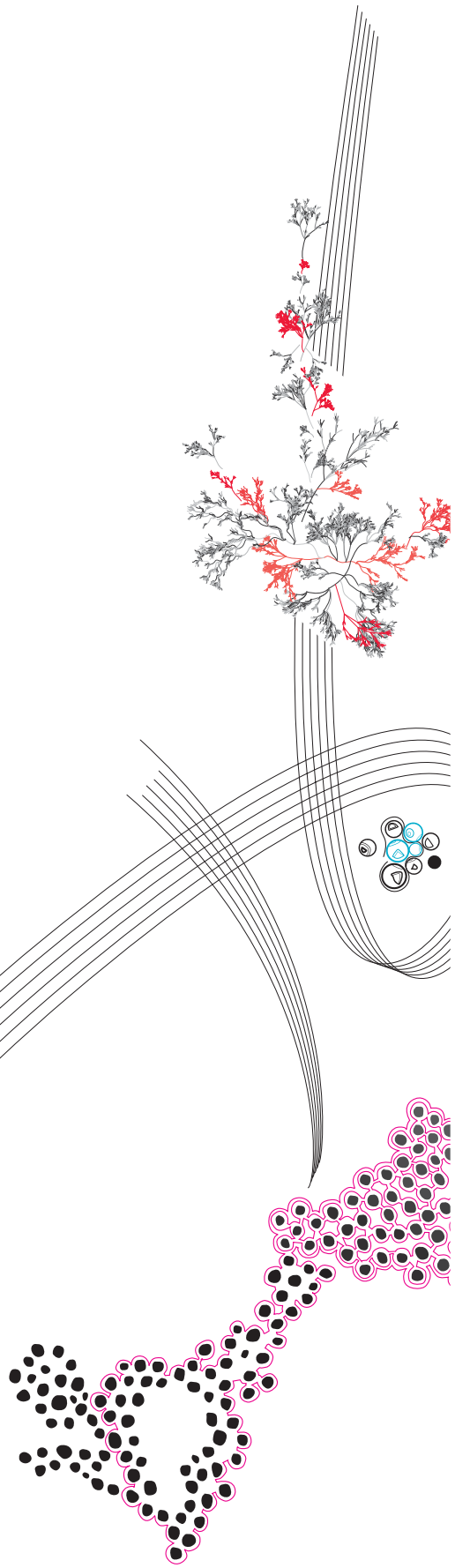


MSc Computer Science
Final Project

Short-Term Passenger Flow Forecasting in Public Transportation Networks Under Event Conditions

Jeffrey Bakker
research@jeffreybakker.me

Supervisors:

University of Twente	dr. Elena Mocanu	e.mocanu@utwente.nl
	dr. Nicola Strisciuglio	n.strisciuglio@utwente.nl
	dr.ir. Rom Langerak	r.langerak@utwente.nl
Info Support B.V.	Tom van den Berg	tom.vandenberg@infosupport.com
	Maarten Vos MSc.	maarten.vos@infosupport.com

2 November 2023

Department of Computer Science
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

infoSupport Solid Innovator

Picture by Jeffrey Bakker.

ABSTRACT

Public transportation networks are an essential part of public infrastructure and have the potential to reduce society's dependency on personal cars. Making public transportation a more attractive option requires better passenger demand forecasts, as these can be used to guarantee enough seating capacity for everyone, especially under event conditions. This Master thesis looks into short-term passenger flow forecasting under event conditions by comparing the performance of forecasting algorithms under event conditions and looking at the impact of including event features in the forecasting approach.

Even though passenger demand in public transportation tends to be quite regular, accurately forecasting the additional peaks of passengers caused by large events has proven to be quite challenging. This research uses the types of events and their venues' capacities as indicators of their actual attendance; however, these indicators have proven to be insufficient in the context of passenger flow forecasting. Some cherry-picked examples show promising results for future works that have access to the correct data.

KEYWORDS

Crowdedness, Events, Occupancy Forecasting, Passenger Flow Forecasting, Public Transportation

1 INTRODUCTION

Many people rely on public transportation networks on a daily basis; it is an essential part of the public infrastructure, which – if implemented well enough – can reduce our collective ecological footprint as more and more people switch to this more environmentally friendly mode of transport [8, 29]. However, getting more people to choose public transportation over their cars requires governments and transportation companies to make it a more attractive option.

Related works have concluded that factors such as fares [1], service reliability [1], and crowdedness[1] [29, 30] have significant impact on passenger demand. Crowdedness (or relative occupancy) can be managed by, for example, scheduling more or fewer materials, like altering the length of trains or service frequency. Furthermore, it is in the best interest of transportation companies to properly balance crowdedness and deployed materials as these directly relate to long-term passenger demand and costs.

[1]We define crowdedness to be the relative occupancy compared to the capacity of passengers, measured by the number of seats.

Properly managing the crowdedness in public transportation requires accurately anticipating future passenger demand, also known as passenger flow forecasting. However, the patterns of passenger demand can be significantly influenced by large events like concerts, festivals, sports matches, et cetera [3, 12, 14, 22, 41]. These effects will only become more prominent as event organizers stimulate visitors to travel by public transportation instead of by car; one example of this is one of the big Dutch festivals, "Lowlands", which encourages visitors to travel by public transportation and even organizes bus transport for attendees between the nearest train station and the festival terrain [18].

Even though there has been some research into passenger flow forecasting in public transportation under event conditions, most of the current research focuses on a single station [21, 41], or departures/arrivals [21, 41] instead of forecasting occupancy of inter-station connections in a larger network under event conditions. Therefore, this research sets out to compare the performance of various passenger flow forecasting algorithms under event conditions and the impact of the selected features describing nearby events. Furthermore, a Graph Neural Network approach is implemented with the anticipation that it is more capable of dealing with the spatial impact of events on passenger flow.

If successful, public transportation network operators could use the approaches discussed in this research to improve their passenger demand forecasts around events in order to be able to accommodate the additional influx of passengers, making trips more pleasant for all passengers and potentially increasing passenger demand in the long term.

The rest of this document is structured as follows: First, a more formal problem definition is given together with concrete research objectives later in this introduction. Then, a background about passenger flow forecasting problems and Machine Learning with Artificial Neural Networks is given in Section 2, followed by an overview of related works tackling passenger flow forecasting under event conditions in Section 3. From there on, we first discuss the influence of the capacity of an event's venue on the observed passenger flow in Section 4, the conclusion of which is used in Section 5 to motivate choices in the configuration of the selected algorithms. The results from the different forecasting algorithms tested in this research are shown in Section 6, followed by a discussion of the limitations of these results in Section 7. Finally, we have the conclusion of this thesis in Section 8 and some opportunities for future works in Section 9.

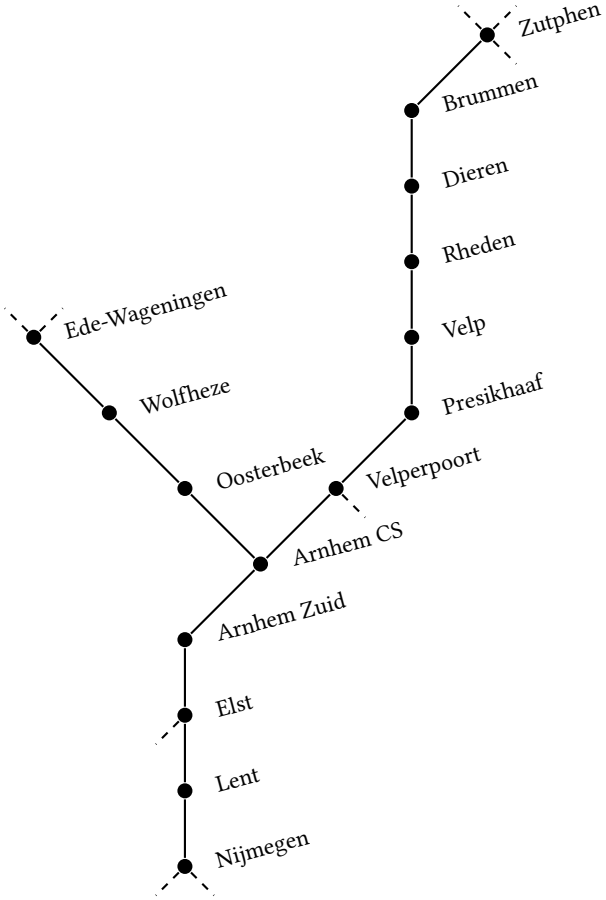


Figure 1: Subgraph of the Dutch train network near Arnhem.

1.1 Problem Statement

Public Transportation Networks consist of stations and connections between those stations. Passengers can move through the network along the connections between the stations, where they can either board, transfer to another route, or alight. Figure 1 shows an example of a subsection of such a network, namely a part of the Dutch train network.

More formally, public transportation networks (like a train network) describing how passengers move through the network can be represented by a bidirectional planar graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \mathcal{F} \rangle$, where:

- $\mathcal{V}$ is the set of vertices (stations);
- $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges (connections between stations); and
- $\mathcal{F} : \mathcal{E} \to \mathbb{N}$ is a function mapping between occupancy (number of passengers) and edges. Throughout this research observations of $\mathcal{F}$ will be indicated by y, and their approximations by $\widehat{y}$.

An example of a bi-directional flow graph $\mathcal{G}$ is given in Figure 2, where the labels of edges $e \in \mathcal{E}$ correspond to $\mathcal{F}(e)$.

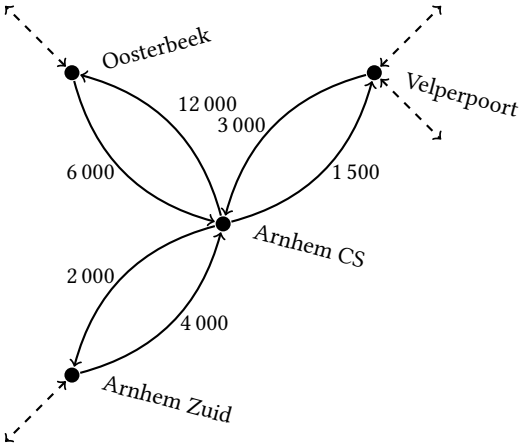


Figure 2: Example bi-directional flow graph on a subgraph of the Dutch train network; the edge label for an edge $e \in \mathcal{E}$ corresponds to the occupancy $\mathcal{F}(e)$ of a fictional observation.

Furthermore, since the bi-directional flow graph represents a public transportation network, which generally offers transportation in both directions along a route, it also has the following property:

Lemma 1. *For a directed graph $\mathcal{G}$ representing a public transportation network, an edge $(v_1, v_2) \in \mathcal{E}$ implies there also exists an edge $(v_2, v_1) \in \mathcal{E}$:*

$$\forall_{v_1, v_2 \in \mathcal{V}} \quad (v_1, v_2) \in \mathcal{E} \implies (v_2, v_1) \in \mathcal{E}$$

In order to optimally balance the deployment of materials and crowdedness, transportation companies need to anticipate how passengers move through their networks. In other words, they need to know how many passengers will traverse the directed edge $e \in \mathcal{E}$ during a specific time window in the future. More specifically, the forecasting target is the number of passengers $\mathcal{F}(e)$ for every single edge $e \in \mathcal{E}$. The goal is to be able to make accurate short-term (up to 72 hours) passenger flow forecasts for a public transportation network in order to make last-minute adjustments to the schedule and show Real Time Crowding Information to passengers [8].

Current solution. The solution of one of the larger public transportation companies in the Netherlands consists of two different approaches: On the one hand, there is a LightGBM regression model to forecast the number of passengers on a train. On the other hand, an LSTM-based regression model is used to forecast Origin-Destination flows, which are then mapped to the train schedule to predict how passengers move through the train network. However, only a limited number of exogenous features are included and only play a minor role in the forecasts; events are not taken into account in the current forecast approaches yet and are adjusted for manually.

1.2 Research Objectives

As mentioned earlier, while there is much research into passenger flow forecasting, research into passenger flow forecasting in public

transportation networks under event conditions is limited, especially for inter-station occupancy as forecasting target. Therefore, this research aims to measure the impact of the selected event-features and to compare the performance of the selected passenger flow forecasting algorithms to determine which performs best under both "normal" and event conditions. Therefore, this research sets out to answer the following research questions:

RQ1 Which passenger flow forecasting algorithm has the best overall performance under both "normal" and event conditions?
- **RQ1.1** What is the minimum venue capacity of an event for it to have a significant impact on occupancy in public transportation networks?
- **RQ1.2** What is the impact on the performance of passenger flow forecasting when including events in the exogenous forecasting features?

The overall hypothesis for this research is that including event-information in forecasting will positively impact the performance of passenger flow forecasting algorithms under event conditions.

2 BACKGROUND

Passenger flow forecasting can be classified as a *spatio-temporal regression problem* [6], where passenger flow can refer to arrivals and departures at stations [22, 34, 36, 42], origin-destination (OD) flows [38], demand in the number of trips per time-step [12, 14], or inter-station crowdedness. There has been much research into passenger flow forecasting recently, where different related works look into passenger flow forecasting in different modes of travel like road traffic [11, 14, 43], ride-hailing [11, 12], metro [3, 11, 22, 34, 35, 42], or trains [11].

