition would mark a significant step forward, empowering stakeholders to implement proactive traffic management strategies grounded in reliable short-term predictions of network deviations.

5.2 Anomaly detection in transport demand

A proper characterisation of the demand mobility patterns enables the detection of anomalies, the identification of possible causes (such as weather conditions or especial events), and the assessment of their impact on the transport network. This analysis allows the classification of demand anomalies and the development of appropriate response plans. Moreover, as some of the factors (like weather conditions, or planned events such as football matches or demonstrations) are known in advance, certain anomalies can be anticipated and mitigated.

The objective of this development is to develop a time series model capable of accurately capturing and forecasting mobility demand patterns. Additionally, the model should be able to detect anomalies in the demand. The expected demand can be used for the strategic planning of the transport network, while anomaly detection enables analysis and classification based on underlying causes and impacts, facilitating anticipation and more efficient decision-making.

5.2.1 Data used

The data used for this development are:

- **Origin-Destination (OD) matrices generated by Nommon**. These matrices are generated from MND using the Nommon Mobility Insights solution. The matrices are segmented by trips' characteristics (time of the day, purpose, and distance) and travellers' characteristics (age, gender, residence place, and income).

The datasets needed to generate the OD matrices are:

- Activity and travel diaries generated from MND using Nommon proprietary algorithms.
- Spanish census data, provided by the Spanish National Statistics Institute.
- Land use information, provided by the Spanish National Geographic Information Centre.
- Transport supply data, provided by the Statistics Institute of the Community of Madrid and the Madrid Regional Transport Consortium.
- Travel surveys, provided by the Regional Transport Authority of Madrid.

5.2.2 Methodology

The algorithm developed consists in two main steps:

1. **Demand forecasting**: a ML time series forecasting algorithm forecasts the expected demand for a day based on a multi-dimensional time series analysis over historical demand information.
2. **Anomaly detection**: the expected demand obtained in the previous step is compared with the actual demand of the day to identify anomalies. For that, a two-step process is defined:

a. Validation of the expected demand based on the historical demand: to find potential anomalies, we must make sure that our reference demand values (i.e., the forecast demand) are accurate in the first place. For that, the expected demand is compared with the demand of the same day of the week of the previous weeks using a variation of Bollinger bands. Bollinger bands provide a criterion to define confidence intervals for the forecast demand. They consist of an *n*-period moving average, an upper bound (band) at *k* times an *n*-period standard deviation above the moving average, and a lower bound (band) at *k* times an *n*-period standard deviation below the moving average. Here, *k*, *n* are parameters that should be calibrated. This way, to validate, for instance, the forecast demand for a Tuesday, the average demand (denoted as *avg*) and standard deviation (denoted as *std*) of the previous *n* Tuesdays are computed to define the Bollinger band as the interval [avg - *k* std, avg + *k* std]. As sometimes the std is not enough to capture the demand variability, to make this interval more flexible to adapt to small changes or fluctuations in demand, we consider a variation of this formulation in the following way [(1- α) avg - *k* std, (1+ α) avg + *k* std]. If the forecast demand falls within this interval, it is considered a normal value. Otherwise, it is deemed abnormal, requiring further analysis to determine whether the value is valid, or it is due to poor predictive performance of the model, indicating the need for retraining or structural adjustments.
b. Comparison of the actual and predicted demand: once the predicted demand is validated, it is compared with the actual demand. For that, a confidence interval of the prediction similar to the one of previous step is considered: [(1- α) prediction-ℓ std, (1+ α) prediction+ℓ std], where std is the standard deviation of the *n* previous days computed in the previous step, ℓ is a parameter that should be calibrated, and, for simplicity, α is the same value as in previous step. If the actual value falls within this interval, it is considered a normal value, otherwise, it is considered an anomaly, and a further analysis is needed.

The workflow of the solution is depicted in Figure 65.

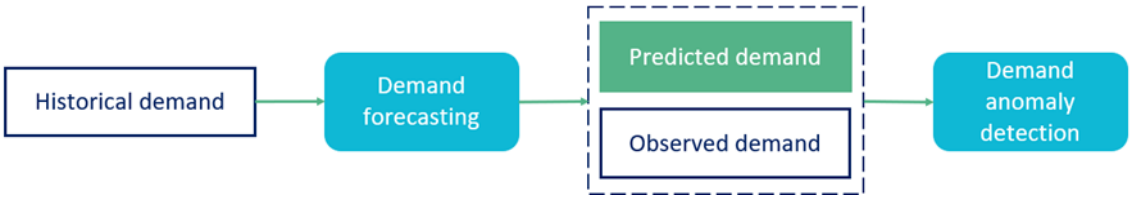

Figure 65: Demand anomaly detection workflow

This methodology allows the identification of anomalies in the demand, enabling their analysis in terms of calendar events (festivities, holidays, etc.), weather conditions, and planned events to look for possible explanations.

5.2.3 Implementation and validation of the methodology

To validate the methodology, it was applied to predict demand in the Madrid Region at the district level, covering a total of 185 districts (see Figure 66). The OD matrices were aggregated to consider the hourly originated trips per district, resulting in 185 hourly time series to be forecast simultaneously.

The data used to test and compare the different approaches correspond to complete, standard weeks of 2024 (i.e., weeks without festivities or holidays, and outside holiday periods such as Christmas, Holy Week, or summer). The goal is to evaluate the performance of all approaches on

regular time series and to identify anomalies in more stable periods, as anomaly detection during holiday periods is more complex.

Thus, the selected dates range from 16 January 2024 to 31 May 2024 and 16 September 2024 to 30 November 2024, excluding the week of Holy Week (25 March 2024 to 31 March 2024) and the May bank holiday week in Madrid (29 April 2024 to 5 May 2024).

Additionally, for each time step, the hour (from 0 to 23) and day of the week (from 0 to 6) were added in sine, cosine coordinates to the demand series.

