Data-Driven Operational Prediction Model

Grant Agreement N°:	875022
Project Acronym:	**SORTEDMOBILITY**
Project Title:	Self-Organized Rail Traffic for the Evolution of Decentralized MOBILITY
Funding scheme:	Horizon 2020 ERA-NET Cofund
Project start:	1 June 2021
Project duration:	3 Years
Work package no.:	2
Deliverable no.:	D2.2
Status/date of document:	Final
Due date of document:	30/08/2023
Lead partner for this document:	DTU
Project website:	www.sortedmobility.eu

Dissemination Level		
PU	Public	**X**
RE	Restricted to a group specified by the consortium and funding agencies	
CO	Confidential, only for members of the consortium and funding agencies	

Revision control / involved partners

The following table gives an overview of the elaboration and processed changes in the document:

Revision	Date	Name / Company short name	Changes
1	02/10/2023	DTU	First draft
2	14/11/2023	Banedanmark	Expansion of sections 1 and 2, draft of sections 3, 5, and 7
3	01/12/2023	DTU	Final version
4	04/12/2023	Univ. Gustave Eiffel	Minor revisions
5	19/12/2023	DTU	Cleaned final version

The following project partners have been involved in the elaboration of this document:

Partner No.	Company short name	Involved experts
1	univEiffel	Bianca Pascariu, Paola Pellegrini
2	SNCF	
3	DTU	Filipe Rodrigues, Carlos Lima Azevedo
4	BDK	Johan Victor Flensburg, Fabrizio Cerreto
5	ISTC-CNR	
6	RFI	
7	TUDelft	

Executive Summary

The objective of this D2.2 is to describe the data-driven demand prediction model used in the demand prediction module of SORTEDMOBILITY. In this report the role of the prediction model in the overall SORTEMOBILITY framework is first outlined. Subsequently we describe the modelling architecture, its input and output, how the output is used and the interfaces with other modules. This is followed by a technical overview of the prediction model details, a description of the data used to estimate the model, and the results obtained from estimation and numerical experiments. Finally, the report touches upon planned extensions of the model until the end of SORTEDMOBILITY.

Table of contents

Index of Figures

Index of Tables

Table of abbreviations

TSP	Traffic State Prediction
RTTP	Real-Time Traffic Plan
OD	Origin-Destination
AFC	Automatic Fare Collection
PAP	Passenger Assignment Plan
MNL	Multinomial Logit
CSV	Comma-Separated Values
GTFS	General Transit Feed Specification
XML	Extensible Markup Language
TAZ	Traffic Analysis Zone
DAS	Day Activity Schedule
GTFS	General Transit Feed Specification
GNN	Graph Neural Network
NRI	Neural Relational Inference
GRU	Gated Recurrent Unit
GCN	Graph Convolutional Network
CBTC	Communications-Based Train Control
RMSE	Root Mean Squared Error
MAE	Mean Absolute Error

1 INTRODUCTION

SORTEDMOBILITY (Self-Organized Rail Traffic for the Evolution of Decentralized MOBILITY) aims at developing concepts, models and algorithms for self-organizing management of public transport operations in urban and interurban areas, specifically focusing on rail transport as a mobility backbone. In addition, a detailed simulation assessment platform will be developed to assess the proposed self-organization approach against a centralized one.

This deliverable describes the data-driven demand prediction model used in the demand prediction module of SORTEDMOBILITY. The report first outlines the role of the prediction model in the framework. Next, it describes the module in which it operates, its input and output, how the output is used and the interfaces with other modules. This is followed by a technical overview of the prediction model architecture, a description of the data used to estimate the model, and the results obtained from experiments. Finally, the report touches upon planned extensions of the model.

2 SORTEDMOBILITY FRAMEWORK

Conceptually, the modelling framework identifies the different actors and processes that are involved in train traffic management. We have developed a framework for the traffic management solution in SORTEDMOBILITY, depicted in Figure 1. The framework determines the way in which the control architecture interacts with the transport (simulation) system, identifying relevant flows of information to be included in a Traffic State and Demand Prediction to determine and implement the Real Time Traffic Plan (RTTP), which is the description of the train routes and schedules that are used within the project.

The aim of the demand prediction module is to produce reliable estimates of passenger behaviour for the traffic control module to manage traffic such that it benefits the passengers. The module is data-driven in the sense that it does not use theoretical knowledge about passenger behaviour to calculate the trip distributions of passengers but rather historical data from a specific case study to "learn" correlations in those data. These data include information about the operations of the trains, also known as "supply", and the behaviour of passengers, also known as "demand". In a real or simulated environment, data on supply and demand are assumed to be collected as they become available, e.g., through a so-called smart card system.