Forecasting Algorithms. Many different works researching passenger flow forecasting have tried many different algorithms, like the Autoregressive Integrated Moving Average (ARIMA) and its variants [37], Linear Regression (LR) [21, 43], Support Vector Machines (SVM) [11, 37, 42, 43], Artificial Neural Networks [11, 43], Recurrent Neural Networks [11, 12, 37, 41], Graph Neural Networks (GNN) [11, 37], Temporal Convolution Networks [37], and many more.

Overall, the consensus appears that statistical and traditional Machine Learning methods are easy to interpret and their results relatively easy to explain. However, it is theorized that these methods would lack the capability to model complex and non-linear relationships in the spatial and temporal domains [11]. Therefore, recent related works mainly use Deep Learning approaches; more specifically, Graph Neural Networks have been described as the state-of-the-art for passenger flow forecasting problems because of their capacity to model the spatial dependencies in transportation networks [11, 37].

Manibardo et al. [17], however, are sceptical of these developments and argue that not every passenger flow forecasting problem requires a Deep Learning approach with millions of trainable parameters. Their work concludes with a set of recommendations for future passenger flow forecasting research, like taking the time to reason about the need for deep learning methods to solve "complex" relations in the data and comparing the results against a baseline in order to determine whether the improvements are statistically significant [17].

The forecasting algorithms used in this research are SARIMA, Hybrid forecasting models (see Section 3), and a GNN approach.

Time step resolution. The proper time step resolution selection is essential in any time series forecasting problem, as smaller resolutions tend to be noisier. In comparison, larger resolutions will obfuscate the peaks in demand that one might be trying to predict. The importance of the selection of the proper time step resolution is confirmed by Zhang [40], who argues that – for an accurate and reliable passenger flow forecast – an appropriate time granularity needs to be chosen that is as small as possible while also providing regularity and stability in the measurements. To measure this regularity, they use the Pearson Correlation Coefficient between time steps at the same time of the day over different days (making distinction – of course – between weekdays and weekends) such that a higher correlation indicates better regularity. Zhang [40] concluded that the often-used time granularity of 15 minutes in subway passenger flow forecasting meets the requirements well enough for that type of public transportation network.

The ideal time step resolution for passenger flow forecasting (used for optimizing short-term service based on expected demand) depends on the mode of travel and, therefore, the service interval. Time step resolutions used in related works are 5 minutes [34], 10 minutes [3], 15 minutes [22, 34], 30 minutes [34], 1 hour [14, 42], and 1 day [12].

In practical applications, however, the selection of the time step resolution is often restricted by constraints posed by available data. This research has one dataset with a time step resolution of 30 minutes and one of 1 hour.

Factors impacting demand. Balcombe et al. [1] researched the important socioeconomic factors influencing the demand for public transport across different modes. Their findings concluded that, among others, fare prices, income, car ownership, population density, service intervals, service reliability, and station quality (i.e. shelter from the weather, comfort, cleanliness, and safety) all influenced the demand for public transport. Furthermore, they discussed the correlation between income and car ownership and that it is hard to separate the impact of these individual factors on demand as they often go hand-in-hand [1]. However, this research is from 2004, meaning that the amount some of these factors influence demand might have changed as the world around us has changed.

Some commonly used external factors by related works for forecasting passenger flows are weather [12, 14, 22, 33, 37], land use [35], and holidays [15, 37]. Furthermore, some related works also use indicators for short-term changes in passenger flow demand like social media posts [21, 34], or views of an app or website [12].

In order to limit the scope, this research solely focuses on event indicators as external factors in the forecasting algorithms.

Datasets. Many related works researching passenger flow forecasting in public transportation use passenger travel data collected with Automatic Fare Collection (AFC) systems [43]; such systems often register passengers' origins, destinations, departure- and arrival timestamps per trip. Zhu et al. [43] mentions that the reason why Big Data collected from AFC systems is often used in research

towards Intelligent Transportation Systems (ITS) is its "potential capacity of offering comprehensive spatial-temporal information on travel behaviour".

Zhu et al. [43] observed that – even though theoretically a lot of data should be available from AFC systems – most existing literature on passenger flow forecasting base their experiments on datasets that span less than a year. Therefore, it is often uncertain how those models perform throughout the year (or over the span of multiple years).

In this research, we have one dataset that spans more than a year and a smaller one.

Metrics. Common metrics for evaluating the forecast accuracy across many related works are the *Mean Absolute Error* (MAE), *Mean Absolute Percentage Error* (MAPE) and the *Root Mean Squared Error* (RMSE) with approximation $\widehat{y_i}$ of observed passenger flow y_i for n measurements [3, 11, 21, 22, 34, 41]:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}| \tag{1}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \widehat{y_i}|}{y_i}\cdot 100\% \tag{2}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \tag{3}$$

The MAE metric indicates how much the average approximation $\widehat{y}$ differs from the observed value y. Similarly, the RMSE metric also indicates how the average approximation differs from the observed value, but the RMSE metric weights larger errors more than smaller errors. Then, MAPE is a relative metric and measures the average forecast error as a percentage of the observation; this metric, however, is highly dependent on the scale of the observations and is, therefore, more helpful in comparing the performance of different outcomes under the same circumstances, rather than comparing different circumstances.

This research compares the different forecasting approaches using the MAE, MAPE, and RMSE.

2.1 Regression with Artificial Neural Networks

Within Machine Learning, there are typically two types of problems, namely *classification* or *regression*:

Classification The goal of a classification problem is to distinguish between any amount of predefined classes.

Regression The goal of a regression problem is to estimate the value of a (most likely continuous) variable.

As mentioned before, passenger flow forecasting is a regression problem, which – in this research – is approached using *Artificial Neural Networks* (ANN) [39] (or approaches generalized to ANNs). This subsection explains the mathematical basis behind this Machine Learning approach, whereas the next subsection (Section 2.2) explains how Artificial Neural Networks are trained using *Gradient Descent*.

Artificial Neural Networks are – as the name suggests – a network made up of artificial neurons, which are – in term – inspired by how the human brain works [39]. Such neurons take multiple

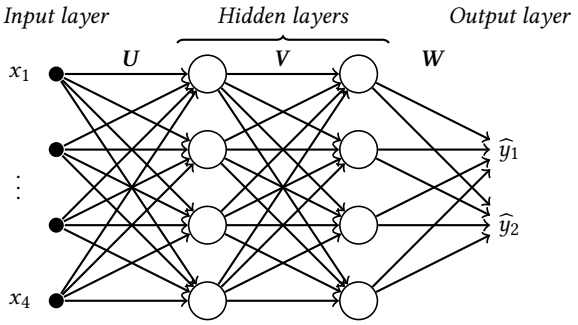


Figure 3: Example of a Multi-Layer Perceptron with two hidden layers and weight matrices U, V and W.

continuous variables as inputs, each of different importance; then, if the aggregated weighted inputs exceed a predefined *activation threshold*, the neuron *activates* and gives an output. In practice, each artificial neuron also has a weighted *bias* term to make it easier or harder to overcome the activation threshold; these biases are excluded from this explanation to reduce its complexity. When excluding the bias term for the artificial neurons, the mathematical equation for a single neuron looks like this:

$$h(\boldsymbol{x}) = \sigma\left(\sum_{i=1}^{|\boldsymbol{x}|}\boldsymbol{w}_i\cdot\boldsymbol{x}_i\right) = \sigma\left(\begin{bmatrix}w_1\\ \vdots\\ w_n\end{bmatrix}^T\cdot\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix}\right) = \sigma\left(\boldsymbol{w}^T\cdot\boldsymbol{x}\right) \tag{4}$$

where:

- $\boldsymbol{x}$ is a vector of input variables;
- $\boldsymbol{w}$ is a vector of weights, such that $|\boldsymbol{w}| = |\boldsymbol{x}|$; and
- $\sigma : \mathbb{R} \to \mathbb{R}$ is an activation function.

The activation function σ maps input space $\mathbb{R}$ onto output space $\mathbb{R}$ and simulates the aforementioned "activation threshold" and neuron activation. Among many potential activation functions, the activation function used in this research is the popular *Rectified Linear Unit* (ReLU) [39]:

$$\sigma(z) = \max(0, z) = \begin{cases}0, & \text{if } z < 0\\ z & \text{otherwise}\end{cases} \tag{5}$$

What we have described so far (minus the activation function) is basically Linear Regression; however, the real strength of Artificial Neural Networks comes in when combining multiple neurons to form a "layer" and multiple layers to form the network. The mathematical equation for a layer of neurons then looks like this:

$$h(\boldsymbol{x}) = \sigma\left(\boldsymbol{W}\cdot\boldsymbol{x}\right) \tag{6}$$

where $\boldsymbol{W} \in \mathbb{R}^{n\times|\boldsymbol{x}|}$ is a matrix of weights such that vector size of output vector $h(\boldsymbol{x})$ is equal to n. When combining multiple layers, the last layer is called the *output layer*, whereas all other layers are called *hidden layers*. It is common not to have an activation function on the output layer. An example of a *Multi-Layer Perceptron* (MLP), an Artificial Neural Network with one or more hidden layers, is given in Figure 3. The equation of this example MLP is as follows:

$$\widehat{\boldsymbol{y}} = \boldsymbol{U}\cdot\sigma\left(\boldsymbol{V}\cdot\sigma\left(\boldsymbol{W}\cdot\boldsymbol{x}\right)\right) \tag{7}$$

where:

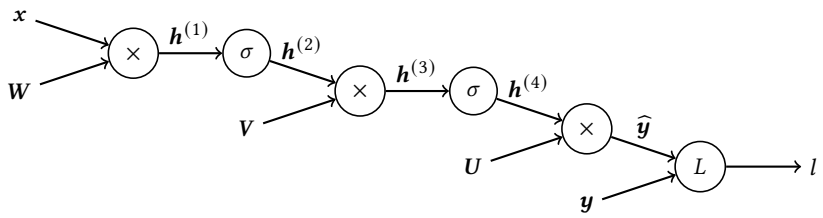


Figure 4: Computational graph of a forward pass of an MLP with two hidden layers (Eq. 7) and loss function L.

$\widehat{\boldsymbol{y}}$ is the approximation of the target outcome vector $\boldsymbol{y}$; and $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$ are matrices of weights.

Throughout this subsection, the term weights was thrown around a lot without mentioning how these weights are obtained: These weights are estimated by observing sample data and – in this research – "trained" specifically with Gradient Descent, which is covered in the next subsection. This training or learning step is why Artificial Neural Networks are sometimes referred to as *learnable functions*, often represented by ϕ, ψ or θ.

2.2 Learning with Gradient Descent

As hinted at in the previous subsection, one of the ways to "learn" weights for Artificial Neural Networks is *Gradient Descent* [39]. Gradient Descent essentially consists of iteratively tweaking all *trainable parameters* (or weights) to optimize the outcome of a given *loss function* by computing the gradients of every trainable parameter with respect to the computed loss for a training sample and applying those gradients to the parameters. This subsection explains the mathematical basis behind Gradient Descent.

Forward propagation. The previous subsection (Section 2.1) explained the mathematics behind a Multi-Layer Perceptron; to obtain the "optimal" weights, the output of a loss function L will need to be minimized. The loss function used in this research is the *Mean Squared Error* (MSE, see Eq. 20 in Section 5.3); so comparing the estimated output vector $\widehat{\boldsymbol{y}}$ to the correct value vector $\boldsymbol{y}$ will yield a loss l:

$$l = L(\boldsymbol{y}, \widehat{\boldsymbol{y}}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \widehat{y}_i)^2 \tag{8}$$

The entire computation from input to loss is called a *forward pass* and is visualized in Figure 4, where each of the (hidden) intermediate outcomes is labelled with $\boldsymbol{h}^{(1)}$ through $\boldsymbol{h}^{(4)}$.

Backpropagation. In order to correctly update the weights, we need to know how much each of the trainable parameters, which are in this case weight matrices $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{W}$, contributed to the obtained loss l. This is done by doing the computation in reverse and computing the partial derivative of the loss with respect to each trainable parameter; the process of computing these derivatives (or gradients) is called *backpropagation*.

The gradient of each set of weights can be computed by applying the product rule on the partial derivatives of each of the intermediate steps of the computation. So the gradient of weights matrix $\boldsymbol{W}$ is equal to the partial derivative of the computed loss l with respect to the $\boldsymbol{W}$, and can be calculated using the following equation:

$$\frac{\partial l}{\partial \boldsymbol{W}} = \frac{\partial l}{\partial \widehat{\boldsymbol{y}}} \frac{\partial \widehat{\boldsymbol{y}}}{\partial \boldsymbol{h}^{(4)}} \frac{\partial \boldsymbol{h}^{(4)}}{\partial \boldsymbol{h}^{(3)}} \frac{\partial \boldsymbol{h}^{(3)}}{\partial \boldsymbol{h}^{(2)}} \frac{\partial \boldsymbol{h}^{(2)}}{\partial \boldsymbol{h}^{(1)}} \frac{\partial \boldsymbol{h}^{(1)}}{\partial \boldsymbol{W}} \tag{9}$$

Updating the weights. Finally, after obtaining the derivative of the loss with respect to each of the weight matrices, the weights can be updated to reduce the loss, so in the case of weight matrix $\boldsymbol{W}$, the weights are updated as follows:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \cdot \frac{\partial l}{\partial \boldsymbol{W}} \tag{10}$$

where η is the *learning rate* with $\eta \in [0, 1]$.