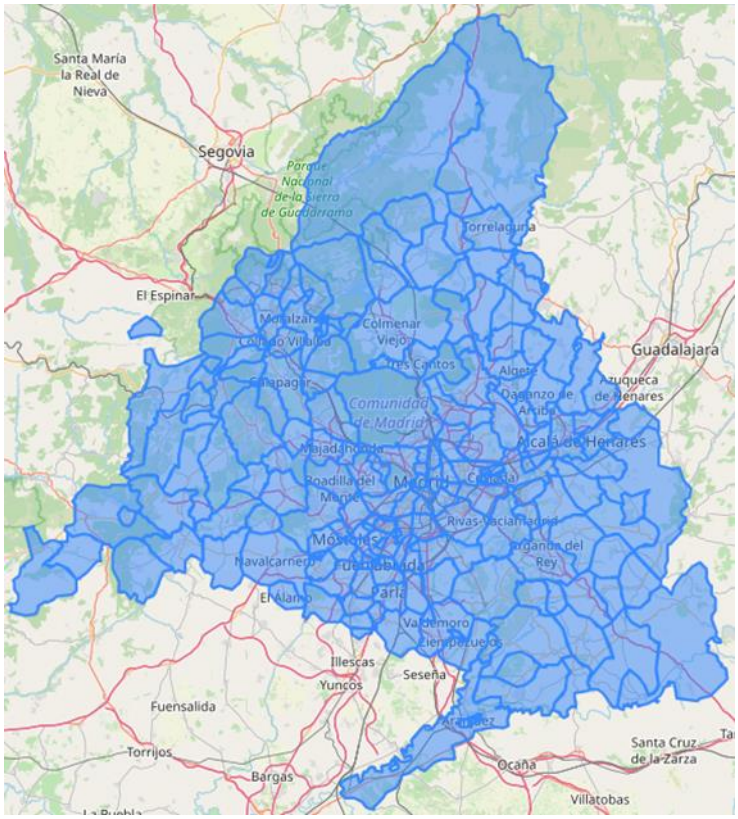

Figure 66: Division into districts of the Madrid region

5.2.4 Implementation of the demand forecasting model

Due to the high volume and dimensionality of the data, traditional time series forecasting models, such as Autoregressive Integrated Moving Average (ARIMA) and its variants, become impractical. These models struggle with long-term dependencies and high-dimensional feature spaces. Therefore, a deep learning-based approach is adopted to develop the forecasting algorithm.

Four different architectures are considered:

1. Long Short-Term Memory (LSTM) networks: LSTM networks are designed to handle long-range dependencies in sequential data. This enables them to learn patterns over extended time horizons, making them well-suited for time series forecasting where past information influences future values.
   This model consists of 3 LSTM layers of 128 units and dropout of 0.3, followed by a dense layer of 1024 neurons and a final dense layer of 24*185 neurons (the size of the output: 24 times steps and 185 zones).

2. LSTM network with multi-head attention: Integrating multi-head attention allows the model to focus on the most relevant past time steps rather than treating all historical data equally. This

mechanism assigns different attention weights to different time steps, enabling the model to capture complex temporal dependencies more effectively. This enhances the learning process by allowing the model to attend to multiple aspects of the input data simultaneously, improving the interpretability and robustness of the predictions.
This model keeps the structure of the previous LSTM model, but between the three LSTM layers and the two dense layers it includes multi-head attention layer (with 4 heads, 32 key dimensions, and 0.1 of dropout) and an average pooling layer.

3. LSTM network combined with convolutional neural network (CNN): The addition of a CNN helps extract local and short-term patterns from the time series before passing the information to the LSTM layers. This also allows the extraction of the spatial information and dependencies in the demand matrices. CNNs excel at identifying spatial and hierarchical relationships in data, making them useful for capturing local features and trends. When combined with LSTMs, CNNs enhance the model's ability to learn both short-term variations and long-term dependencies, leading to improved forecasting accuracy.
   This model also keeps the structure of the initial LSTM model, but it adds on top a convolutional layer with 64 filters and kernel size of 5.

4. LSTM networks with multi-head attention combined with convolutional networks: This hybrid architecture leverages the strengths of all the previous models, making it particularly effective for complex, high-dimensional time series data.
   The architecture of this model combines the previous three architectures: A convolutional layer, followed by three LSTM layers, a multi-head attention layer, and a global average pooling layer, finishing with two dense layers (all of them with the same parameters already described).

The objective is to compare the four approaches to determine which one best fits the demand forecasting problem. To establish a benchmark, the LSTM network is used as the baseline model, allowing us to evaluate the impact of each additional component (multi-head attention and CNNs) on the model's performance. This comparison provides insights into how each architectural enhancement contributes to prediction accuracy and overall model effectiveness.

The models are trained using the mean absolute error as the loss function and the optimization method Adam (adaptive moment estimation), which is a faster and more efficient extension of the stochastic gradient descent, providing adaptive learning rates for improved convergence.

Following standard machine learning practices, the data is split into two datasets, each of which assists in a different task of the model implementation process:

- Training set: this set is used to train the model. It includes all days up to 14 November 2024.
- Test set: once the model is trained, this set is used to assess its predictive performance on new data (i.e., its ability to generalise). It consists of the last 16 days (15 November to 30 November 2024) of the dataset.

To generate the samples, a 9-days window was selected. This way, the model predicts the hourly demand of one day (i.e., 24 time steps) based on the observed hourly demand of the previous nine days. Thus, the predicted demand has size 185x24 (185 districts x 24 hours). To generate completely independent train a test sets, the samples were computed considering only the dates within each set. This way, no single day belongs to both the training and test set, allowing for a cleaner evaluation of predictive performance and generalisation ability. Considering the 9-days window restriction, the train set contains 4177 samples, and the test set contains 145 samples. Moreover, for the test set, as the nine first days are used to generate the first sample, only the last 7 days of November are used to test the predictive performance of the model.

The predictive performance of the model is assessed using the square root mean square error (RMSE) and the mean absolute percentage error (MAPE) metrics, defined as follows:

$$\text{RMSE} = \sqrt{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2 / n}\,,$$
$$\text{MAPE} = 1/n \sum_{t=1}^{n} |(y_t - \hat{y}_t)/y_t|\,,$$

where *n* is the number of observations in the set, y_t is the actual value of the series at time *t*, and $\hat{y}_t$ is the model prediction at time *t*.

The RMSE provides the mean predictive error at each predicted instant. While the MAPE provides the average percentage that the predictive errors suppose with respect to the real value at each predicted instant. It is important to note that, given the same predictive error in terms of RMSE, the MAPE value can vary significantly depending on the real value of the variable. For example, a prediction of 3 when the real value is 4 is not the same as a prediction of 49 when the real value is 50. In both cases, the RMSE is 1, however, in the first case the MAPE is ¼= 0.25 and in the second, 1/50=0.02. This means that the MAPE metric contextualizes the magnitude of the error, penalising more the predictive error of small values of the variable. Therefore, both metrics provide complementary information on the predictive error, giving both absolute and relative information with respect to the real value, and it is important to take both values into account when interpreting the results.

5.2.5 Implementation of the anomaly detection algorithm

The last step of the process involves the calibration of the parameters defining the confidence intervals for: i) the validation of the predicted demand, and ii) the identification of anomalies.

For that, different values were considered, and the obtained results were contrasted with experts of Nommon's team. The selected values were a 6-period moving average, a Bollinger bands of k,ℓ =2 times the std and α=1 for the validation of the expected and actual demand.