As depicted in Figure 1, the demand prediction module can be viewed as an integrated part of traffic control. It functions as a support for the automated dispatching module and also receives input from this module. Specifically, it receives supply input from the traffic state prediction and demand input from smart card streaming data, which is accessed through the control module. Notably, the supply data also include forecasts of traffic resulting from hypothesized traffic management decisions, so-called Real-Time Traffic Plans (RTTPs), for which the demand prediction model can produce predictions. The prediction consists of the origin-destination (OD) demand and the assignment of this demand to the expected future train runs, which is used in the control module optimization model to account for passengers in the decision-making. The prediction may come into play in two ways. Primarily, the prediction contains both expected and desired arrival times at the destination for every passenger group, which enables a measure of passenger delay for different RTTPs. Secondly, the prediction includes a detailed list of train runs to which passenger groups are assigned and the connections they need in order to complete their journey. These connections can be accounted for in different ways in the traffic management optimization problem. The way the information in the prediction is used is independent of the prediction model.

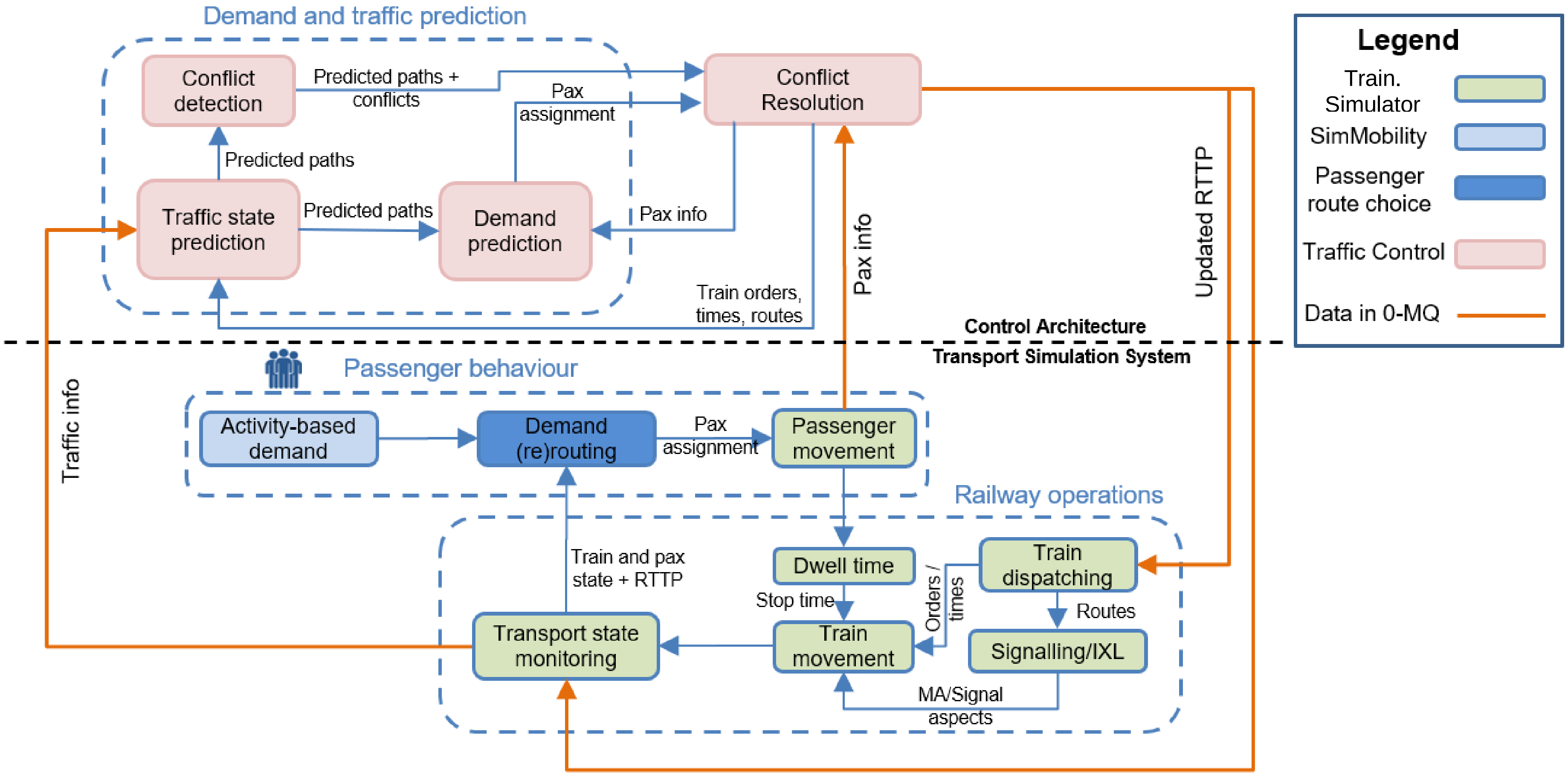

Figure 1: Modelling framework of the SORTEDMOBILITY traffic management

3 DEMAND PREDICTION MODULE

3.1 Introduction

The demand prediction module consists of two parts: a prediction model for the OD passenger demand and a model for assigning predicted demand to trains, thereby producing a prediction of the number of passengers in each train, their destinations and their route choice.

At the beginning of a short time interval, e.g., of 5, 10 or 20 minutes, the prediction model forecasts OD passenger demand within this interval and a certain small number of intervals into the immediate future. Since traffic management and control require information on demand in terms of passengers in each train and the passengers' destinations, a passenger assignment model transforms the OD prediction into a passenger assignment to individual trains in accordance with the RTTP.