If this process of updating the weights is repeated many times for all samples of a sufficiently large dataset, then the weights will slowly converge to a point where any changes in weights will yield worse results; at this point, we say that the Artificial Neural Network is done training.

3 RELATED WORK

Multiple authors argue that passenger flow is significantly impacted by holidays [42], large public events [3, 8, 22, 28, 41], or both [12, 14, 37] for different modes of travel. However, there is little research into passenger flow forecasting under such conditions. Next to that, research into passenger flow forecasting under anomaly conditions (like holidays and events) is limited by the availability of data [11].

Holidays. In 2017, Laptev et al. [12] used a Long Short-Term Memory (LSTM) model to forecast the daily amount of rides for the ride-hailing company Uber during the holidays. Their initial implementation performed worse than a baseline of Quantile Random Forests. However, significant performance improvements were achieved when using the latent space of an Auto-Encoder as inputs for the LSTM model.

Later, in 2020, Zhou and Tang [42] used an SVM to forecast the hourly inbound passenger flow of metro stations during the holidays.

Events. In 2017, Ni et al. [21] forecast the aggregate in- and outflow of a single subway station in 4-hour intervals using Twitter hashtags with a hybrid model between SARIMA and Linear Regression. Similarly, in 2022, Xue et al. [34] attempted to use hashtags and geotags in social media posts to predict disturbances in the regular passenger flow patterns related to events with a multi-stage forecasting framework. Their proposed framework yielded significantly more accurate forecasts than the selected baselines.

In 2018, Noursalehi et al. [22] forecasts deviations of arrivals per 15 minutes at stations along a single metro route in London

caused by planned events. For their 2-step real-time forecasts (30 minutes), Dynamic Factor Models (DFM), which are often used for time series forecasting in finance, are applied because such models tend to scale well to higher dimensionalities and are transparent in their forecasting. Their model appears to have performed well but was not compared to other baselines.

Then, in 2020, Chen et al. [3] constructed a hybrid model consisting of ARIMA and Nonlinear and Asymmetric Generalized Autoregressive Conditional Heteroscedasticity (NAGARCH) to forecast the arrivals and departures of two subway stations at 10-minute intervals. This hybrid model performed significantly better than the ARIMA baseline, but the authors did not compare their results to other models.

In 2021, Santanam et al. [28] investigated the additional influx of passengers in the MARTA rail network in Atlanta; they compared the performance of multiple Machine Learning approaches (Linear Regression, Random Forests, and a combination of the two) for estimating the amount of influx in ridership at the end of an event based on, among others, the event type, location and expected attendance. They showed a clear linear relationship between event attendance and ridership in public transportation, and their combined model outperformed the Linear Regression and Random Forests baselines.

Finally, in 2022, Zhao and Ma [41] tackled the forecasting problem with a whole other approach; they argue that it is "hardly possible" to train a single model that fits different scenarios. Therefore, they proposed a Naïve Bayes-based Transition model that transitions between a Gradient Boosting Decision Tree for regular passenger flow and a Deep Learning model including an LSTM for passenger flow forecasting under planned events [41]. Their model performed better than the selected baselines on forecasting in- and outbound passenger flows for a single station.

Hybrid Forecasting Approaches. Like the various works described above researching passenger flow forecasting, we instinctively know that large events can cause additional passenger flow in public transportation networks; similarly, Xue et al. [34] (2022) believes that for an observed passenger flow $X = S + T + N$, a signal composed of a seasonal component S, trend T, and noise N, the trend and noise components can be caused by additional in- and outbound passenger flow because of external factors like events.

Similar reasoning can be observed in the application of *hybrid algorithms*, an ensemble model which combines multiple forecasting algorithms with addition, in passenger flow forecasting problems. Unlike Mixture of Experts models [2], which form a gated ensemble of the same model, hybrid approaches often use one forecasting algorithm to forecast a (seasonal) baseline and another algorithm to forecast the deviations on top of the baseline [8]. Furthermore, Hoppe et al. [8] (2023) found that the usage of simple models (like ARIMA) in hybrid models with another predictor can result in high accuracy and efficiency of the forecasting method.

Some of the baseline models that have been mentioned are Historical Average (HA), K-Nearest Neighbours (K-NN), Linear Regression, and Seasonal Auto-Regressive Integrated Moving Average (SARIMA) [8]. The deviation forecasting models that have been tried are K-Means and K-Medioids [8].

From \ To	Arnhem CS	Arnhem Zuid	Oosterbeek	Velperpoort
Arnhem CS	-	500	200	600
Arnhem Zuid	300	-	400	700
Oosterbeek	500	200	-	400
Velperpoort	0	500	100	-

Table 1: Example Origin-Destination matrix on a subgraph of the Dutch train network with fictional data.

This research tests various Hybrid model architectures based on a SARIMA baseline and several components outside of this structured architecture.

4 IMPACT OF EVENT SIZE ON PASSENGER FLOW (RQ1.1)

As discussed in Section 3, Xue et al. [34] brought forward the hypothesis that external factors could explain the trend and noise components of an observed passenger flow signal. Furthermore, Santanam et al. [28] showed a clear linear relationship between attendance at sports matches and additional inbound passenger flow at three train stations.

In this research, however, the goal is to forecast passenger flows on a larger network of stations and connections; next to that, there are many more types of events other than sports, which can also impact passenger flow. Therefore, we analyze the impact of events on passenger flow for all kinds of events on the entire Bay Area Rapid Transit (BART) [26] rail network. As this is the same dataset that we will be using for forecasting later in this research, we can use the conclusions from this section to guide our decisions later.

4.1 Data

The dataset used for this analysis is the public ridership dataset from the Bay Area Rail Transit (BART) network in California [26]. The dataset consists of hourly Origin-Destination (OD) matrices $\boldsymbol{A} \in \mathbb{N}^{|\mathcal{V}|\times|\mathcal{V}|}$, where the cell $\boldsymbol{A}_{t,(o,d)}$ describes the number of passengers with origin station o and destination station d at time t. So, the time series OD matrices with a time step resolution of 1 hour denote the number of people that arrived at their destination station in one hour. Table 1 gives a fictional example of such an OD matrix.

Data regarding events was sourced from the DoTheBay website [5], which lists many events in the California Bay Area. Event information (name, date, start time, venue, location) is scraped from the website for the year 2022. After that, the expected number of visitors is matched based on the published capacity[2] of the venue.

This part of the research, the data analysis into the relationship between event size and passenger flow, uses the data from the BART network in *2022* and the 35 769 scraped events that occurred in that time frame.

[2]The published capacity is manually searched using search engines and not available for all venues.

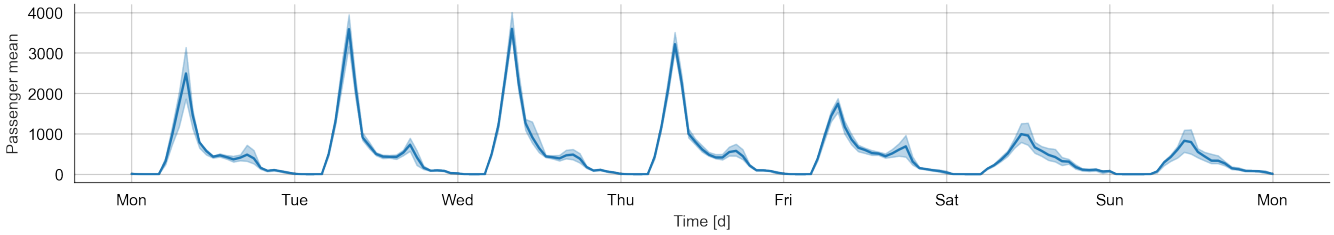


Figure 5: Mean and 80^{th} percentile range of (trend-corrected) alighting passenger flow at station *Embacadero* throughout *2022*.

Pre-processing. Before analyzing the data, it needs to be pre-processed in order to ensure it is usable. First, the BART OD matrices are tackled, which exclude any observations without any passengers. Therefore, these missing observations are re-introduced and set to 0.

Then, in order to be able to compare data points throughout the year, the overall trend needs to be removed from the data. This, however, proves to be more challenging than it sounds, as passenger flow data contains multiple seasonalities, namely a daily- and a weekly seasonality. Therefore, the data is grouped per station, day of the week, and hour of the day; then, the trend is removed per group using Season-Trend decomposition [4].

Next to passenger flow data, we also have scraped data describing events. First, all events which do not have a location are discarded so that events can be matched to the nearest station in the BART network. Then, any event which lies more than 5 km from any station is discarded as it appears to be very unlikely that attendees would walk more than 1 hour (assuming an average walking speed of 5 km/h) from the nearest station to the event.

4.2 Measuring Event Impact

Observing Figure 5, which shows the mean and the 80^{th} percentile range of alighting passenger flow per hour of the day per day of the week for the station *Embarcadero* throughout the year *2022*, we see that passenger flow tends to be highly regular. So, similarly to the approach of Santanam et al. [28], we should be able to determine the amount of extra passengers in public transportation based on the amount the observed amount of passengers deviates from what is considered to be "normal". First, the amount of alighting passengers at destination station d is needed; this can be computed from the OD matrices by combining the observations for all origin stations o:

$$p_d(t) = \sum_{o \in \mathcal{V}} A_{t,(o,d)} \tag{11}$$

As Figure 5 shows, passenger flow tends to be highly regular under "normal" conditions. Therefore, the number of extra passengers caused by events could be estimated by the difference between the observed amount of passengers $p_d(t)$ and the historic average $\mu_{d,t}$, if the observed amount of passengers $p_d(t)$ exceeds the historic 90^{th} percentile[3] $\zeta_{d,t}$. $\mu_{d,t}$ and $\zeta_{d,t}$ are computed using the trend-corrected observations at station d with the same weekday and hour of the day as t. Combining this for an event e near station d,

[3]The 90^{th} percentile threshold was determined by Pereira et al. [25] based on the authors' domain knowledge and input from local experts.

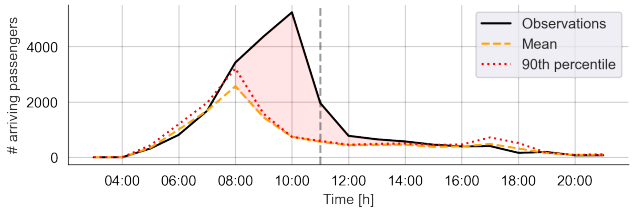


Figure 6: Alighting passenger flow at station *Embarcadero* around the *Warriors Victory Parade* on *2022-06-20* at *11:00*. The additional 10 240 arrivals at this station caused by the event are highlighted in red. *Figure representation inspired by [25].*

we get the amount of extra arriving passengers $\widehat{p}_e$:

$$\widehat{p}_e(t) = \begin{cases} 0, & \text{if } p_d(t) < \zeta_{d,t} \\ p_d(t) - \mu_{d,t} & \text{otherwise} \end{cases} \tag{12}$$

The total passenger event attendance $\widehat{P}_e(t)$ for an event at time t can then be estimated using:

$$\widehat{P}_e(t) = \sum_{i \in [t-4, t+2]} \widehat{p}_e(i) \tag{13}$$

Figure 6 shows how additional passenger flow caused by events is identified using Equation 12 for one of the most significant events in *2022*. In this figure, the highlighted area between the *Observations* and *Mean* curves corresponds to the amount of extra passengers taking the train in order to attend the event; in the case of the *Warriors Victory Parade* on *2022-06-20* an additional 10 240 passengers were observed arriving at just one of the jam-packed stations.

Matching events. Doing the same computation for every event in the dataset will yield many candidate matches between observed additional passenger flow and the event. However, there are a lot of false positives or insignificant results; therefore, candidate matches are dropped if:

- $\widehat{P}_e(t) < 50$, as train cars have at least 50 seats (and many more standing spots) [27] and trains often have multiple cars, therefore fewer than 50 passengers do not significantly impact the perception of crowdedness on an entire train;
- $\widehat{P}_e(t)$ is bigger than the event venue's capacity; or
- Another event with a bigger venue occurs near the same station within 4 hours.

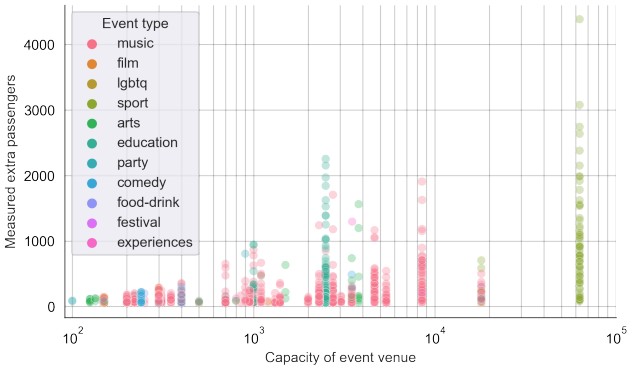


Figure 7: Lot-plot of events with their corresponding venue capacity and observed extra passengers on the train attending the event.