5.2.6 Results

The results obtained are shown below. Table 13 shows the RMSE and MAPE values for the full week of the test set for each approach, and Figure 67 displays the RMSE and MAPE values for each day of the test set.

Table 13: Predictive performance of each model on the test set

Error	LSTM	LSTM with multi-head attention	LSTM & CNN	LSTM with multi-head attention & CNN
RMSE	3205.66	4033.71	3423.13	3028.76
MAPE	0.07	0.08	0.07	0.08

As can be seen in the table and figure, the four models have a remarkable predictive performance, with MAPE values below 10% for every day in all the cases. This shows very good and stable predictions along the week. The LSTM with multi-head attention & CNN approach is the one with lowest RMSE. However, when contextualising this value with the MAPE, the one of this model is higher than that of the two approaches without multi-head attention. This is because the model better predicts higher demand volumes (more penalised by the RMSE error) but fails to forecast lower demand volumes (more penalised by the MAPE error).

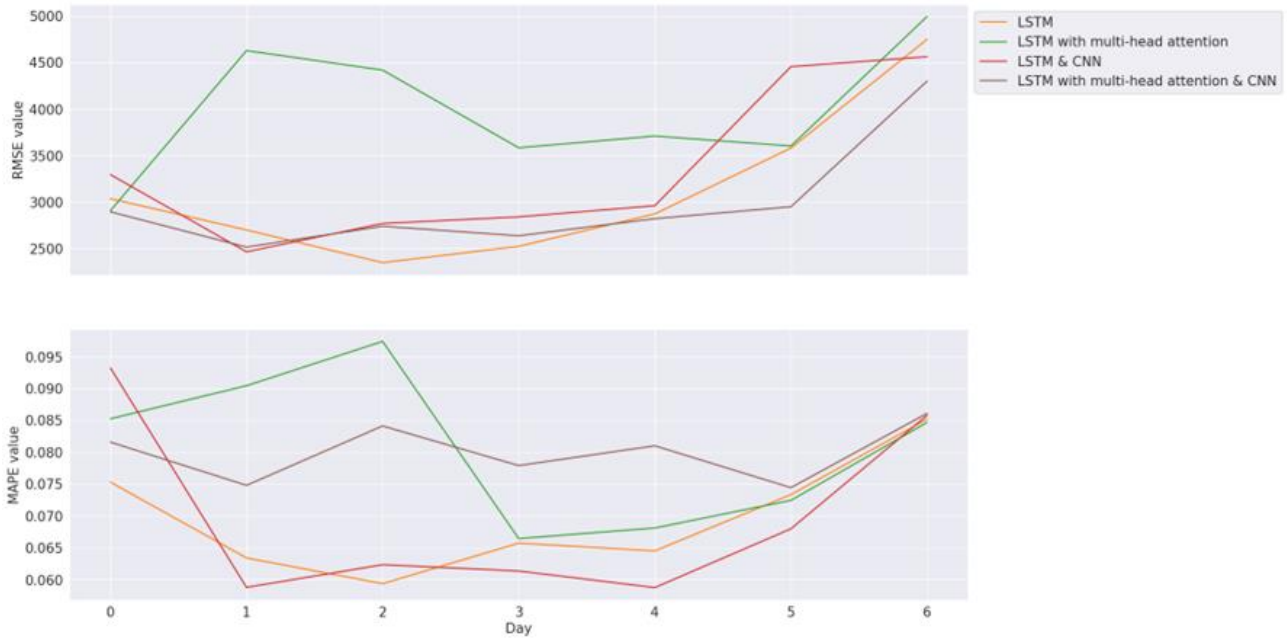

Figure 67: RMSE and MAPE values for each day of the test set

Next, the predictive capability of each model is compared for specific zones. Figure 67 illustrates this analysis with one district for the test set. As can be seen, the predictive performance of the models is very good for each day. It is interesting to note that the models without multi-head attention better fit the low demand curves of the first hours of the day, whereas their predictions are less accurate for the peak demand periods during the daily rush hours, failing to fit well to the three peaks. With the models with multi-head attention the situation is precisely the opposite, they fit the three peak demands accurately, but show worse predictive results for the low demand volumes during the first hours of the day. This same behaviour is observed in the rest of the zones and explains why the models with multi-head attention have better RMSE but worse MAPE (as already noticed).

Next, we analyse the ability of each approach for anomaly detection. Table 14 presents the number of abnormal predictions of each model on a full week of the test set and the number of districts in which they are done. Percentages are shown in brackets. As we are analysing hourly predictions over a full week for 185 districts, the total number of predictions is 24×7×185=31080.

Table 14: Abnormal predictions for each model

	LSTM	LSTM with multi-head attention	LSTM & CNN	LSTM with multi-head attention & CNN
Abnormal predictions	733 (0.02)	1304 (0.04)	455 (0.01)	1252 (0.04)
Districts with abnormal predictions	146 (0.79)	184 (0.99)	138 (0.75)	172 (0.93)

Once the algorithm detects abnormal values, these should be further analysed by a technical expert to determine whether they are actual abnormal demand situations, or on the contrary, they are valid values. In this case, all the identified anomalies correspond to valid values. They are detected as anomalies since their confident interval is too narrow. This is due to a very small variability in the previous days (translated in a very small std value, which determined the width of the interval). A simple visual inspection of the data reveals that those values are pretty close to their confident interval, confirming that they are valid predictions (this is illustrated with an example next). The

values of the confident analysis can be tuned to make them wider or narrower, by multiplying by a given factor, according with the specific needs of the final users.

Table 15 presents the number of anomalies detected by each model, and the number of districts with anomalies. As in previous case, percentages are shown in brackets.