3.2 Input and output

The input to the demand prediction model comprises:

1. At the start of operations, historical records of observed demand for all OD pairs
2. At the start of operations, historical records of observed traffic in the network, including realised timings at stations
3. During operations, records of observed demand in terms of tap-in and tap-out locations and times of passengers in real-time
4. During operations, records of observed and predicted traffic plans, including realised and predicted timings at stations

Historical data (points 1 and 2) will come from databases collecting information on demand and supply, respectively. The demand data will typically come from an automatic fare collection (AFC), often based on smart cards, in which passengers interact with physical card readers at stations or in vehicles to record the time and location of entering, transferring, or exiting a public transport system or vehicle. The supply data may be recorded in an operations database by an operator or infrastructure manager and must include at least scheduled and realized arrival times of vehicles at all stations in the network of interest.

The real-time data (points 3 and 4) are recorded by these same systems as time passes. In a simulation, this will be carried out by the simulator, which must handle both operations of infrastructure, vehicles, and virtual passengers interacting with the transport system. In practice, the AFC system and the operations database will have to be updated almost immediately with the latest recorded data and then release it to the traffic management and the associated prediction module.

The output of the prediction model is the estimated OD demand of travellers starting their journey in the current (and possibly near-future) time interval(s).

The input to the assignment model comprises:

a. A network-specific set of paths covering all OD pairs, each path consisting of combinations of lines available to connect origin and destination. These lines must correspond to the current traffic service.
b. Specification of network-specific parameters of the statistical arrival distribution, which passengers are expected to follow at origin stations. The arrival distribution is in the form of a mixed Beta and Uniform distribution, corresponding to a mixture of passengers who are aware of the schedule (Beta) and passengers who are unaware (Uniform). Thus, the parameters must contain at least the mixture parameter (proportion of aware to unaware passengers) and the shape parameters of the Beta distribution. Note that these parameters may depend on the service headway.
c. Specification of network-specific route choice parameters corresponding to the average user. The journey attributes are in-vehicle travel time, waiting time, walking time (at transfers), and number of transfers.

These inputs should have a relation to the transportation network and its users, i.e., they should ideally be calibrated on empirical data. For instance, the pathset in point *a.* should be computed based on observed paths in the AFC data, the arrival distribution parameters in point *b.* should be calibrated using observed arrival times dependent on train departures, and the route choice parameters in point *c.* should be calibrated using a route choice model of users within the studied area.

The output of the assignment model is a passenger assignment plan (PAP) consisting of the predicted OD demand distributed on passenger groups, grouped by OD pair, arrival time at origin, and chosen path and containing train-specific itineraries including departure and arrival times as well as transfers.

3.3 Module setup and structure

The prediction model exists on two operational levels. The first level is the training model, which essentially calibrates the parameters of the model with the goal of achieving the best generalized performance at deployment. The training is carried out using a large training data set and a smaller validation data set, each consisting of features related to historical demand observations as well as observations on the reliability of supply, such as frequency of service, train delays, and cancellations. The structure of the data associates several features with each time interval for which a demand prediction is needed. These features include observed demand from time intervals immediately preceding the one(s) to be predicted, as well as older observations providing context about the historical demand patterns.

The features are extracted from the data such that no time interval features contain information that would only be available after the start of that interval, thereby simulating real-time information at each time interval. The training setup feeds the model with features for each time interval. The output of the model is compared to the target observed demand for the given time interval, and an optimization algorithm uses the deviation from the target to update the parameters of the model in an iterative manner. The generalized performance of the model is estimated by running the model on the validation data set, which is not used for parameter optimization. The training process is time-consuming but is only required to be re-run occasionally offline to update parameters with larger sets of new data.

The second operational level is the deployment model, which is used in real time with the same feature types to predict the demand for the current time interval. The deployment process is fast and is integrated with the traffic control algorithms in an online fashion.

As the control algorithms do not accept the OD demand directly as input and instead need information on the distribution of this demand over the trains in the system, an assignment model is built on top of the OD demand prediction model. The goal of the assignment model is to assign the OD demand for a time interval (e.g., 20 minutes) to specific trains in order to maintain information about the passengers on each train, the origins of the passengers, their destinations, and their personalized journey plan, including start and end times of journey and transfers.

The assignment module has several stages after the prediction model itself, see Figure 2.