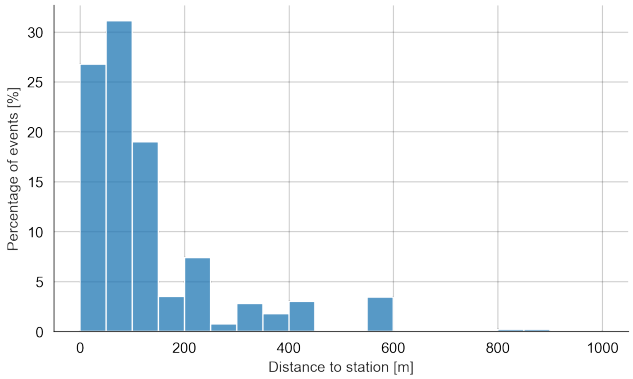


Figure 8: Distance from station to event for the matched events.

This filtering method will – of course – discard some correct matches. However, it is more important to accurately identify a relationship between events and passenger flow than to include more false negatives at the cost of noise.

The resulting number of events after filtering is 1 363.

4.3 Minimum Event Size

Figure 7 shows for each matched event the event venue capacity and the number of extra passengers on the train attending that event. There appears to be a clear upward trend in maximum observed passengers as the venue capacity increases. However, there are also observations with very few passengers for any of the venue sizes.

The fuzziness of the results combined with the density of observations with fewer extra passengers suggests that there are a lot of false positives, presumably caused by noise in the data that cannot be attributed to events. Therefore, no definitive conclusion will be drawn for research question RQ1.1; however, a threshold for event sizes still needs to be set for use later in this research. Starting around a venue capacity of 1100, the observations appear less densely positioned, and there appears to be a slight upward trend in some of the observations. Therefore, the minimum venue capacity of 1100 will be used in the rest of the research.

This tentative decision, however, is highly subjective and should be taken with a grain of salt. Future work will need to look into better indicators of event attendance than those used in this research.

Event Distance. Furthermore, Figure 8 shows how many events can be matched as the distance from the station to the event venue increases. The largest observation appears to be smaller than 900 m, but most observed events are within 500 m of the nearest station.

5 PASSENGER FLOW FORECASTING

The research aims to determine which – out of the given options – performs best at multi-step passenger flow forecasting under both "normal" and event conditions. The prediction target is the volume of passengers along all inter-station connections in a rail-based public transport network within the selected time step resolution.

5.1 Forecasting Datasets

In this research, two datasets are used from two different regions of the world:

BART The primary subject of this research's experiments is the *Bay Area Rapid Transport* (BART) network [26] in California since both ridership and event data are publicly available online. This concerns the same dataset as the one used in Section 4.

NS In cooperation with *Nederlandse Spoorwegen* (NS), a private dataset describing passenger flow in parts of the Dutch train network has been obtained. Even though the Netherlands and the United States are significantly different, we hope to observe similar results for forecasts on the NS dataset. Events related to this dataset have been manually collected by exhaustively searching for event venues in the region and scraping their past events.

The two datasets are described in Table 2.

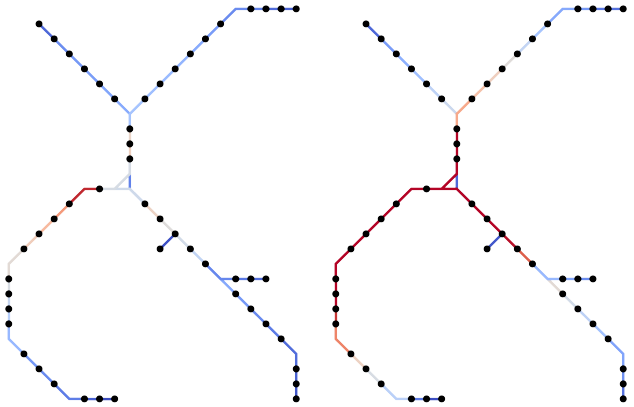
Next to the historic passenger flow observations y_t, the forecasting models also have access to exogenous features ϵ_t for the destination station per edge $e \in \mathcal{E}$, which consists of the following features:

Weekday (one-hot encoded) describing the day of the week.
Time of day (one-hot encoded) describing the time of day in blocks of 4 hours.
Event capacity describing the capacity of the event venue.
Event distance describing the distance between the event location and the station.
Event type (one-hot encoded) describing the type of the event (music, sports, et cetera).

Pre-processing. This research's forecasting target is the inter-station passenger flow $\mathcal{F}$ (or y). And while the NS dataset was delivered as the inter-station passenger flow $\mathcal{F}$ (as defined in Section 1.1), the BART dataset consists of hourly OD matrices. Therefore, these hourly OD matrices $\boldsymbol{A}_t$ are aggregated to hourly bi-directional flow graphs such that $\mathcal{F}_t(e)$ for $e \in \mathcal{E}$ is equal to the number of all passengers that traversed the given section of rail. This is obtained by summing the number of passengers along all paths that traverse

	BART	NS
# stations	50	14
# connections	102	28
max. passengers	11 288	2 363
mean passengers	354	306
# total observations	1 336 608	123 648
# events	6 764	51
resolution	1 hour	30 minutes
train set	2022-01-01 ⟶ 2022-12-31 (66.8%)	2022-07-01 ⟶ 2022-08-15 (50.0%)
validation set	2023-01-01 ⟶ 2023-03-31 (16.5%)	2022-08-16 ⟶ 2022-08-31 (17.4%)
test set	2023-04-01 ⟶ 2022-06-30 (16.7%)	2022-09-01 ⟶ 2022-09-31 (33.7%)

Table 2: Overview of datasets used in this research



(a) 07:00 - 08:00 **(b) 10:00 - 11:00**

Figure 9: Example observed passenger flow in the BART network on *2022-06-20*, where blue indicates very few passengers and red a lot of passengers.

edge e:

$$\mathcal{F}_t(e) = \text{sum}\left(\left\{\boldsymbol{A}_{t,(o,d)} \textbf{ for } o,d \in \mathcal{V}\times\mathcal{V} \textbf{ if } e \in \mathcal{P}(o,d)\right\}\right) \quad (14)$$

where $\mathcal{P} : \mathcal{V}\times\mathcal{V} \to \mathcal{E}^*$ is a function that gives the shortest path from origin station o to destination station d as a sorted list of edges. An example of observed passenger flow throughout the entire BART network is given in Figure 9; the visualization of passenger flow throughout the entire day is given in Appendix A.

Furthermore, events are matched based on location and start time to the directed flow of an edge if:

- the edge's destination is the same station as the event's nearest station;
- the event's venue has a capacity of $> 1\,100$ attendees (as determined in Section 4); and
- the distance between the event's venue and the nearest station is $< 500\,m$.

On the NS dataset, however, we also include events up to $2\,500\,m$ distance from the nearest train station, since the same experiments as those in Section 4 could not be done for the NS dataset due to the limited amount of scraped events for this dataset. Moreover, based on domain knowledge of the author, it is not uncommon to walk up to 30 minutes from public transportation to an event location in the Netherlands for larger events and festivals.

5.2 Forecasting Algorithms

As mentioned in Section 3, there has been much research towards Passenger Flow Forecasting, and with that, many different algorithms have been tried. However, the research under event conditions, especially with inter-station occupancy as a forecasting target, remains limited.

Therefore, this research compares a few forecasting algorithms to determine which one(s) can be used for passenger flow forecasting under normal and event conditions in real-world scenarios. Based on these results, we implement a Graph Neural Network (GNN) algorithm (as will be discussed later in this section). The model architectures listed below have been keeping a few things in mind:

- Following the thoughts set forward by Manibardo et al. [17], it might not always be necessary to use overly complex Deep Learning approaches with millions of trainable parameters for Passenger Flow Forecasting problems.
- The models must have the capability to take exogenous features (like holidays and events) as input.
- The models must be able to handle public transportation networks of different sizes without re-training the entire model.

Baseline - SARIMAX. The baseline model is SARIMAX [31], consisting of some extensions on top of a commonly used and traditional time series forecasting model across many domains, namely the *AutoRegressive Integrated Moving Average* (ARIMA) model [10]. The SARIMAX model (as shown in Figure 10) is essentially a Linear Regression model that is very effective for its subjectively low complexity compared to more complex Deep Learning approaches. This model will serve as a baseline for comparing differences in performance against the added complexity of the other selected models.

After selecting the proper order $(p,d,q)(P,D,Q)_m$ of the SARIMAX model as described in Appendix B, the future passenger flow,

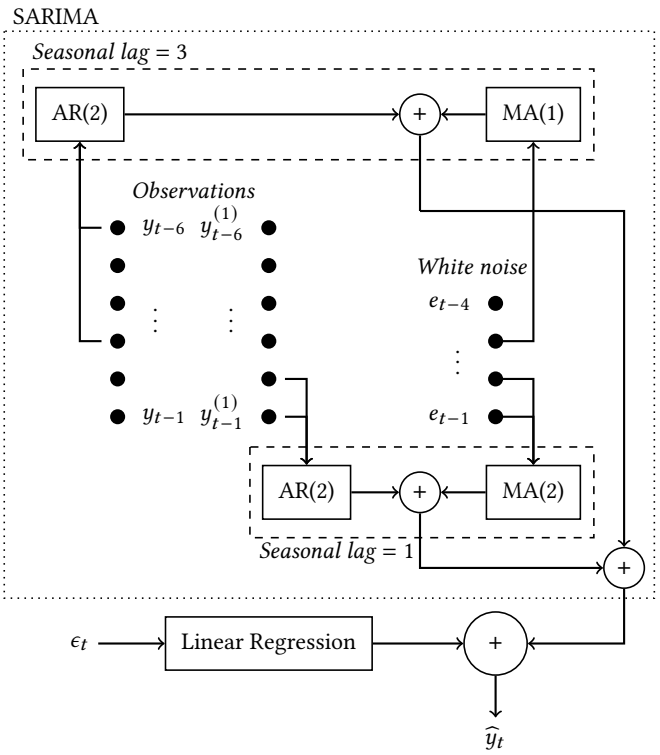


Figure 10: SARIMAX model architecture for example configuration $SARIMAX(2,1,2)(2,0,1)_3$.

$\widehat{y}_t$, can be forecast using the following formula:

$$\widehat{y}_t = \underbrace{\sum_{k=1}^{p} u_k \cdot y_{t-k}^{(d)} + \sum_{k=1}^{q} v_k \cdot e_{t-k} + e_t + c}_{\text{ARIMA}} + \underbrace{\sum_{k=1}^{P} w_k \cdot y_{t-k\cdot m}^{(D)} + \sum_{k=1}^{Q} z_k \cdot e_{t-k\cdot m}}_{\text{Seasonal component}} + \underbrace{\sum_{k=1}^{|\epsilon_t|} s_k \cdot \epsilon_{t,k}}_{\text{Exogenous parameters}} \tag{15}$$

where:

- $y^{(i)}$ is the i^{th} order difference of observed passenger flow y;
- e is i.i.d. white noise and is often normally distributed;
- ϵ_t is a vector with observed exogenous variables at time t;
- c is a bias / constant; and
- u, v, w, z, and s are vectors with learnable weights.

Hybrid Forecasting. As was discussed in Section 4, passenger flow tends to be highly regular, but large-scale events can correlate with additional passenger flow on top of the "normal" patterns. Similar theories were also discussed in Section 3 like Hoppe et al.'s Hybrid Models [8], which consist of a simple baseline such as SARIMA to predict the basic patterns and another model to predict additional peaks (deviations) caused by exogenous influences (like events).

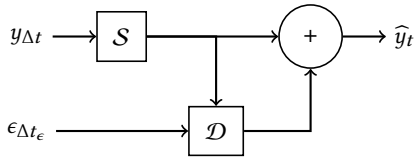


Figure 11: Hybrid model architecture.

Let $\mathcal{S}$ be a tuned SARIMA model (see Equation 15, without the last summation), then the future passenger flow $\widehat{y}_t$ can be forecasted using the following formula:

$$\widehat{y}_t = \mathcal{S}(y_{\Delta t}) + \mathcal{D}(\mathcal{S}(y_{\Delta t}), \epsilon_{\Delta t_\epsilon}) \tag{16}$$

where:

$\mathcal{D}$ is the Deviation model;
Δt is the set of indices of historical observations for the SARIMA model, such that $\forall_{t \in \Delta t} t < 0$;
Δt_ϵ is the set of indices of observations of exogenous features that can be negative as well as positive.

The Deviation model $\mathcal{D}$ can be any regression model; whereas Hoppe et al. [8] used K-Means or K-Medioids, we implement a Linear Regression (LR) model and Multi-Layer Perceptron (MLP) model with two hidden layers as Deviation models.