Table 15: Anomalies identified with each model

	LSTM	LSTM with multi-head attention	LSTM & CNN	LSTM with multi-head attention & CNN
Anomalies	844 (0.03)	956 (0.03)	344 (0.01)	660 (0.02)
Districts with anomalies	167 (0.90)	182 (0.98)	137 (0.74)	160 (0.86)

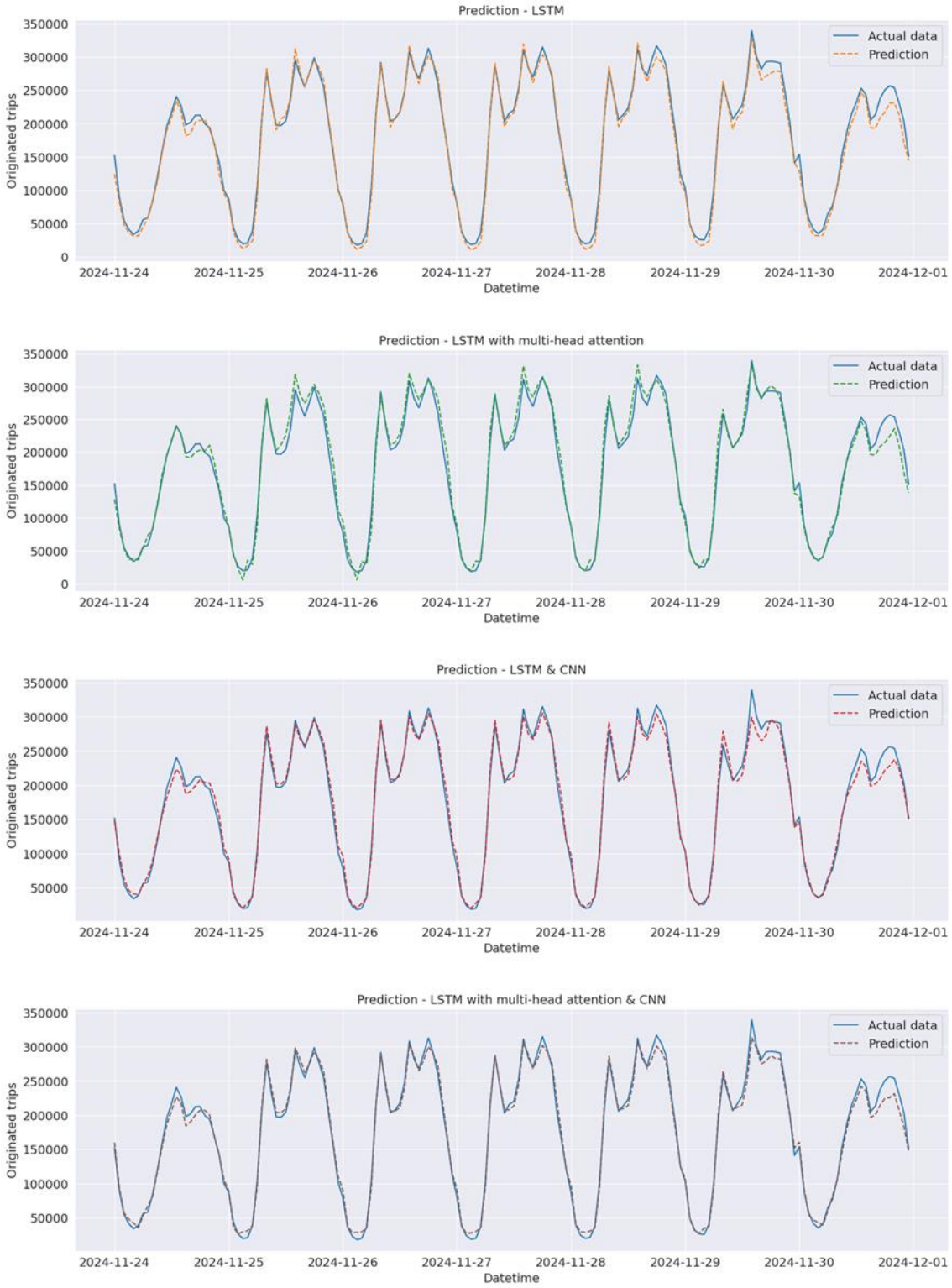

Figure 68: Predictive comparison of the four models for one district

As can be seen, very few abnormal predictions and anomalies are identified (less than 5% and 4%, respectively). This is due to two facts: 1) the regularity of the times series and 2) the good predictive performance of the models. None of the anomalies detected correspond to real anomalies but just like before, their confident interval is too narrow due to a very small variability in the previous days, making abnormal small fluctuations in the demand. A simple visual inspection of the data reveals

that those values are similar to the ones detected in the previous step and are very close to their confident interval as well (this is also illustrated with an example next).

The model yielding the best anomaly detection results is the LSTM & CNN model. Despite this model achieves very similar accuracy results to the LSTM model (Table 13), its anomaly detection capability is far better, presenting much fewer mislabelled abnormal values (Table 14) and less than a half of mislabelled anomalies (Table 15). This makes sense, as this model combines the capability of LSTMs for long-term pattern analysis with that of CNNs for short-term and spatial pattern analysis, making it more robust. On the contrary, the models with multi-head attention presents more mislabelled abnormal values and anomalies. This may be due to the same reason that caused their MAPE error to be greater: their predictive error is bigger for smaller demand values, for which the confident interval is narrower.

To illustrate the results with a concrete example, Figure 69 displays the abnormal predictions of each model for the same district as before and Figure 70 shows the anomalies identified with each model. As can be easily seen in the graphs, none of these values are anomalies. Indeed, they are very close to the confident interval. As these values are identified for further analysis, a simple data visualisation is enough to conclude that all the values are normal, yielding a complete validation of both the predicted and the observed demand of the week for the district.