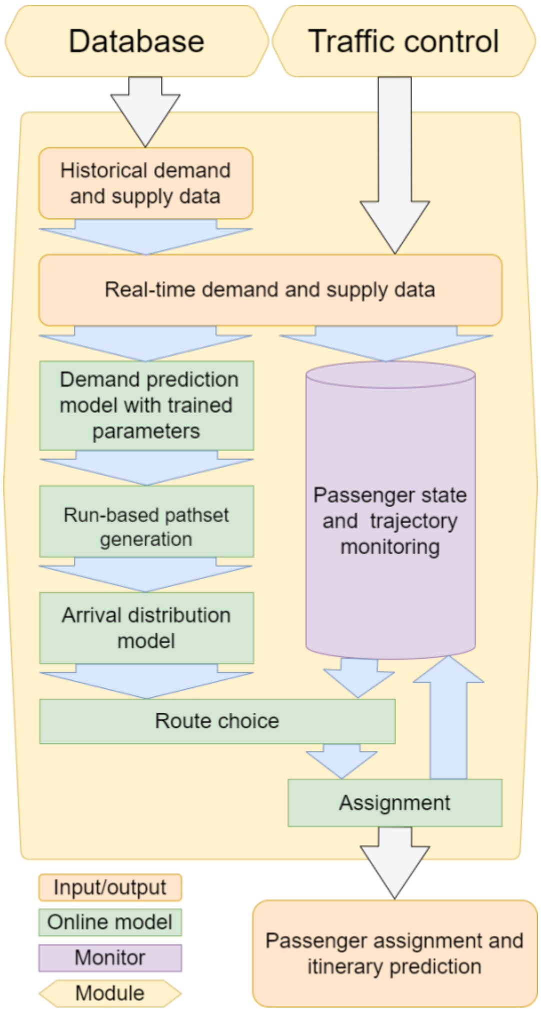

Figure 2: Demand prediction and assignment module

First, using knowledge on the static choice set of lines to take for a given OD pair, the model generates a pathset of specific train runs for each OD pair relevant to the passengers in the current time interval. This run-based pathset consists of paths representing journeys, each containing an itinerary or sequence of train journey legs. Each journey leg is defined by an entry station, an exit station, and a train ID. Additionally, the journey leg contains information about the departure time of the train from the entry station and arrival time at the exit station. This way, the transfers included in the itinerary can be inferred.

Then, based on the train departures at the origin station, the distribution of arrivals of the passengers at the station for the OD demand is estimated. This arrival time, while discretized, is notably more detailed than the interval for the OD prediction, e.g., to a 1-minute resolution. This will associate a subset of paths available to each group of passengers with the same OD and similar arrival time. Based on the previously computed choice set itineraries, the discrete choice attributes of each compatible pair of path and origin arrival time are computed. These attributes include in-vehicle travel time, waiting time, walking time at transfers, and number of transfers, which all influence the choice of one path over another.

Finally, the OD demand estimated by the demand prediction model is distributed on these paths according to a discrete choice model, specifically a multinomial logit (MNL) model, which stochastically simulates the choice of paths of passengers. However, instead of simulating a specific choice for each passenger, the overall distribution of demand based on the probabilities of the MNL model is used for the assignment of demand to each passenger group. The passenger groups are collected into a passenger assignment plan (PAP), which is the final output of the assignment module.

Each passenger group is defined by the OD pair, the arrival time at the origin station, and the path chosen with a non-negative real-valued number assigned, which represents the expected number of passengers in the passenger group. Furthermore, the passenger group includes an estimate of the "desired" arrival time at the destination, which is computed based on the assignment plan using the nominal timetable instead of the RTTP. The "desired" arrival time is the weighted mean of the arrival time at the destination for that assignment plan.

The set of passenger groups from the RTTP is used as input to the traffic management module, along with the information about the itineraries of those passenger groups.

3.4 Use

The model is called whenever an RTTP needs to be evaluated. This happens at set intervals at which the traffic management needs to resolve predicted conflicts. Concretely, it is first called to predict and assign demand for the RTTP as it is after letting the system run since the last evaluation. Then, the traffic management module produces one or several new RTTPs, and demand is predicted and assigned for each of these RTTPs for evaluation in the optimizer. In this way, the optimizer receives feedback about the impact of each RTTP on the demand. The information used for evaluating the RTTP is the difference between the estimated arrival times at the destination and the estimated *desired* arrival time at the destination. This difference is equivalent to the personalized delay of the given passenger group, and one of the optimization objectives is to minimize this delay. Furthermore, the keeping of passenger-specific transfers can be ensured or encouraged in the optimization based on the itineraries in the PAP.

3.5 Interfaces

The prediction model only interacts with the control architecture, receiving observed passenger states from the traffic state prediction (TSP) and a timetable corrected for current delay perturbations in the form of an RTTP. The observed passenger states are in the form of tap-ins and tap-outs in a CSV file passenger by passenger, while the RTTP is in the standard format General Transit Feed Specification (GTFS), a collection of CSV files containing routes, stops, trips, and stop times. In return, the prediction model sends the output PAP in the form of an XML file to the control architecture.

4 MODELING ARCHITECTURE

4.1 Brief Outlook

The origin-destination (OD) demand prediction problem is formulated as a supervised machine-learning problem. To capture spatial correlations between demand across OD pairs, we use a graph neural network (GNN) where each OD pair corresponds to a node in the graph. The graph neural network then computes information at the node level (OD-pair) that will be propagated through the neighbouring nodes in the graph. The input of the model consists of features describing the recent state of each OD pair in the rail network (recent demand for each OD pair, information about delays and cancellations, etc.) - node-specific (local) features, as well as other relevant context information – global features (weather data, information about the time of day, day of week and special holidays, etc.). The output of the model consists of the demand predictions for each node in the graph (OD-pair) for the next time step t+1. Figure 2 depicts the modeling approach. A detailed description of the modelling methodology is provided in Section 4.2. Kindly note that this is a joint model of the demand that jointly considers all the information from all OD-pairs to produce demand predictions for all OD-pairs at once.