In contrast to the SARIMAX model, our implementation of the Hybrid model does not just consider the exogenous features at time t, but, as seen in the equation above (Equation 16), the Deviation model also considers exogenous features in the past and future as events also have an impact on passenger flow for some time before their start and after their end.

The following equations define the MLP and LR deviation models:

$$LR(z, \epsilon) = \boldsymbol{T} \cdot \begin{bmatrix} z \\ \epsilon \end{bmatrix} \tag{17}$$

$$MLP(z, \epsilon) = \boldsymbol{U} \cdot \sigma\left(\boldsymbol{V} \cdot \sigma\left(\boldsymbol{W} \cdot \begin{bmatrix} z \\ \epsilon \end{bmatrix}\right)\right) \tag{18}$$

where:

z is the output of the SARIMA model;
ϵ is the set of exogenous features for multiple time steps;
σ is the ReLU activation function (see Eq. 5); and
$\boldsymbol{T}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$ are matrices consisting of learnable weights.

The architecture of the hybrid models is visualized in Figure 11.

Furthermore, the selected Deviation architectures (MLP and LR) are also tested in a standalone configuration in order to test the potential benefit of the Hybrid model architecture over its simpler counterparts.

Graph Neural Networks. The aforementioned models only operate on a single edge $e \in \mathcal{E}$ at a time instead of simultaneously considering the entire network of edges $\mathcal{E}$. Incorporating information about the entire network in forecasting in a structured way might positively impact short-term forecasting performance, as

	SARIMA	SARIMAX	Hybrid LR	Hybrid MLP	LR	MLP	GNN
# trainable parameters	31	52	138	1 888	136	2 321	3 973
# historic observations (Δt)	168		168		168		168
# exogenous features	-	23	23		23		23
# exogenous observations (Δt_ϵ)	-	$\{0\}$	$\{-2, \ldots, 4\}$		$\{-2, \ldots, 4\}$		$\{-2, \ldots, 4\}$
Batch size	256		256				32
Epochs	10		25				50
Initial learning rate	$1 \cdot 10^{-3}$		$1 \cdot 10^{-3}$				$1 \cdot 10^{-3}$
Criterion	Mean Squared Error (Eq. 20)						
Optimizer	AdamW [16]						
Learning rate decay	Exponential (0.9^i)						

Table 3: Hyper-parameters

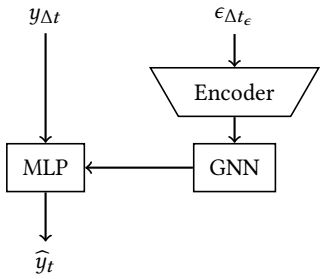


Figure 12: GNN model architecture.

extra crowdedness in one region can often impact crowdedness in other regions.

Graph Neural Networks (GNN) offer a way to model the complex spatial dependencies in public transportation networks [11, 37]. Temporal dependencies in *Spatio-Temporal Graph Neural Networks* (STGNN) are often modelled using either time-convolutions [11, 37] or recurrent neural networks [11, 37].

In this research, we implement a *Message-Passing Graph Neural Network* (MPNN) [11] to model the spatial dependencies of just the event features. Peeking forward to the results in Section 6, we observe that the plain MLP model performs best; so for this architecture, we use the MLP architecture to model the edge-level temporal complexities. Furthermore, to limit the computational complexity of the GNN component, the exogenous features ϵ are encoded using an MLP with one hidden layer.

To generally explain the functionality of an MPNN, we consider an edge $e \in \mathcal{E}$ with edge-level features $\widehat{\epsilon}_e$, then the output $f(e)$ is computed as follows:

$$f(e) = \phi\left(\bigoplus_{n \in \mathcal{N}(e)} \psi\left(\widehat{\epsilon}_n\right)\right) \tag{19}$$

where:

$\mathcal{N}(e) : \mathcal{E} \to \mathcal{E}^*$ yields a set of neighbouring edges incident to e;

$\bigoplus$ is a *permutation-invariant*[4] aggregation operator, which is the sum operator $\sum$ in the case of this implementation; and

ϕ, ψ are learnable functions (such as MLPs).

The architecture of this GNN-based model is visualized in Figure 12.

5.3 Implementation Details

All models are implemented and trained using the popular Python library PyTorch [24]; Table 3 shows the hyper-parameters used for each of the trained models. However, since the time step granularity of the NS dataset is half of the BART dataset, more past observations and exogenous observations are included to match the same amount of time these observations would have been in the BART dataset. Furthermore, after observing the training of multiple forecasting algorithms, the amount of epochs during training for the NS dataset has been set to 25 epochs based on how long it took each algorithm to converge during training.

In order to stabilize the performance of the model, observations y of $\mathcal{F}$ are normalized per edge $e \in \mathcal{E}$ by scaling these observations to a $[0, 1]$ range based on the maximum observation y in the window of past observations Δt.

All models, even the Linear Regression-esque models, are trained using *Mini-Batch Gradient Descent* for two reasons:

(1) It allows us to train the same model multiple times (with multiple random seeds) to end up with slightly different weights (stochasticity), giving a distribution of forecasts allowing us to interpret how confident certain forecasts are.
(2) It allows us to train more complex models based on SARIMA in a single framework so that gradients for the entire model can be computed and applied during training.

The selected implementation of Gradient Descent is *AdamW* [16] because of its popularity. The same reasoning goes for the selection of the ReLU activation function in the forecasting algorithms with multiple layers of neurons.

The loss function that was selected for this research is the *Mean Squared Error* (MSE), which is essentially the squared variant of the RMSE (see Eq. 3). This loss metric was chosen as the goal of this

[4]We call a function f *permutation-invariant* if the order of items (or in this case, edges) has no impact on the outcome of the function, so for all possible permutations P: $f(X) = f(PX)$

research is to accurately predict the additional passengers in public transportation caused by large events; therefore, an error metric which "punishes" larger errors more is needed. The equation of the MSE is given below:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 \tag{20}$$

Finally, the actual implementations of the models can be found on this research's GitHub repository:
`jeffreybakker/passenger-flow-forecasting`[5]

5.4 Experiments

This subsection describes the experiments that will be performed in order to answer the research questions defined in Section 1.2. All experiments are performed on both datasets separately.

Forecast Performance Metrics. The performance of the different forecasting approaches is measured using the different error metrics highlighted in Section 2, namely the *MAE* (Eq. 1), *MAPE* (Eq. 2), and *RMSE* (Eq. 3).

Furthermore, another popular performance metric in regression problems is the *Coefficient of Determination* R^2, which is an indicator of how well the forecasts $\widehat{y}$ represent the observations y:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y_i})^2}{\sum_{i=1}^{n}(y_i - \mu_y)^2} \tag{21}$$

where μ_y is the mean observation of y.

Since every model is trained multiple times, the values of each of these metrics (MAE, MAPE, RMSE, and R^2) are averaged over all sets of trained weights per model. The results displayed in Section 6 show the mean of these metrics, or in the case of figures, the mean forecast value per forecasting algorithm.

Forecasting Performance. To get an overview of the forecasting performance of the different forecasting approaches, the aforementioned metrics are compared for all models under different conditions:

Overall performance The overall performance is sampled over the entirety of the test set and across the entire network, giving an overview of the overall performance of the different approaches on the passenger flow forecasting problem.

Event conditions The performance under event conditions is sampled for all events in the test set between 4 hours before the event's start and 2 hours after the event's start on all edges within 2 steps of the event.

Impact of Event Features on Forecasting Performance (RQ1.2). In order to determine the impact that including event-related features (as mentioned in Section 5.1) in the forecasting approaches has, all forecasts are done twice, once including all features and once with the event-related features masked out.

[5]https://github.com/jeffreybakker/passenger-flow-forecasting

Multi-step Forecasting Performance. Public transportation companies need forecasts for multiple different steps into the future; therefore, the forecast performance for multiple n-step forecasts is compared for the different algorithms using the Mean Absolute Error (MAE, see Eq. 1):

1 hour in order to be able to provide *Real Time Crowding Information* to passengers [8]. This allows them to alter their travel plans and select a less busy route if they wish to do so [29].

6 hours in order to make some last-minute adjustments to the schedule; as the NS does up to 4 hours before departure [20].

72 hours in order to schedule materials and staff.

Residual Diagnostics. Next to the various metrics that have been described above, some tests can be performed on the *residuals* of the various forecasting algorithms to verify the quality of the trained forecasting algorithms. The *residuals* are defined as the difference between the estimated value $\widehat{y}$ and the actual observation y:

$$residual = \widehat{y} - y \tag{22}$$

First, a good estimator should be unbiased [10]; therefore, the average *bias* of all residuals must equal 0. The formula for computing the bias is given below:

$$bias = \frac{1}{n}\sum_{i=1}^{n}\widehat{y} - y \tag{23}$$

Secondly, the residuals should be normally distributed [10]; the plotted residual densities should resemble the bell curve shape associated with a normal distribution.

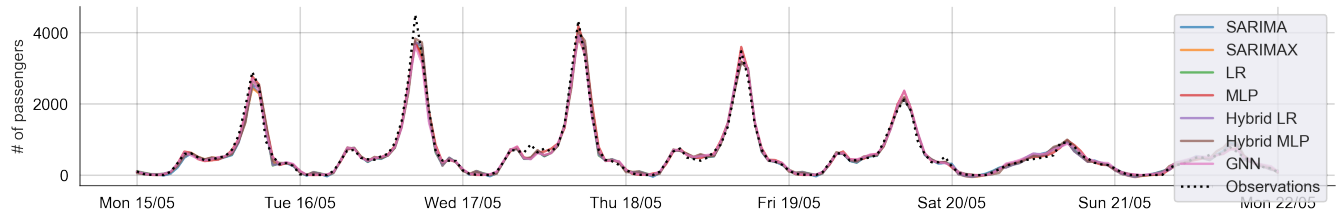
Then, the residuals should be uncorrelated [10]; if there are any significant correlations in the residuals, there are still patterns in the data that could be learned.

Transfer. Finally, this research has two datasets from two different geographical and societal locations. It would be interesting to see how algorithms trained on one dataset (BART) are able to perform out-of-the-box on the other dataset (NS). However, as these two datasets have different time step resolutions, forecasting on the NS dataset using the trained weights from the BART dataset is done by discarding every other observation in the NS dataset such that the expectations of daily and weekly seasonality in the data from the trained forecasting algorithms will still hold.

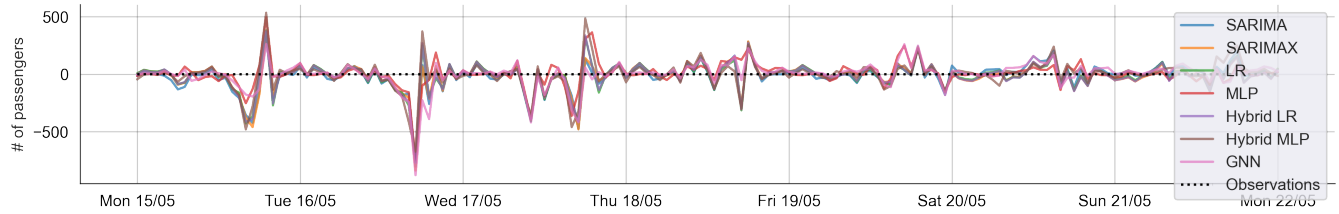
6 RESULTS

As mentioned in the previous section (Section 5.4), each forecasting algorithm is trained multiple times to end up with different sets of trained weights. This section only displays the averaged metrics or results from the various forecasts with different sets of weights; Appendix E also displays the standard deviation (σ) between the different sets of weights for each forecasting algorithm. Overall, over various runs of the same forecasting algorithms, most of the forecasting algorithms appear to have converged to the same (local) optimum.

This is confirmed by the figures in Appendix D, which show how the MAE and RMSE on the training and validation datasets evolved as the forecasting algorithms continued training. Most forecasting



(a) Forecasting results.



(b) Forecasting errors.

Figure 13: 1-step forecasting results and errors from station *12th St* (12) to station *19th St* (19).

algorithms converged to very similar metrics on the training and validation datasets with every set of weights; the exception to this statement is the GNN model, which converged to slightly different (local) optima.

Furthermore, the run times of training the various forecasting algorithms are summarized in Appendix C.

Then, before delving into the performance metrics, a sample of 1-step forecasts of the different forecasting algorithms in this research are compared against the observed passenger flow in Figure 13; all of the models appear to follow the line of observations relatively well, however, there are some forecast errors. These errors are highlighted in Figure 13b, where most errors appear to resemble random noise, except for some larger peaks, which coincide with the peaks in the observed passenger flow (as seen in Figure 13a).

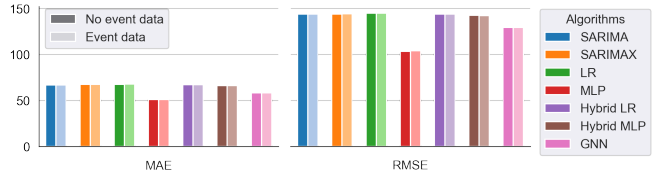
Overall forecasting performance. The overall performance of the different models over the entire test set and across the entire BART network is given in Figure 14a and Table 4; both of these show how the different models perform overall when having access to data regarding events, versus having event-related features masked out. As can be seen in both Table 4 and Figure 14a, there are no significant differences in forecasting performance between the forecasts with event data compared to the forecasts without event data. More specifically, most models' forecasts with event data have slightly worse performances across most metrics.