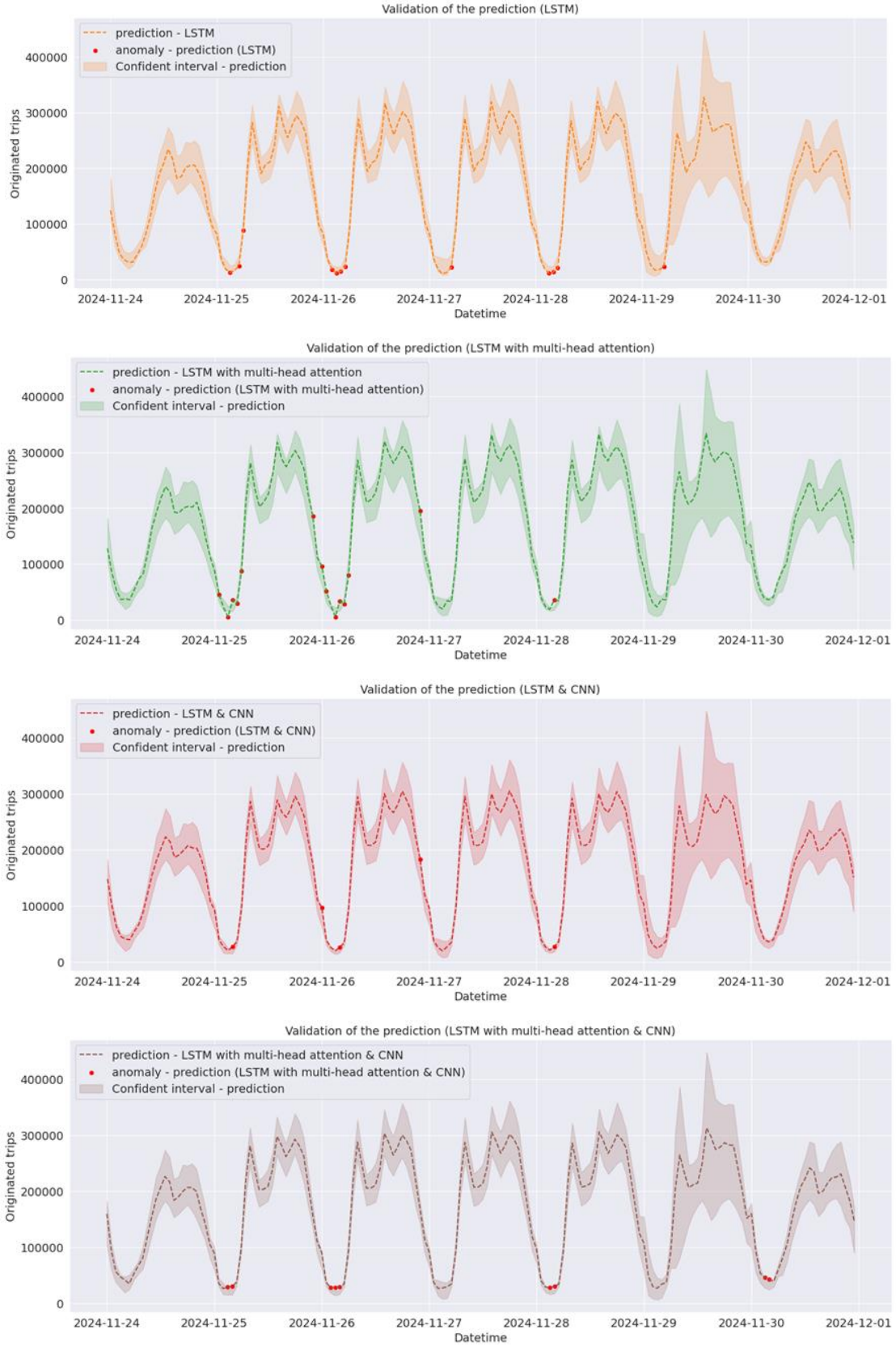

Figure 69: Abnormal predictions for one district

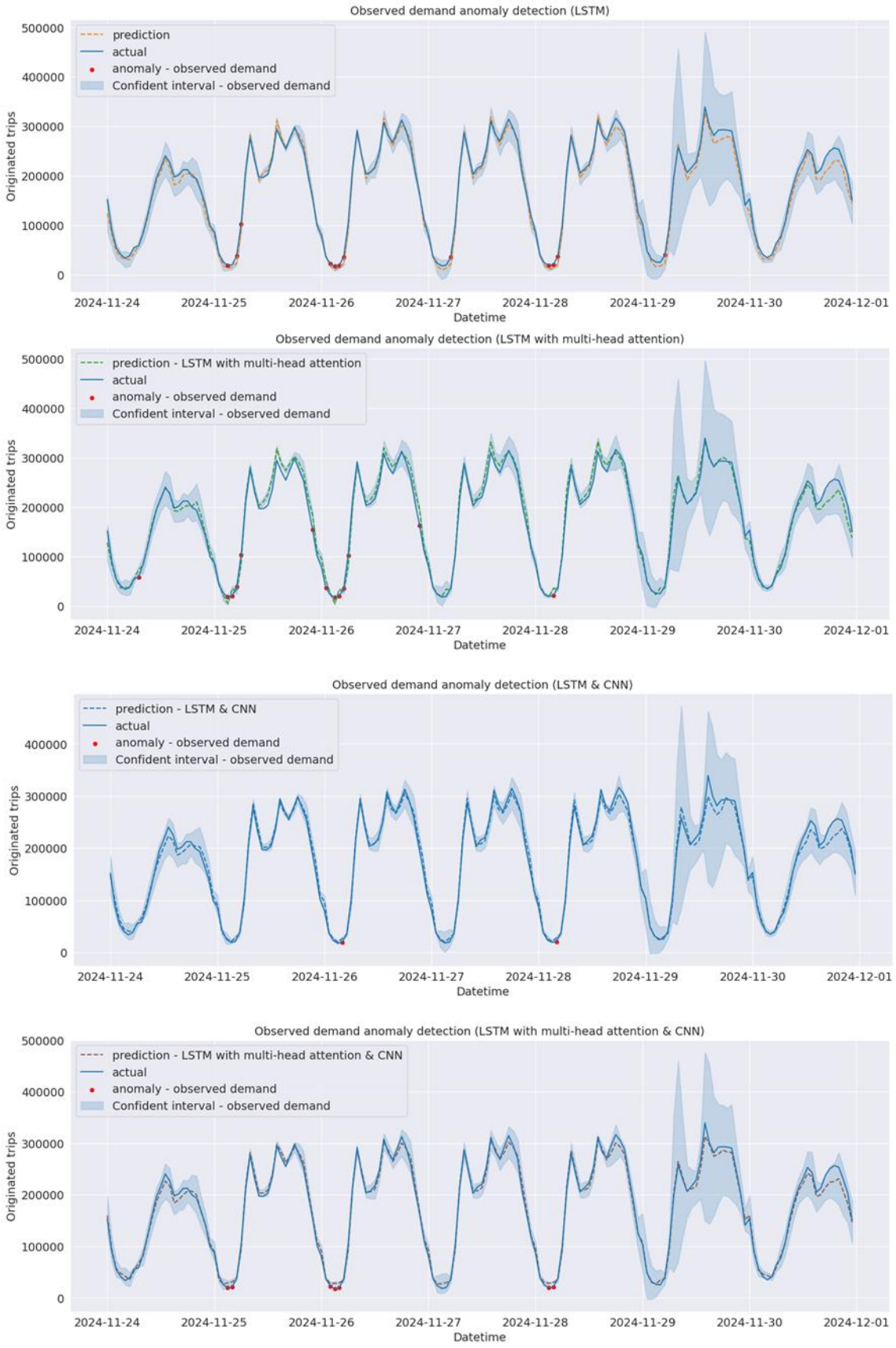

Figure 70: Anomalies identified for one district

6. CONCLUSIONS

Deliverable D3.4 concludes the technical work carried out within Work Package 3 (WP3) of the CONDUCTOR project, bringing together the final implementations of the key modules related to data harmonisation, fusion, dynamic optimisation, anomaly detection, and network load balancing. These components have been progressively developed and refined through previous iterations (D3.1, D3.2, and D3.3), and their final versions—presented herein—reflect the results of extensive simulation activities, technical validation, and feedback integration from cross-WP collaboration, particularly with WP5. Across all tasks, progress beyond the state of the art has been achieved, with the main innovations being summarised below.