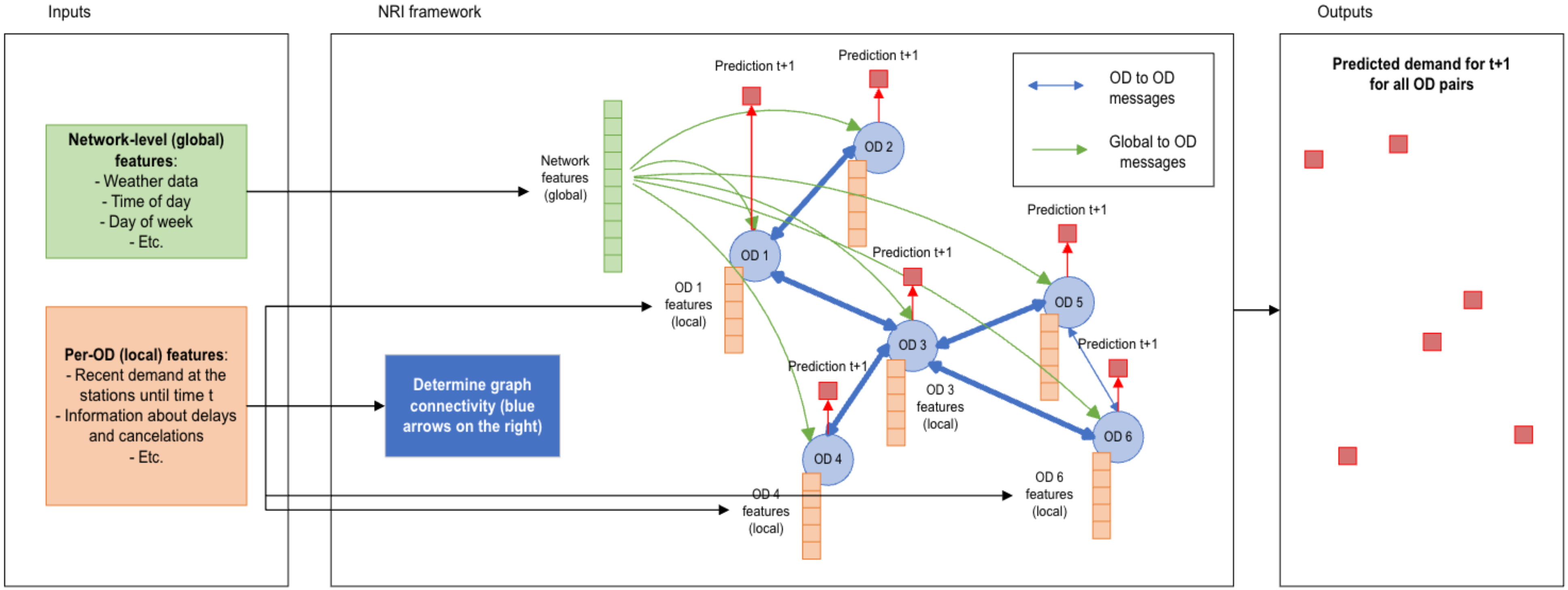

Figure 2: Overview of the GNN-based modelling approach used for OD demand prediction.

4.2 Description

A key challenge in applying GNNs for OD demand prediction is defining the graph connectivity. Recall from the previous section that, in our graph neural network formulation of the OD demand prediction problem, each OD pair corresponds to a node in the graph, and the connections between nodes represent dependencies/correlations between the observed demand in those nodes (OD pairs). While it is intuitive that the demand of a given OD pair should exhibit correlations with other OD pairs, there is no obvious way of pre-determining the dependencies in the graph - i.e., the graph adjacency matrix. A heuristic commonly used in the literature relies on spatial proximity between the origins and destination under the assumption that nearby stations should have similar demand. While this assumption is reasonable when considering aggregated demand at individual stations, its generalization to individual ODs is problematic since a given station may exhibit completely different demand patterns depending on the destination considered. Moreover, this approach assumes that the spatial correlations between demand at different OD pairs are static through time rather than dynamic.