Next to that, the SARIMA, SARIMAX, and Hybrid algorithms all have similar performance, but the MLP and GNN algorithms perform measurably better across all metrics. The cause of this is presumably that the MLP and GNN models can better model the distribution of previous observations with multiple layers of neurons. In contrast, these past observations are bottle-necked by a single neuron in the SARIMA, SARIMAX, and Hybrid algorithm architectures.

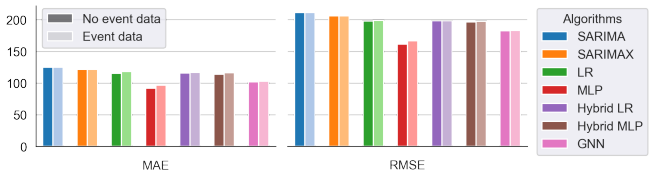
Forecasting performance under event conditions. Then, looking at a different subset of the testing data, Figure 14b and Table 5 summarize the performance of the various forecasting algorithms under event conditions. And again, both of these show the difference in performance when forecasting with event data versus forecasting without event data. Once more, all forecasting approaches appear to perform slightly worse when forecasting with event data than without event data, except that the differences in performance under

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	67.02	101.76	143.83	0.9616
	SARIMAX	67.36	110.49	144.03	0.9615
	LR	67.59	110.64	144.72	0.9611
	MLP	50.94	49.90	103.55	0.9801
	Hybrid LR	67.29	111.36	143.95	0.9615
	Hybrid MLP	66.08	78.94	142.46	0.9623
	GNN	58.44	66.26	129.40	0.9689
With event data	SARIMA	-	-	-	-
	SARIMAX	67.37	110.50	144.04	0.9615
	LR	67.72	110.67	144.80	0.9611
	MLP	51.14	49.94	103.90	0.9799
	Hybrid LR	67.35	111.38	143.96	0.9615
	Hybrid MLP	66.10	78.97	142.37	0.9624
	GNN	58.42	66.27	129.28	0.9689

Table 4: Difference in overall 1-step forecasting performance in the BART dataset.



(a) 1-step forecasting performance.



(b) 1-step forecasting performance under event conditions.

Figure 14: 1-step forecasting performance for the entire test set or specifically under event conditions in the BART dataset.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	125.07	14.93	211.19	0.9588
	SARIMAX	121.42	14.66	205.76	0.9608
	LR	115.51	15.73	197.66	0.9635
	MLP	91.78	13.41	161.11	0.9765
	Hybrid LR	115.57	15.05	198.08	0.9619
	Hybrid MLP	114.07	15.54	196.46	0.9621
	GNN	102.05	14.44	182.42	0.9714
With event data	SARIMA	-	-	-	-
	SARIMAX	121.51	14.68	205.95	0.9607
	LR	118.24	16.37	198.96	0.9630
	MLP	96.48	14.26	166.55	0.9749
	Hybrid LR	116.83	15.35	198.36	0.9618
	Hybrid MLP	116.12	16.20	197.14	0.9619
	GNN	102.66	14.57	182.77	0.9713

Table 5: 1-step forecasting performance under event conditions in the BART dataset.

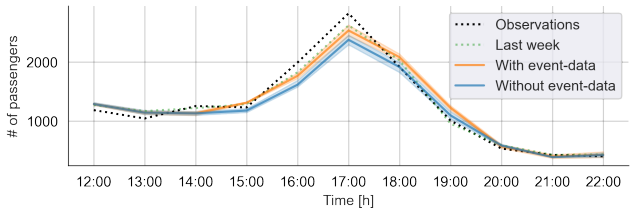


Figure 15: Impact of event features on 6-step rolling forecasting results using the MLP model under event conditions in the BART dataset on the connection from *Embarcadero* to *Montgomery St.* on *2023-05-02*.

	1 hour	6 hours	72 hours
SARIMA	67.02	75.28	80.07
SARIMAX	67.37	76.10	81.26
LR	67.72	76.55	82.12
MLP	51.14	63.28	67.41
Hybrid LR	67.35	75.58	80.09
Hybrid MLP	66.10	73.37	77.48
GNN	58.42	66.79	69.59

Table 6: Comparison of multi-step rolling forecasting performance in the BART dataset measured with MAE over the entire network.

event conditions are slightly more pronounced, especially for the MAE metric.

Regardless of the difference in performance with or without event data, the difference between the performance of the MLP and GNN algorithms compared to the rest of the algorithms is even more significant under event conditions. Furthermore, whereas the MLP model performs measurably worse when forecasting with event data than without event data, the GNN algorithm's performance is more stable.

Something that does give a slightly optimistic outlook on the impact of forecasting using event-related features is Figure 15; this is a cherry-picked example of how using event-related features can have a positive impact in passenger flow forecasting. In this figure, we can observe how a part of the peak in passenger flow, which is – presumably – caused by the event, is accounted for by the MLP algorithm's forecast when provided with information regarding upcoming events.

Multi-step rolling forecasting performance. On another note, Figure 16 and Table 6 indicate how the models perform as they have to make predictions for further in the future. Over the first hours into the future, some significant drops in performance can be observed across all forecasting algorithms. The plain MLP model still seems to outperform all of the other models. However, there are two more pressing observations to be made:

- There is a significant drop around the 24 hour mark, and another smaller one around the 48 hour mark suggesting a loss in the quality of indicators between days; and
- Models with similar architectures (MLP-based, LR-based with exogenous features, or SARIMA) appear to converge to similar errors as our forecasts continue.

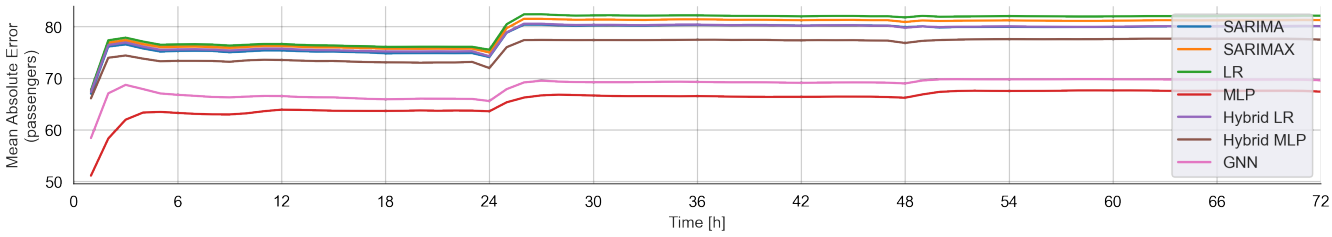


Figure 16: Multi-step rolling forecasting error for the different forecasting approaches in the BART dataset, measured with MAE.

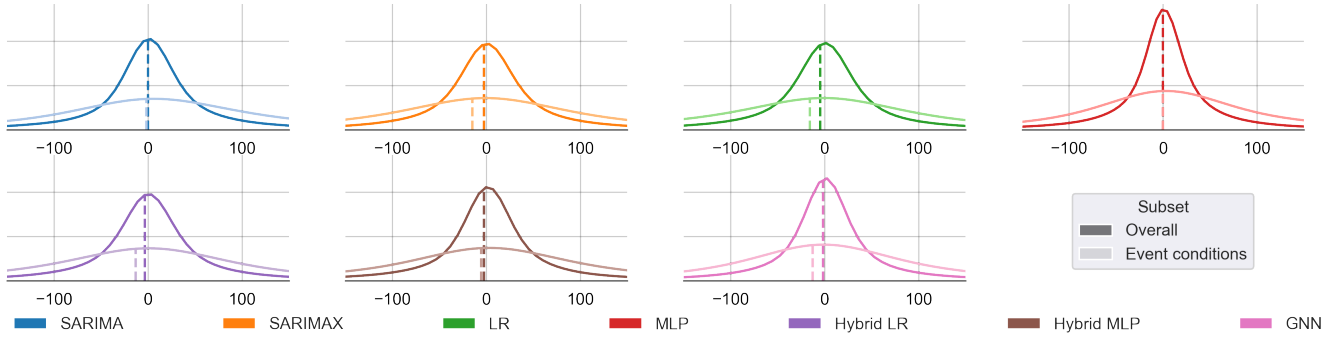


Figure 17: Overall distribution density of forecasting residuals and under event conditions in the BART dataset. The mean residual (or bias) is given by the vertical dashed lines.

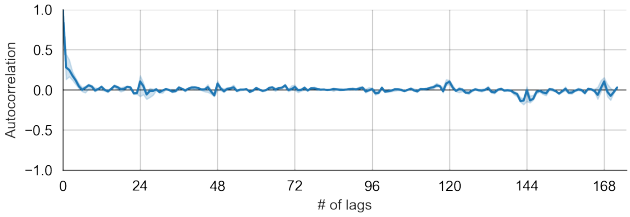


Figure 18: Averaged autocorrelation of forecasting residuals of all forecasting algorithms on the connection between *Embarcadero* and *West Oakland* in the BART dataset.

Residual diagnostics. Figure 17 shows the distribution density of forecasting residuals over the entire test set and under event conditions. All distributions in this figure resemble the bell curve of the normal distribution, although some of these bell curves are a bit more stretched out horizontally. Furthermore, the overall bias of most algorithms is very close to 0, but the bias of these same algorithms tends to be slightly more negative under event conditions. Furthermore, 18 shows no significant correlations in the forecasting residuals.

6.1 NS Dataset

This subsection covers the results that follow from the NS dataset; in some cases, the data might be represented slightly differently compared to the results from the BART dataset due to the sensitive nature of the dataset. This also means that this thesis does not mention the specific stations or dates of certain sensitive observations. However, this should not diminish the value of the insights gained from these results.

The first thing to do is to sketch the context of these results; as was seen in Figure 5, passenger flow in the BART dataset appeared to be highly regular with steep peaks every day, which are – presumably – caused by daily commutes. Trends such as these are not as apparent in this example from the NS dataset between neighbouring stations Q and R in Figure 19. In contrast to the sample from the BART dataset, there are some observable peaks at the beginning and end of the day on Monday through Thursday, but the passenger flow appears to be more steadily dispersed throughout the day. Furthermore, there appears to be more noise between observations in this figure because the time step regularity here is 30 minutes instead of 1 hour, and there is probably a train that is not scheduled twice per hour.

Then, just as before with the BART dataset (in Figure 13), we observe a generic forecast example in Figure 20; it immediately becomes evident that the additional noise that is present in the passenger flow data has a negative impact on the forecast performance as the forecasting errors are relatively quite large. Overall, the errors tend to resemble a noise pattern, except for some significant errors at the morning commutes.

Overall forecasting performance in the NS dataset. This negative impact on the forecasting performance can also be observed in Table 7, which shows the performance of the different forecasting

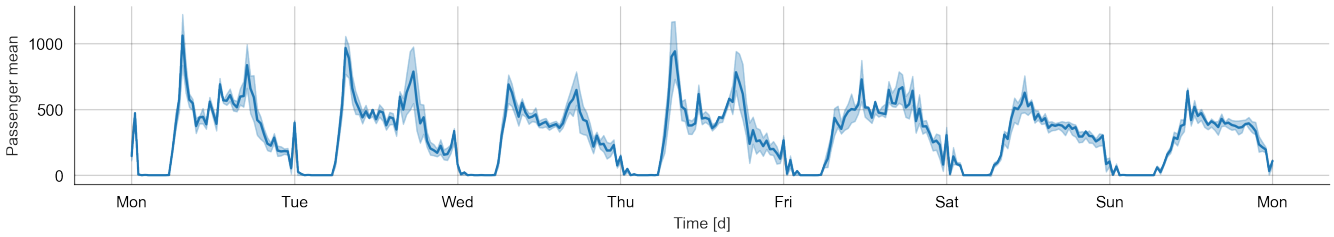


Figure 19: Mean and 80^{th} percentile range of (trend-corrected) passenger flow between stations R and Q in the NS dataset.