Data Harmonisation was used to adopt a standard information model for schemas and semantics for data used by CONDUCTOR applications. Within CONDUCTOR, we developed various components, tools, and algorithms to support the investigation of CCAM services. Our harmonised representation of data models allowed the seamless integration of applications from different partners into CONDUCTOR's platform.

The CONDUCTOR dataspace ensures that no single entity controls all data, allowing each stakeholder to manage their own data sharing. It includes decentralised and centralised components, such as an Identity Service and a Catalogue Service, connected to the Orion Message Broker. The key components selected for its realisation ensure interoperability and security and encompass a data exchange process involving contract offering and negotiation, including defining asset data, creating assets and policies, configuring contracts, querying offers, and performing data transfers. CKAN enables API access, datastore, metadata management, search functionality, visualisation tools, and federation capabilities. Finally, the deployment of the dataspace adopts a containerised big data architecture for flexibility and efficient resource management.

CONDUCTOR developed **several data fusion approaches** for harmonising and fusing data from various sources. These involved algorithms and models for:

- **Characterisation of last-mile delivery trips and estimation of last-mile delivery demand from mobile network, surveys, and logistic operation data.** This methodology allows the characterisation of last-mile delivery trips and flows. Detailed mobility pattern information is obtained, enabling the informed definition of delivery routes needed for the urban logistics UC of the project. As a final step, this methodology is being applied to identify logistic MND users. To do so, the trip features defined in step 5 are computed for each MND user, and the clustering model obtained in step 6 is used to assign them to a pattern. Specifically, the probability of a user belonging to each cluster is calculated by applying the softmax function to their distances to the cluster centroids (these probabilities add up to 1). Users with a probability above a given threshold for a specific cluster are assigned to that cluster. Those who do not meet the threshold for any cluster are labelled as non-logistics. Finally, the methodology defined can be applied to the flow characterisation of many kinds of users (e.g., taxi drivers), as long as a sample of ground truth data for such types of users is available.
- **Shared mobility demand estimation.** The shared mobility demand estimation methodology effectively combines real-world shared mobility data with behavioural and mobility pattern information from surveys and mobile network data. The resulting CCAM-DRT demand estimates are both spatially and temporally realistic, and the mode substitution model provides valuable insights into the expected impact of CCAM services on the existing transport system. This enables more informed scenario planning and

decision-making in urban mobility contexts, particularly in the application areas of UC1-Madrid and UC3.

- **Enrichment of users' profile.** The results obtained for each cluster are very positive, showing the effectivity of the methodology for the characterisation of car ownership in a region. This corroborate the initial hypothesis that the car ownership of a user can be characterised in terms of their sociodemographic and economic profile and mobility patterns, combined with the sociodemographic features and mobility patterns of the population that resides in the same zone and the availability of public transport services in the residence's zone. A proof of that is the generalisation capability of the models, showing impressive predictive accuracy on the zones not considered for training (i.e., zones never seen before).
- **Household size assignment.** This methodology estimates the household size of MND users based on survey data, using sociodemographic features and place of residence to probabilistically assign household sizes. Data from various sources (e.g., census data, household surveys, mobile network data, etc.) have been utilised to realise the model. An iterative algorithm assigns household sizes in two main steps: (a) *Unipersonal Households*, which involves (i) the calculation of the number of unipersonal households per census tract and (ii). The assignment of users to unipersonal households based on age group distribution; (b) *Multi-person Households*, which (i) assigns the remaining users to multi-person households using age group distribution, (ii) updates household size distribution to match census tract averages, and (iii) repeat until the average household size matches the census tract data. The developed methodology was tested in the Madrid region and showed a high accuracy with an R^2 score of 0.89, indicating effective household size estimation.
- **Identification of unusual traffic patterns caused by large-scale events.** This involves an AI-powered decision support tool that enhances situational awareness and assists with mobility planning. The system comprises two main components: a fuzzy inference engine and a multi-criteria decision analysis. The former uses fuzzy logic to classify traffic conditions based on vehicle density, traffic speed, and the gap between vehicles, and by utilising 27 fuzzy logic rules to interpret traffic conditions, it outputs a traffic condition score (0-100), categorised as free-flowing, moderate, or congested. The latter applies the TOPSIS methodology to prioritise traffic events and recommend optimal routes. It constructs a weighted decision matrix, normalises data, and calculates closeness coefficients to rank alternatives. This is used to evaluate routes based on criteria like travel time, emissions, distance, and number of traffic events. The developed tool helps decision-makers identify the most optimal routes by providing a comprehensive analysis of traffic conditions and route performance.
- **Coupled Aimsun-FleetPy Simulation Data.** This suite of techniques enables the integration of urban logistics into DRT services, and can be simulated in Madrid using data from Nommon. FleetPy, a Python-based DRT simulation tool, is coupled with Aimsun Next to account for realistic traffic conditions. Initially planned for microscopic simulation, the project shifted to using a macroscopic model of Madrid. Data inputs include a calibrated macroscopic model, origin-destination pairs for DRT and freight requests, and FleetPy simulation parameters. Freight requests are clustered for same-day delivery, ensuring DRT passenger service quality is maintained. Traffic state data from Aimsun Next is used to plan routes and assign vehicles, with travel times calculated using the Dijkstra algorithm.
- **Space-time context and heterogeneous data fusion.** This methodology adopted a context graph as a framework for data fusion that provides semantic descriptions of entities

and encodes their spatiotemporal and hierarchical context. It links data points to structured context layers, enabling advanced reasoning and feature extraction across diverse data sources. Contexts and entities are stored explicitly in the graph, with measurements encoded as properties on the edges. Contexts are structured hierarchically, with spatial and temporal relationships represented by specific types of edges. Two context encoding methods were compared: explicit and implicit. Explicit context encoding, where space and time are represented jointly, proved more effective for retrieval and aggregation due to better indexing options. Implicit encoding, which separates spatial and temporal contexts, caused issues with query processing and was less efficient. The context graph architecture was used to create an API for accessing Slovenian traffic data, primarily for Traffic Events Assessment Services in UC2. However, graph databases face limitations as primary stores for time series measurements, including performance bottlenecks and inefficiencies in querying time series data. Therefore, a columnar database is recommended for storing time series measurements, while graph databases are better suited for storing rich semantic data for traffic management and forecasting.