To overcome this limitation and entirely bypass the problem of defining an adjacency matrix, we employed the neural relational inference (NRI) framework from Kipf et al (2018), where the graph adjacency matrix is determined dynamically in a data-driven way through the use of a neural network – we refer to this neural network as the "Encoder". The Encoder takes in as input the information (features) for each OD-pair and outputs the probability of each link being "active". Together with a predefined cut-off threshold for this probability (a hyper-parameter of the model), the Encoder then defines the graph that will be used by the GNN in the "Decoder" neural network – see Figure 3. For each time-step t, the Decoder takes as input the global and local features described in Section 4.1 and the dynamic graph produced by the Encoder, and outputs a prediction for the demand at time t+1 for all nodes. In order for the Decoder to be able to model the temporal dependencies, we utilize a recurrent neural network, namely a Gated Recurrent Unit (GRU) cell, in the Decoder. The latent state h of the GRU at each time step t is determined based on message-passing neural architecture used by the

Graph Network framework proposed in (Tygesen, Pereira, & Rodrigues, 2023). At each time step t, each node computes a message (a numeric vector of a predefined length – a hyperparameter) to be sent out to its neighbours using a fully connected neural network. Each node then aggregates all incoming messages (vector) from its neighbours using summation. The aggregated vector is then provided as input to the GRU unit to determine the next latent state h for each node. A detailed description of this NRI modelling approach is provided (Tygesen, Pereira, & Rodrigues, 2023).

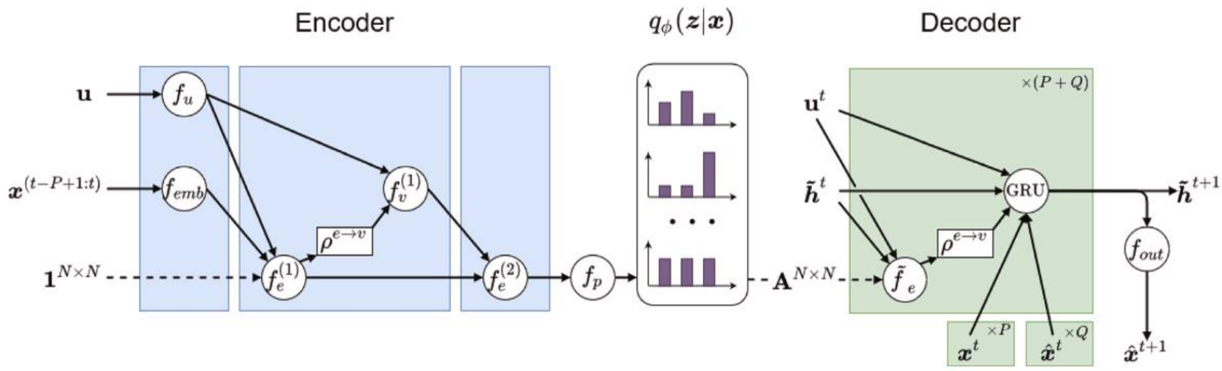

Figure 3: Architecture of the NRI model used for OD demand prediction. u denotes global features, x denotes local features, q(z|x) represents the probability distribution of each link being active, while A denotes the adjacency matrix derived from q(z|x). The Decoder uses a GRU unit to model the latent state h at each time step t.

4.3 Benchmarks

Graph neural networks are state-of-the-art for spatiotemporal prediction problems in transportation (and also other domains), including OD-demand prediction. The literature is vast on research papers demonstrating the superiority of GNNs over more traditional methods such as linear regression, auto-regressive models, historical average models, etc. (Tygesen, Pereira, & Rodrigues, 2023). Therefore, our main focus is on understanding the impact of different types of input features on the prediction error. Nevertheless, we still compare our NRI approach with a vanilla graph convolutional network (GCN), as well as with the state-of-the-art approach for OD demand prediction (referred to as STZINB) as proposed in in (Zhuang, Shenhao, Koutsopoulos, & Zhao, 2022).

5 CASE-STUDY

5.1 Overview

The case study used for the development and evaluation of the demand prediction module is the Copenhagen suburban railway, the “S-bane”. This railway network consists of 170 km of electrified double track with homogenous passenger traffic, which is entirely separated from the long-distance rail network. The annual number of passengers was 58.8 million in 2019 (Trafikstyrelsen, u.d.). The network is owned and managed by Banedanmark, operated by the national operator DSB, and has since September 2022 been running 100% on the communications-based train control (CBTC) system, which enables autonomous operation of trains, although this has not yet been realized. In addition to the train control system, the network is well integrated with Copenhagen's multi-modal transportation system, consisting of regional, suburban, and urban rail (Metro), with buses filling the gaps.

All public transportation in the region is integrated into a zone-based fare system with almost entirely homogenous prices for all fares, the Metro being the exception. A common way of paying the fare is via the nationwide AFC system “Rejsekort”, which works as a smart card, where the user taps in and out of the system, and the fare is automatically calculated and charged. It is possible to combine the card with a commuter card. The locations and time stamps of the tap-in and tap-out (including transfers between modes) are recorded in a central database from which OD demand can be extracted. Although “Rejsekort” does not account for all demand in the network, it does represent the main fare collection source over short and medium distances in the Copenhagen region.

Banedanmark, being responsible for the infrastructure, the timetable, and the traffic management, keeps records of the performance of traffic on the network, including timings of all train arrivals at stations. These data are kept in a database called “RDS” and contain information about each specific train arrival, including the train number, the direction of travel, the scheduled time of arrival, and the deviation from the schedule.

Furthermore, the data indicate whether a train was cancelled. These data can be collected into time- and station-specific performance indicators pertaining to traffic volume, punctuality, and reliability.