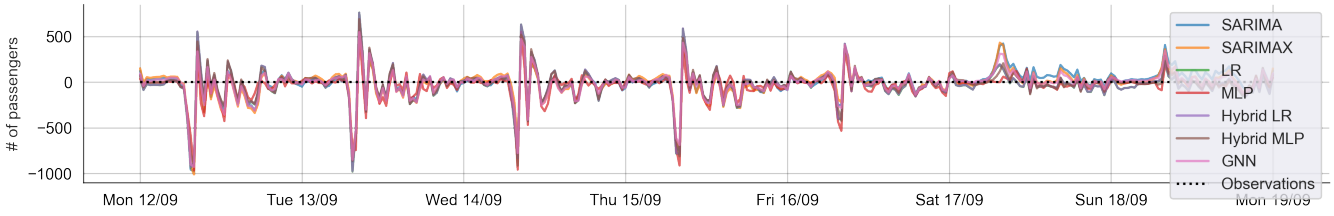
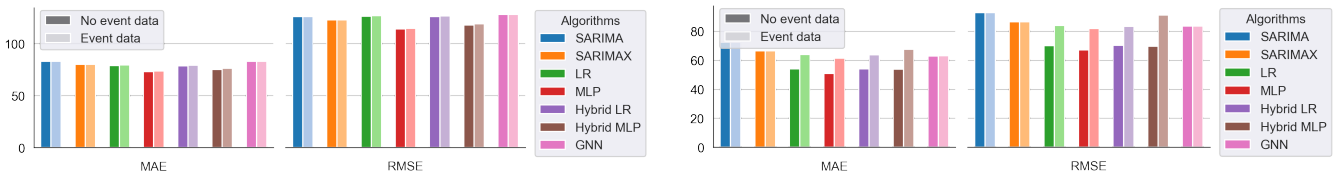


Figure 20: 1-step (30 minutes) forecasting errors from station R to station Q in the NS dataset.



(a) 1-step forecasting performance.

(b) 1-step forecasting performance under event conditions.

Figure 21: 1-step (30 minutes) forecasting performance for the entire test set or specifically under event conditions in the NS dataset.

algorithms over the entire test set and across all stations in the NS dataset; even though the overall passenger flow in the NS dataset is smaller than the passenger flow in the BART dataset, the MAE is significantly higher and the R^2 score is significantly worse for the NS dataset. Next to that, the RMSE and MAE lie relatively closer together, which – combined with the higher MAE – means that forecasting algorithms have worse overall performance compared to the performance on theBART dataset, but there are relatively fewer large errors in the forecasts. However, this can also be caused by fewer events or other anomalous conditions.

Furthermore, both MAE and RMSE are visualized in Figure 21a, where, in contrast to the same figure for the BART dataset (Figure 14a), the differences in forecasts with and without event data are slightly more pronounced.

Forecasting performance under event conditions in the NS dataset. In Figure 21b and Table 8, we observe similar patterns in the forecasting performance under event conditions in the BART dataset (see Table 5); namely that most algorithms perform significantly worse under event conditions when having access to features describing those events as opposed to those features being masked out. This indicates that the very few events found for the NS dataset are insufficient for forecasting algorithms to "learn" anything about the impact of events on passenger flow.

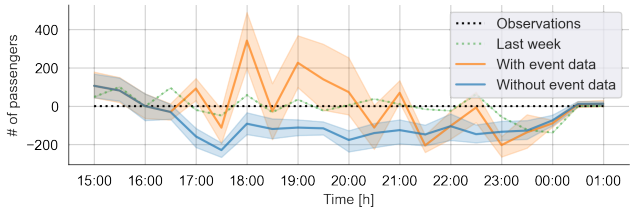


Figure 22: Impact of event features on 6 hour (12 steps) rolling forecasting errors under event conditions near station S in the NS dataset.

Then, considering the cherry-picked example in Figure 22, there appears to be an attempt to forecast the additional passenger flow caused by the event, but the forecasting algorithm overshoots the target; it is evident that both forecasts appear to be lacking in matching all of the other observations in this figure.

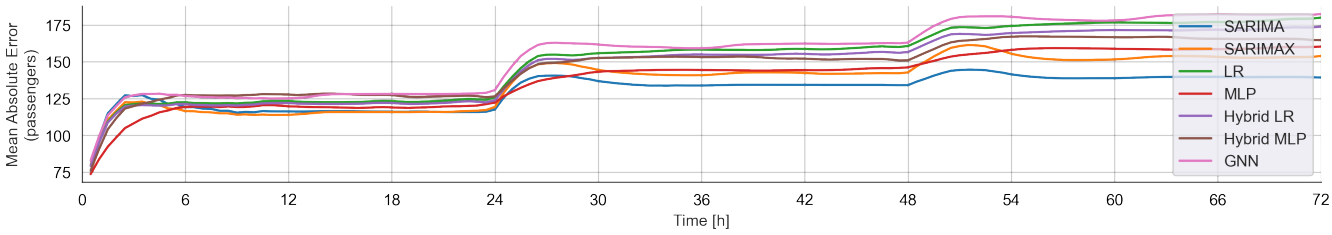


Figure 23: Multi-step rolling forecasting error for different forecasting approaches on the NS dataset, measured with MAE.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	82.79	28.51	125.88	0.7897
	SARIMAX	80.00	26.90	122.62	0.8004
	LR	78.85	24.89	126.37	0.7880
	MLP	72.95	23.43	114.11	0.8271
	Hybrid LR	78.62	24.92	126.06	0.7890
	Hybrid MLP	75.15	24.45	117.91	0.8154
	GNN	82.87	27.57	127.92	0.7804
With event data	SARIMA	-	-	-	-
	SARIMAX	80.00	26.91	122.62	0.8004
	LR	79.49	25.21	126.92	0.7861
	MLP	73.62	23.72	114.72	0.8252
	Hybrid LR	79.27	25.23	126.60	0.7872
	Hybrid MLP	76.09	24.90	118.90	0.8123
	GNN	82.87	27.57	127.93	0.7804

Table 7: Difference in overall 1-step (30 minutes) forecasting performance on the NS dataset.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	72.33	28.15	92.84	0.7703
	SARIMAX	66.53	25.46	86.48	0.8006
	LR	54.04	20.53	70.12	0.8723
	MLP	50.91	18.99	67.17	0.8873
	Hybrid LR	54.12	20.88	70.19	0.8683
	Hybrid MLP	53.75	21.51	69.60	0.8705
	GNN	62.99	23.31	83.49	0.8167
With event data	SARIMA	-	-	-	-
	SARIMAX	66.55	25.60	86.47	0.8006
	LR	63.87	25.30	83.92	0.8171
	MLP	61.40	23.49	81.85	0.8326
	Hybrid LR	63.66	25.57	83.48	0.8137
	Hybrid MLP	67.64	28.12	91.17	0.7776
	GNN	63.03	23.31	83.54	0.8165

Table 8: 1-step (30 minutes) forecasting performance under event conditions around station Q in the NS dataset.

Multi-step rolling forecasting performance in the NS dataset. Table 9 and Figure 23 show the performance of the various forecasting algorithms when performing a rolling forecast for multiple steps into the future. The figure shows totally different results from the BART dataset; here, SARIMA performs best over time, and all other forecasting algorithms have varying rates of errors. This suggests that there is too much noise and too little data for the forecasting algorithms to learn the distributions of passenger flow in this dataset properly.

Residual diagnostics in the NS datataset. Figure 24 shows the distribution density of forecasting residuals over the entire NS test set and under event conditions. All distributions in this figure resemble the bell curve of the normal distribution, but some of them appear to have a slight skewness to either the positive or negative side. Furthermore, the overall bias of most algorithms is obviously not close to 0 (or at least not as close as was observed

	30 minutes	1 hour	6 hours	72 hours
SARIMA	82.79	99.25	119.13	139.30
SARIMAX	80.00	95.37	116.46	153.91
LR	79.49	95.77	122.46	179.99
MLP	73.62	83.90	119.15	160.33
Hybrid LR	79.27	95.45	121.24	173.86
Hybrid MLP	76.09	90.84	127.40	164.70
GNN	82.87	99.67	126.63	182.35

Table 9: Comparison of multi-step rolling forecasting performance measured with MAE over the entire network in the NS dataset.

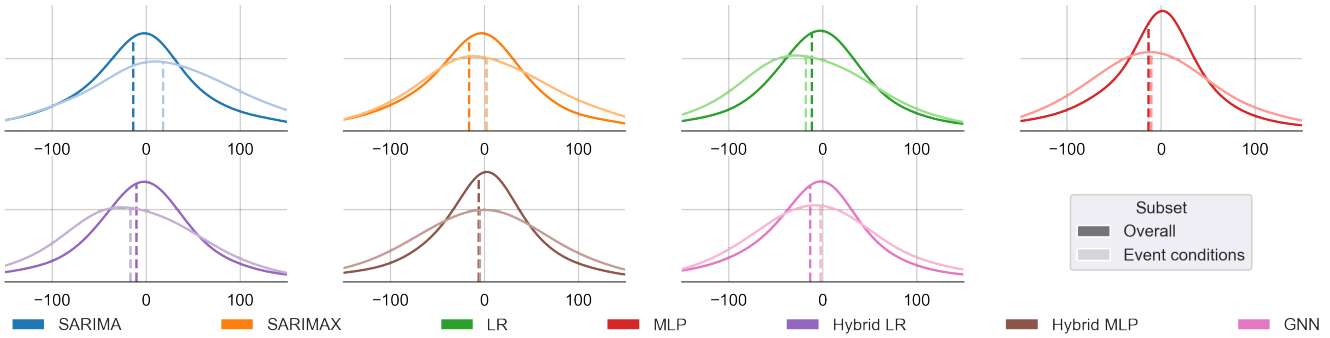


Figure 24: Overall distribution density of forecasting residuals and under event conditions in the NS dataset. The mean residual (or bias) is given by the vertical dashed lines.

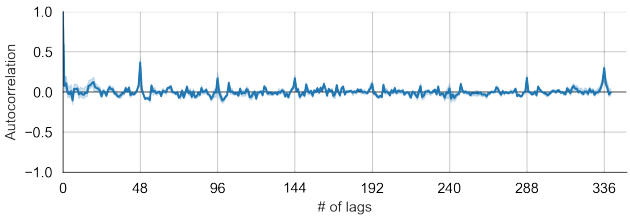


Figure 25: Averaged autocorrelation of forecasting residuals of all forecasting algorithms on the connection between Q and R in the NS dataset.

for the BART dataset in Figure 17). This, again, suggests that the forecasting algorithms have not seen enough examples to properly learn the relationships in the data of the NS dataset.

This belief is once more confirmed by looking at Figure 25, which shows some note-worthy correlations in residuals at 48 lags (1 day), 288 lags (6 days) and 336 lags (7 days). Therefore, there are noticeable patterns in the residuals that have not yet been learned by the forecasting algorithms.

6.2 Transferred Weights

This subsection summarizes how the weights trained on the BART dataset perform out-of-the-box on the NS dataset, which is perfectly done in Figure 26. This figure shows how the transferred weights all perform worse than the weights trained on the training set, which corresponds to the test set. However, the out-of-the-box performance of the transferred weights lies within the same ballpark as the weights trained on the NS training set.

Amazingly, the out-of-the-box overall forecasting performance of the transferred weights for the MLP forecasting algorithm performs even better than some of the other forecasting algorithms trained on the NS training set. However, the same cannot be said for such performance under event conditions, where all forecasting algorithms with transferred weights performed significantly worse than their counterparts, which have been trained on the NS training set.

More detailed performance metrics of the transferred weights can be found in the tables of Appendix E.

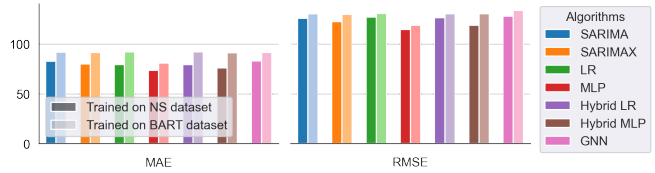
7 DISCUSSION

In this section, we discuss the limitations of this research and hint at some potential starting points for future works; the actual starting points for future works are discussed in further detail in Section 9.

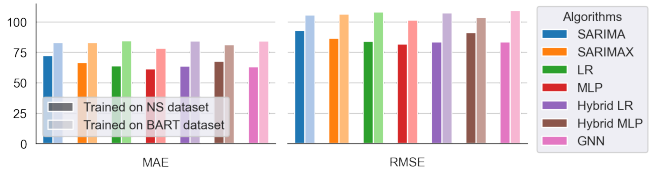
Event attendance. In this research, we have operated under the assumption that the events' venue capacities, in combination with the type of the event, were adequate indicators of the eventual attendance of the event; however, the opposite has proved to be true. Santanam et al. [28], for example, showed a direct linear relationship between event attendance and the amount of extra passengers using public transportation at a few selected stations, the relationship between the events' venue capacities and the extra passengers shown in Section 4 is a lot more fuzzy and does not show such a clear linear relationship.

Data quantity. As is visible in Table 2 in Section 5, events are under-represented in both datasets, but even more so in the NS dataset than the BART dataset. That events are under-represented in the NS dataset is evident from the fact that the difference in performance between forecasting with and without event data under event conditions is significantly larger than those in the BART dataset. Therefore, the limited number of events scraped for the NS dataset was definitely too few to properly train forecasting algorithms.

Modes of transport. This research only covers the rail network in the California Bay Area and a part of the train network in the Netherlands. In reality, however, the commute of a passenger often does not stop at the destination station, but a passenger often uses a secondary form of transportation (like walking, cycling, bus, metro, taxi, et cetera) to arrive at their destination; for example at station Arnhem Centraal, which is also represented in for the NS, which observed that 30% of alighting passengers continued their trip by bus in 2022 [19]. Back in Section 4, we saw the number of events we could match with significant peaks in passenger flow decreased significantly as the distance to the event location increased. So, while these passengers might have travelled using the BART public



(a) Overall forecasting performance.