The development of models for network balancing, dynamic optimisation, and anomaly detection for transport-related applications was a core objective of WP3. In the network balancing domain, various solutions were developed. A **traffic management solution for signal vehicle couple control** implemented a decentralised control scheme, which demonstrated enhancements in the transport network's throughput while improving the system's resilience against unforeseen incidents and disruptions. The model-based optimisation efficiently encapsulates the optimal route distribution task and dynamic route choice function based on the current state of the network, making efficient rerouting possible for better network load balancing. In this research direction, we have tested the centralised CAV routing and signal optimisation. Decentralised incident management using CAV routing has been successfully tested along the main route and signal optimisation process. A **social routing with multimodal perspective** was developed as a mechanism that persuades travellers to choose routes for the benefit of the system. In the multimodal setting of D3.2, the social rerouting framework is applied and tested in the public transport network of Twente, illustrating that 25% of the maximum improvement in efficiency can be obtained with 20% of travellers willing to act socially. We considered a stochastic extension of this model, assuming the portion of travellers willing to act socially is random following an exogenous empirical discrete distribution. This leads to additional challenges since travel times depend on route flows, and thus also become random. Consequently, travel advice should be adopted accordingly, illustrated to provide opportunities to reach policy objectives, e.g. since longer routes may be accepted as long as one is compensated by suggesting faster or even individually optimal routes in other cases. **Prediction models for demand-responsive transport were developed** by testing various model architectures and finding that a single holistic model, which takes route information and other features to predict for the entire year, worked best. This approach reduces computational resources, minimises overfitting, and allows knowledge transfer between different cases. The model architecture includes embedding layers for time and route information, convolutional layers for current orders, and dense layers with dropout. It outputs five quantiles for passenger numbers, the probability of at least one passenger, and the current passenger count. These additional outputs improve training efficiency and accuracy. Training uses mean absolute error for current counts, binary cross-entropy for classification, and pinball loss for quantiles. Metrics like mean absolute error (MAE) and mean absolute percentage error (MAPE) assess model accuracy. The model outperforms a baseline model, especially for short-term forecasting, with finer time resolution being more suitable for short-term and coarser for long-term forecasting.

Optimisation techniques based on metaheuristics were designed for the realisation of an **urban freight distribution with the DRT service for last-mile delivery**. The solution is flexible, handling various vehicle types, capacities, access restrictions, time windows, and the integration of people and parcels. Due to large input data sizes, a pre-optimiser was implemented to cluster tasks, making them manageable for the optimiser. This pre-optimiser processes and cleans the data, divides it into smaller groups, and consolidates outputs into a single result. Strategic meeting points were defined to improve DRT efficiency, balancing user convenience and operational efficiency. A clustering model was designed considering distance, direction, and time properties of trajectories. Two methods for assigning initial delivery time windows using spatio-temporal aspects were investigated. The first method, Region-Based Equal Delivery Time Windows, counts the number of freight requests within each region and assigns them 2-hour time windows during working hours, ensuring an equal number of deliveries per time window. The second method, DRT Demand-Based Delivery Time Windows, considers the forecast of DRT passengers. It counts the number of DRT passengers for each region and 2-hour time window during working hours. Freight requests are then assigned 2-hour time windows proportional to DRT demand, increasing the likelihood of meeting delivery windows due to vehicle availability in those regions. Optimisation models were also used for tackling the problem **of adaptive routing based on minimising the risk to provide a more accurate description of passenger behaviour in a PT network**. The two main difficulties we take into account for approaching this problem are (i) modelling adaptive route choice in a stochastic time-dependent PT system, and (ii) defining risk and the choice of a suitable risk measure. The work involved the definition of a stochastic time-dependent network and routing policies as an adaptive extension for deterministic path choice. The model is kept very general, which allows it to be applied to a variety of different settings. But due to the generality, it is less intuitive and applicable for concrete examples. As the second goal was however to analyse risk measures, the generality of the model allows it to take a broad view at them, define risk in a general manner and study the properties and suitability of different examples without restricting ourselves to a specific setting. Furthermore, a solution was developed for the **optimisation of demand-responsive transport**. This component enables accurate forecasting of vehicle requirements by combining demand predictions with insights from historical reservation data. By simulating realistic future reservation scenarios and optimising routes across multiple samples, the system identifies patterns in fleet usage and variability, allowing for precise estimation of the number and type of vehicles needed across different routes and periods. These insights form a critical foundation for future applications such as dynamic pricing, where prices can be adjusted based on predicted demand and fleet availability to balance load, optimise resource utilisation, and maximise revenue. Beyond dynamic pricing, other use cases include: (i) workforce planning, namely helping in the allocation of drivers and supporting the staff more efficiently; (ii) operational scheduling, allowing for better shift management, vehicle maintenance planning, and reduction of idle times and (iii) strategic decision-making, such as fleet expansion, service area adjustments, or introduction of new vehicle types based on long-term demand patterns. Ultimately, this approach boosts operational efficiency and provides a data-driven foundation for scalable and customer-responsive transport services.

In the domain of anomaly detection, models for detecting anomalies in traffic patterns and transport demand were developed. **Anomaly detection in traffic patterns** aims to identify irregularities in traffic behaviour, offering timely insights into deviations from expected operational norms. The Anomaly Detection component is designed for efficiency and integration within the CONDUCTOR framework, tailored for the Slovenian pilot. The system periodically analyses current traffic data, flags anomalies, and sends notifications to enhance traffic management. The developed solution encompasses an ensemble framework that integrates the outputs of multiple heterogeneous base models (one-class support vector machines, isolation forest, local outlier factor, and long short-term memory neural networks) through both majority voting and weighted averaging schemes.

These combination strategies were selected for their capacity to improve detection consistency while reducing the sensitivity to noise, model-specific errors, and overfitting tendencies. Results demonstrated accuracy of up to 88% detecting reduced-speed anomalies during off-peak hours, a scenario often difficult to capture due to limited variation in input signals. Conversely, **anomaly detection in transport demand** aimed to develop a time series model capable of accurately capturing and forecasting mobility demand patterns and detecting anomalies in the demand. The algorithm developed consists in two steps including a ML time series forecasting algorithm that forecasts the expected demand for a day based on a multi-dimensional time series analysis over historical demand information and an anomaly detection algorithm that compares the expected demand (obtained in the previous step) with the actual demand of the day to identify anomalies. Four approaches were tested for their suitability in realising the forecasting algorithm, including techniques such as long short-term memory and convolutional neural networks. Following experimentations, it was determined that the developed models have an excellent predictive performance, with MAPE values below 10% for every day in all the cases. The LSTM with multi-head attention & CNN model is the one with the lowest RMSE, while the LSTM model is the one with best MAPE. The models with multi-head attention display better behaviour for the peak demands but show worse predictive performance for the hours with lower demand volumes. The models without multi-head attention exhibit the opposite behaviour. Regarding the anomaly detection, the models output very few abnormal predictions (less than 5%) and detect even less anomalies (less than 4%), because of the regularity of the series and their good prediction performance. The best performing model is the LSTM & CNN model, with an average of less than 70 abnormal predictions per day, and less than 50 anomalies detected per day (for the complete Madrid Region). Each of these anomalies must be analysed by an expert to determine whether it truly represents an anomaly. This volume is entirely reasonable and allows for an individual analysis by an expert. These results confirm the effectiveness and the practical feasibility of the methodology for anomaly detection.