5.2 Data

The "Rejsekort" data consists of smart card transactions with each line being a transaction. A transaction is an interaction between a card and a smart card terminal. The terminal might be located on a station platform or inside a vehicle. There are two types of such terminals, namely tap-in terminals and tap-out terminals. There are several types of transactions, which can broadly be divided into three main types, namely entry, transfer, and exit. Entry is a tap-in at the start of the journey, while exit is a tap-out at the end of the journey. Transfer is a tap-in during the journey, i.e., any tap-in happening after an initial tap-in but before a tap-out. It is possible to transfer by tapping out and tapping in again within a certain amount of time, continuing the journey as if it were simply another successive tap-in.

A journey is defined as a series of transactions starting with an entry and ending with an exit transaction. Each trip is assigned a trip ID. Since "Rejsekort" covers several modes and operators, each terminal is associated with a certain mode and operator, e.g., a terminal in a bus is assigned the mode "Bus" and the operator of the bus, while a terminal at a suburban rail station is assigned the mode "S-Train" and "DSB S-Train" as the operator. It is possible to track each individual card by a card ID, although this is pseudonymised, so it cannot be tied to a specific customer directly. Importantly, each transaction is associated with a stop name and ID, which is identical to the stop ID in the GTFS. Furthermore, each transaction is associated with a time stamp in local time. This timestamp is tied to the internal clock of the terminal, which has some uncertainty, especially if it is on board a vehicle.

Using all the journey entry and exit transactions, we aggregate demand for each time interval, and each OD pair into the number of passengers travelling from a given origin to a given destination, having started their journey within a given time interval. The result of this process is used as the "true" demand with which the model will compare its own output and from which it "learns". However, the model also takes as input the observable demand for recent time intervals, i.e., at time interval t, the OD demand for time intervals $t-1$, $t-2$, etc., will be used as input, discounting the journeys that haven't been concluded at the start of time interval t. Thus, the aggregation is also done for each pair of time intervals that are at most a certain number of time steps apart, also known as "lags".

The" RDS" data consist of timings of train arrivals at stations and are processed to collect information about reliability at each station during each time interval. Specifically, for each station, train line, and direction, we aggregate the total number of trains scheduled for arrival at the station, the mean deviation from the schedule in seconds, and the proportion of trains that were cancelled, all during the given time interval. These are used as supply input in the model.

5.3 Experimental set-up

The training and testing of the model are based on predicting OD demand for adjacent but non-overlapping 20-minute intervals. For the development of the prediction model, the data are split into a training set and a validation set for each experiment. In order to best discern differences in performance between models, an extensive period is selected for training and validation, respectively, and only the OD pairs carrying the most passenger volume, and thus with the most variation in demand, are selected for training.

In Copenhagen, twelve OD pairs are selected:

1. Nordhavn-Nørreport
2. Nørreport-København H
3. Lyngby-Nørreport
4. Nørreport-Nordhavn
5. København H-Nørreport
6. Nørreport-Lyngby
7. Hillerød-Nørreport
8. Svanemøllen-Nørreport
9. Hellerup-Nørreport
10. Nørreport-Svanemøllen

11. Nørreport-Hillerød
12. Østerport-Nørreport

These constitute the twelve OD pairs with the highest average demand. Many of the stations reoccur, which, on the one hand, narrows the focus of the model and risks poor generalization of performance, but which, on the other hand, represents OD pairs which can interact through their similarity and thus capture replacement dynamics amongst each other.

The training data consists of 52 weeks from January 29, 2017, through January 27, 2018, i.e., the model will see an entire year of data. The validation data consists of 20 weeks immediately following the training data period, i.e., from January 28 2018 through June 16 2018. There are 72 time intervals of 20 minutes in a 24-hour period. Since most of the demand happens during the daytime, only the period 6 am through 10 pm is used, which amounts to 51 time intervals. All seven days of the week are included, which means that the training data contain 52x7x51=18564 time intervals, each with 12 OD pairs. Similarly, the validation data contain 20x7x51=7140 time intervals. The model operates with eight lags of demand data.

Kindly note that the training of the model should be done separately when varying either the period of the day to be investigated, the subset of OD pairs, or any combination of input variables.

6 RESULTS

Table 1 shows the results for the different models for the 12 OD pairs described in the previous section. We report root mean squared error (RMSE) and mean absolute error (MAE) as metrics of prediction quality. Our analysis starts with the comparison between different models, namely GCN, STZINB and NRI. To simplify the comparison, we consider only the case where only information about the observed OD demand for the previous time steps (lags) is used as input for the models. As the results in Table 1 demonstrate, the NRI model obtained the lowest prediction error of the three. Interestingly, STZINB obtained the worst results, being even outperformed by a simple GCN model.