7. REFERENCES

Abdel-Aty, M., Kitamura, R., & Jovanis, P. (1995). Exploring Route Choice Behavior Using Geographic Information System-Based Alternative Routes and Hypothetical Travel Time Information Input. Transportation Research Record 1493.

Andreatta, G., & Romeo, L. (1988). Stochastic shortest paths with recourse. Networks, 18, 193–204.

Artzner, P., Delbaen, F., Eber, J.-M., & Heath, D. (1999). Coherent measures of risk. Mathematical Finance, 9/3, 203–228.

Carrion-Madera, C., & Levinson, D. (2010). Value of Travel Time Reliability: A Review of Current Evidence. University of Minnesota: Nexus Research Group, Working Papers, 46.

Chabini, I. (1999). Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. Transportation Research Record, 1645, 170–175.

Chen, A., & Zhou, Z. (2010). The α-reliable mean-excess traffic equilibrium model with stochastic travel times. Transportation Research Part B: Methodological, 44(4), 493–513.

Croucher, J. (1978). A note on the stochastic shortest-route problem. Naval Research Logistics Quarterly, 25, 729–732.

Egidio dos Reis, A., Gaspar, R., & Vicente, A. (2009). Solvency II - An Important Case in Applied VAR (pp. 267–294).

Eikenbroek, O. A., Still, G. J., & Van Berkum, E. C. (2022). Improving the performance of a traffic system by fair rerouting of travelers. European Journal of Operational Research, 299(1), 195-207.

Filippi, C., Guastaroba, G., & Speranza, M. G. (2020). Conditional value-at-risk beyond finance: A survey. International Transactions in Operational Research, 27(3), 1277–1319.

Föllmer, H., & Schied, A. (2016). Stochastic Finance: An Introduction in Discrete Time (4th Rev. ed.). de Gruyter.

Frittelli, M., & Maggis, M. (2011). Dual Representation of Quasi-convex Conditional Maps. SIAM J. Financial Mathematics, 2/1, 357–382.

Gao, S., & Chabini, I. (2006). Optimal routing policy problems in stochastic time-dependent networks. Transportation Research Part B: Methodological, 40(2), 93–122.

Gao, S., Frejinger, E., & Ben-Akiva, M. (2010). Adaptive route choices in risky traffic networks: A prospect theory approach. Transportation Research Part C: Emerging Technologies, 18(5), 727–740. https://doi.org/10.1016/j.trc.2009.08.001

Hall, R. W. (1986). The fastest path through a network with random time-dependent travel times. Transportation Science, 20(3), 182–188.

Insurance, F. O. of P. (2006). Technical document on the Swiss Solvency Test. https://www.finma.ch/FinmaArchiv/bpv/download/e/SST_techDok_061002_E_wo_Li_20070118.pdf

Kahneman, D., & Tversky, A. (1979). Prospect theory: An analysis of decision under risk. Econometrica, 47(2), 263–291.

Miller-Hooks, E. D. (2001). Adaptive least-expected time paths in stochastic, timevarying transportation and data networks. Networks, 37(1), 35–52.

Miller-Hooks, E. D., & Mahmassani, H. S. (2000). Least expected time paths in stochastic, time-varying transportation networks. Transportation Science, 34(2), 198–215.

Philippe, J. (2006). Value at Risk: The New Benchmark for Managing Financial Risk. McGraw-Hill.
Rockafellar, R. T., & Uryasev, S. (2000). Optimization of conditional value-at risk. Journal of Risk, 3, 21–41.

Small, K. (1982). The Scheduling of Consumer Activities: Work Trips. American Economic Review, 72, 467–479.

Small, K., Noland, R., Chu, X., & Lewis, D. (1999). Valuation of Travel-Time Savings and Predictability in Congested Conditions for Highway User-Cost Estimation. Transportation Research Board.

Szegö, G. (2002). Measures of risk. Journal of Banking & Finance, 26(7), 1253–1272.

Szep, T., van den Berg, T., Cointe, N., Daniel, A. M., Martinho, A., Tang, T., & Chorus, C. (2023). Give and take: Moral aspects of travelers' intentions to participate in a hypothetical established social routing scheme. Cities, 133, 104132.

Vreeswijk, J., Bie, J., Van Berkum, E., & Van Arem, B. (2013). Effective traffic management based on bounded rationality and indifference bands. IET Intelligent Transport Systems, 7(3), 265-274.

ABBREVIATIONS AND DEFINITIONS

Term	Definition
ADAM	Adaptive Moment Estimation
API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
ASGI	Asynchronous Server Gateway Interface
AUC	Area Under the Curve
CAV	Connected And Automated Vehicle
CCAM	Connected, Cooperative and Automated Mobility
CDR	Call Detail Record
CKAN	Comprehensive Knowledge Archive Network
CNN	Convolution Neural Network
CRON	Command Run On Notice
CRTM	Madrid Regional Transport Consortium
CRTM	Madrid Regional Transport Consortium
DRT	Demand-Responsive Transport
DTA	Dynamic Traffic Assignment
EDA	Exploratory Data Analysis
EDSC	Eclipse Data Space Components
EMD	Mobility Household Survey (acronym in Spanish)
ETA	Estimated Time of Arrival
GPS	Global Positioning System
IDSA	International Data Spaces Association
INE	Spanish National Statistical Office (acronym in Spanish)

JAX	Just Another XLA
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LNS	Large Neighbourhood Search
LOF	Local Outlier Factor
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MND	Mobile Network Data
OD	Origin Destination
ORP	Optimal Routing Problem
OSRM	Open-Source Routing Machine
PCA	Principal Component Analysis
POI	Point of interest
PT	Public Transport
RFE	Recursive Feature Elimination
RMSE	Root Mean Square Error
ROC	Receiver Operating Characteristic
SDVRP	Stochastic and Dynamic Vehicle Routing Problem
STD	Stochastic Time-Dependent
SVCC	Signal-Vehicle Cooperative Control
SVM	Support Vector Machines
TMC	Traffic Management Centres

TOPSIS	Technique for Order Preference by Similarity to Ideal Solution
UC	Use Case
XLA	Accelerated Linear Algebra