After observing that the NRI model was providing the best results, we performed an ablation study on the input features of the model in order to understand their impact on the model's prediction error. From the results presented in Table 1, we can observe that, as expected, adding information to the neural network model generally leads to better prediction performance. Note that each row on results builds on the features of the previous row – i.e., "NRI - adding supply features" also includes "node ID" features. Interestingly, we can observe that weather and node ID features produce the greatest improvements in prediction performance. This suggests that the weather has an important impact on demand and that it is important for the graph neural network model to distinguish between different OD pairs (node ID) when propagating information in the graph – i.e., the propagation becomes conditional on which nodes are involved, rather than being generic across the graph. Unsurprisingly, including supply features as input to the model does not lead to a noticeable improvement in overall prediction performance (in fact, the prediction error is slightly worse). This is expected since supply disruptions are very rare events, and therefore, improvements made regarding modelling the impact of those events won't be noticeable when analyzing the aggregate prediction error over the entire period of the test set.

	RMSE	MAE
GCN - with lags only	5.006	3.458
STZINB – with lags only	5.159	3.663
NRI - with lags only	4.921	3.449
NRI - adding weather features	4.887	3.454
NRI - adding day of week and time of day features	4.853	3.413
NRI - adding node ID features	4.819	3.382
NRI - adding supply features	4.824	3.390

Table 1: Prediction error statistics for the different models considered.

In order to better understand the impact of the supply features on the prediction error of the model, we analysed the RMSE and MAE for different time periods. Namely, we investigated the prediction error statistics for periods when there were cancellations at either the origin (O) or the destination (D). The results are presented in Table 2. As one can observe, for these time periods, there is a positive impact in including supply features (except for periods when the number of cancellations at origin is greater than 0).

	RMSE	MAE
NRI without supply (all periods)	4.819	3.382
NRI without supply (when cancellations at origin > 0)	5.287	3.897
NRI without supply (when cancellations at origin > 1)	5.349	4.195
NRI without supply (when cancellations at destination > 0)	5.303	3.983
NRI without supply (when cancellations at destination > 1)	5.084	3.926
NRI with supply (all periods)	4.824	3.390
NRI with supply (when cancellations at origin > 0)	5.362	3.986
NRI with supply (when cancellations at origin > 1)	5.296	4.150
NRI with supply (when cancellations at destination > 0)	5.215	3.877
NRI with supply (when cancellations at destination > 1)	4.993	3.858

Table 2: Prediction error statistics for the NRI model with and without supply information at different time periods.

7 EXTENSIONS

7.1 Tiny CPH

In order to carry out experiments on the case study, two versions of the Copenhagen case study have been carried out, namely "tiny CPH" and "full CPH", the former encompassing 12 contiguous stations in the network for preliminary testing and development, and the latter encompassing all 84 stations in the 2017 version of the network.

Tiny CPH consists of the area to the north of the centre of Copenhagen, starting with Nordhavn on the trunk line and fanning out to Ryparken, Sorgenfri, and Charlottenlund stations on the three northern fingers of the network carrying lines A, B, Bx, C, and E. Additionally, the ring line carrying line F is included between Nørrebro and Hellerup. This area includes all lines except line H, thus having a complex pattern of traffic with lines converging and diverging, sufficiently large demand volumes, and retaining the need for transfers for certain OD relations.

With twelve stations and, therefore 132 OD pairs, the network is an order of magnitude larger than the dataset used for developing the prediction model. This entails heavier processing of input features and possibly extended training time along with marginally longer prediction time. If confronted with extensive slow-downs, it may be necessary to optimize certain areas of the framework. The data period used for training and validation may also be narrowed.

7.2 Full CPH

The entire network consists of 84 stations, meaning that as many as 6972 OD pairs must be predicted for each time interval. This may be too much to handle in reasonable time and memory when training the prediction model and may slow down prediction. If code optimization is insufficient to mitigate the extra workload, it is possible to cut down on the number of OD pairs that need to be considered. This could, for instance, involve removing the OD pairs for which the average demand is below a certain threshold and using other assumptions for them.

8 BIBLIOGRAPHY

Albrecht, T., Binder, A., & Gassel, C. (2012). Applications of real-time speed control in rail-bound public transtransport systems. *IET Intelligent Transportation Systems*.

Sobieraj, S., Rodriguez, J., & Marliere, G. (2011). Simulation of solution of a fixed-speed model for the real-time railway traffic optimisation problem. *4th International Seminar on Railway Operations Modelling and Analysis RailRome.*

Jaekel, B., Albrecht, T., & Thonig, F. (2013). Comparative Analysis of Algorithms for Train Running Simulation. *5th International Seminar on Railway Operations Modelling and Analysis RailCopenhagen2013.*

Trafikstyrelsen. (u.d.). *Danish Transport Authority passenger numbers*. Hentet 15. 11 2023 fra https://passagertal.dk/

Kipf, T., Fetaya, E., Wang, K. C., Welling, M., & Zemel, R. (2018, July). Neural relational inference for interacting systems. In *International conference on machine learning* (pp. 2688-2697). PMLR.

Tygesen, M., Pereira, F., & Rodrigues, F. (2023). Unboxing the graph: Towards interpretable graph neural networks for transport prediction through neural relational inference. *Transportation research part C: emerging technologies*.

Zhuang, D., Shenhao, W., Koutsopoulos, H., & Zhao, J. (2022). Uncertainty Quantification of Sparse Travel Demand Prediction with Spatial-Temporal Graph Neural Networks. *KDD '22: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining.*