(b) Forecasting performance under event conditions.

Figure 26: Comparison of the forecasting performance on the NS test dataset between weights trained on the NS training set versus weights trained on the BART training set.

transportation network, we were not able to detect their presence in the data because not all of them would have arrived at the nearest BART station.

8 CONCLUSION

This research has tested and compared various forecasting algorithms and the impact of features pertaining to large events for passenger flow forecasting under different conditions. While the hypothesis of a positive performance impact of the event-related features in passenger flow forecasting cannot be accepted based on the results of this research, there have been some indications that including large events in passenger flow forecasting algorithms might yield improved forecasts under event conditions given the better event-indicators.

All of the selected forecasting algorithms performed admirably overall, but there were significant performance improvements for the MLP-based model architectures (namely the MLP and GNN algorithms). The Hybrid MLP algorithm, while having an MLP component, does not show these overall performance improvements; since the auto-regressive part of the Hybrid forecasting algorithms is bottlenecked by a single neuron output, which is not the case for the MLP and GNN algorithms, this indicates that there are more complex relationships in the past observations that cannot be solved by SARIMA or a single neuron and require multiple neurons (and potentially multiple layers of neurons) to model correctly.

The main problem of this research, however, lies in the fact that the currently selected event-related features are not enough to model the number of passengers in public transportation that are caused by events, as was already indicated by the non-linear results obtained in Figure 7 of Section 4. Presumably, one needs to obtain the actual (or anticipated) attendance of the event in order to correctly model the amount of extra passengers caused by the event; even though the linear relation showed by Santanam et al. [28] only accounts for two types of sport events at two train stations, it is likely that the linear relationship extends beyond the scope of the research by Santanam et al. [28].

That being said, event conditions are relatively rare, especially for larger events, so obtaining a dataset with enough large events for forecasting algorithms to learn their behaviour might be hard. Luckily, it was observed that algorithms with weights transferred from the BART dataset showed decent out-of-the-box performance on the NS dataset. Therefore, there is a lot of potential for using Transfer Learning in the case that the target dataset is insufficient.

9 FUTURE WORK

This section gives some pointers for future works based on the limitations of this research, as discussed in Section 7, and experience gained from doing this research.

Event data. As discussed in Section 7, venue capacity is not a good enough indicator for actual event attendance and, therefore, limits the potential performance of the different forecasting approaches used in this research. Future research into passenger flow forecasting under event conditions or public transportation network operators should approach various event/music/theatre locations to get a proper dataset of events along with attendance estimates.

Santanam et al. [28] showed a linear relationship between event attendance and extra passengers in public transportation for a small set of large sports events; if this relationship holds for many other types and sizes of events, then event attendance should be an excellent indicator for additional passengers in public transport.

Feature selection. The exogenous features selected for events in this research are based on some basic domain knowledge of the author. Many different target demographics of events will have different preferences and patterns when it comes to travel preferences. However, this is just one example of the many factors impacting the usage of public transportation for travel to and from events. So this means that there are potentially many better features to use for passenger flow forecasting under event conditions. Therefore, a possible direction of future work is to do a data analysis to investigate which external factors influence the number of additional passengers in public transport due to events.

Transfer learning. This research's small weight transfer experiment showed some decent out-of-the-box performance; *transfer learning* [23] using pre-trained weights from other datasets could offer better results for smaller datasets or require less computational power to converge to an optimum for the target dataset. This might be a solution in the case of forecasting under event conditions, as large impactful events are less common and will require more samples to learn properly.

Federated learning. Continuing in the direction of training weights on multiple datasets yields the next opportunity for future work. As was seen in the small transfer learning experiment, the out-of-the-box forecasting performance of transferred weights was quite decent even though the two datasets concerned strictly different

public transport networks from different socio-political environments. If this same thing works for many multiple public transportation networks, then there could be opportunities for inter-operator collaboration, in which multiple public transport operators would train a forecasting algorithm together, which would be able to forecast passenger flow on all of their respective networks. One of the ways to achieve this is through *federated learning* [13], in which multiple parties train the weights of a shared Machine Learning model without sharing each party's private dataset with any of the other parties involved. Future work could look into the feasibility of such a cooperation between operators.

Multi-modal transportation. As mentioned in the discussion in Section 7, this research only considers a single mode of transportation. In contrast, public transportation commutes might often use a secondary mode of transport after arriving at their destination station. Some future work might look into this by accounting for more favourable connections to secondary modes of transport passengers might use to get to their event; proper attribution of the likely alighting station for passengers attending an event might improve the performance of passenger flow forecasting under event conditions.

Model architectures. Finally, there are some potential improvements for the architectures of the various Machine Learning models applied in this research:

Bottleneck in Hybrid algorithms In the current implementation of the Hybrid algorithms, there is a single scalar output of the SARIMA component to the deviation forecasting component; this bottleneck is not present in the MLP model architecture, which performs significantly better. Therefore, a slightly larger bottleneck (in the form of a small vector) could be tried to see whether it would increase the performance of the Hybrid model architectures.

Attention The current GNN approach is based on a Message Passing Graph Neural Network, which – in this implementation – treats all incoming messages equally. However, spatial influences from one connection might be less impactful than those from another. Therefore, an attention component could be added to tune the different levels of importance of spatial information from neighbouring connections.

REPRODUCIBILITY

The results obtained in this research can be reproduced as both the code and one of the datasets are publicly available online:

- The BART dataset is published by San Francisco Bay Area Rapic Transit District [26] under the *Creative Commons Attribution License (cc-by)*:
  bart.gov/about/reports/ridership
- The code used in this research is copyrighted by Info Support B.V. and published under the *Apache-2.0 license* on GitHub:
  github.com/jeffreybakker/passenger-flow-forecasting

ACKNOWLEDGMENTS

First of all, I would like to thank the *Nederlandse Spoorwegen* (NS) for their collaboration in providing me with one of the datasets used in this research, as well as sharing some knowledge and insights about their experiences.

Second, I thank my direct supervisors, Tom van den Berg and dr. Elena Mocanu, for their supervision, input, and nice conversations during our weekly meetings. Next to that, I thank *Info Support* for their welcoming environment in which I have been allowed to perform my research.

Finally, I thank everyone who gave feedback on my work or assisted me in some other form throughout the process.

Most visualizations in this research are made using Python libraries matplotlib *[9],* seaborn *[32], and* networkx *[7].*

REFERENCES

[1] R. Balcombe, Roger Mackett, N. Paulley, John Preston, Jeremy Shires, Helena Titheridge, Mark Wardman, and P. White. 2004. The demand for public transport: A practical guide. *Balcombe, R. and Mackett, R. and Paulley, N. and Preston, J. and Shires, J. and Titheridge, H. and Wardman, M. and White, P. (2004) The demand for public transport: a practical guide. Technical report. Transportation Research Laboratory Report (TRL593). Transportation Research Laboratory, London, UK.* (Jan. 2004).

[2] Rajarshi Chattopadhyay and Chen-Khong Tham. 2022. Mixture of Experts based Model Integration for Traffic State Prediction. In *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring).* IEEE, Helsinki, Finland, 1–7. https://doi.org/10.1109/VTC2022-Spring54318.2022.9860682

[3] E. Chen, Z. Ye, C. Wang, and M. Xu. 2020. Subway passenger flow prediction for special events using smart card data. *IEEE Transactions on Intelligent Transportation Systems* 21, 3 (2020), 1109–1120. https://doi.org/10.1109/TITS.2019.2902405

[4] Robert B Cleveland, William S Cleveland, Jean E McRae, and Irma Terpenning. 1990. STL: A seasonal-trend decomposition. *J. Off. Stat* 6, 1 (1990), 3–73.

[5] DoStuff Media, LLC. [n. d.]. DoTheBay | What to do in The Bay Area. https://dothebay.com/ *Accessed 2023-04-11.*

[6] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (July 2019), 922–929. https://doi.org/10.1609/aaai.v33i01.3301922

[7] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring Network Structure, Dynamics, and Function using NetworkX. In *Proceedings of the 7th Python in Science Conference*, Gaël Varoquaux, Travis Vaught, and Jarrod Millman (Eds.). Pasadena, CA USA, 11 – 15.

[8] Josef Hoppe, Felix Schwinger, Henrik Haeger, Jonas Wernz, and Matthias Jarke. 2023. Improving the Prediction of Passenger Numbers in Public Transit Networks by Combining Short-Term Forecasts With Real-Time Occupancy Data. *IEEE Open Journal of Intelligent Transportation Systems* 4 (2023), 153–174. https://doi.org/10.1109/OJITS.2023.3251564 Conference Name: IEEE Open Journal of Intelligent Transportation Systems.

[9] J. D. Hunter. 2007. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* 9, 3 (2007), 90–95. https://doi.org/10.1109/MCSE.2007.55

[10] R.J. Hyndman and G. Athanasopoulos. 2021. *Forecasting: Principles and Practice (3rd ed)* (3rd edition ed.). OTexts: Melbourne, Australia. OTexts.com/fpp3. https://otexts.com/fpp3/

[11] W. Jiang and J. Luo. 2022. Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications* 207 (2022). https://doi.org/10.1016/j.eswa.2022.117921

[12] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. 2017. Time-series Extreme Event Forecasting with Neural Networks at Uber. *ICML 2017 Time Series Workshop* (June 2017).

[13] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. 2023. A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection. *IEEE Transactions on Knowledge and Data Engineering* 35, 4 (April 2023), 3347–3366. https://doi.org/10.1109/TKDE.2021.3124599 arXiv:1907.09693 [cs, stat].

[14] W. Li, M. Zhou, and H. Dong. 2019. Forecasting Model and Control Strategy of Large Passenger Flow under Large-scale Activity Conditions. 438–442. https://doi.org/10.1109/IMCEC46724.2019.8983909

[15] Wei Li, Min Zhou, Hairong Dong, Xingtang Wu, and Qi Zhang. 2021. Forecast of Passenger Flow of Urban Rail Transit Based on the DNNC Model. In *2021 33rd Chinese Control and Decision Conference (CCDC).* 4615–4620. https://doi.org/10.1109/CCDC52312.2021.9602086 ISSN: 1948-9447.

[16] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. arXiv. https://doi.org/10.48550/arXiv.1711.05101 arXiv:1711.05101 [cs, math].

[17] Eric L. Manibardo, Ibai Laña, and Javier Del Ser. 2020. Deep Learning for Road Traffic Forecasting: Does it Make a Difference? https://doi.org/10.48550/arXiv.2012.02260 arXiv:2012.02260 [cs, eess].

[18] MOJO. [n. d.]. Lowlands Green & Clean. https://lowlands.nl/green-clean/ *Accessed 2023-08-17.*

[19] Nederlandse Spoorwegen. [n. d.]. Reizigersgedrag Arnhem Centraal. https://dashboards.nsjaarverslag.nl/reizigersgedrag/arnhem-centraal *Accessed 2023-09-17.*

[20] Nederlandse Spoorwegen. 2020. Publicatiedocument "Bezetting". http://data.ndovloket.nl/docs/bezetting/Publicatiedocument%20Bezetting%20NS%20versie%201.0.pdf *Accessed 2023-04-23.*

[21] M. Ni, Q. He, and J. Gao. 2017. Forecasting the Subway Passenger Flow under Event Occurrences with Social Media. *IEEE Transactions on Intelligent Transportation Systems* 18, 6 (2017), 1623–1632. https://doi.org/10.1109/TITS.2016.2611644

[22] P. Noursalehi, H.N. Koutsopoulos, and J. Zhao. 2018. Real time transit demand prediction capturing station interactions and impact of special events. *Transportation Research Part C: Emerging Technologies* 97 (2018), 277–300. https://doi.org/10.1016/j.trc.2018.10.023

[23] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering* 22, 10 (Oct. 2010), 1345–1359. https://doi.org/10.1109/TKDE.2009.191 Conference Name: IEEE Transactions on Knowledge and Data Engineering.

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[25] Francisco C. Pereira, Filipe Rodrigues, Evgheni Polisciuc, and Moshe Ben-Akiva. 2015. Why so many people? Explaining Nonhabitual Transport Overcrowding With Internet Data. *IEEE Transactions on Intelligent Transportation Systems* 16, 3 (June 2015), 1370–1379. https://doi.org/10.1109/TITS.2014.2368119

[26] San Francisco Bay Area Rapic Transit District. [n. d.]. Bay Area Rapid Transit. https://www.bart.gov/ *Accessed 2023-04-16.*

[27] San Francisco Bay Area Rapic Transit District. 2023. New Train Car Project | Bay Area Rapid Transit. https://www.bart.gov/about/projects/cars *Accessed 2023-08-19.*

[28] Tejas Santanam, Anthony Trasatti, Pascal Van Hentenryck, and Hanyu Zhang. 2021. Public Transit for Special Events: Ridership Prediction and Train Optimization. http://arxiv.org/abs/2106.05359 arXiv:2106.05359 [cs, math].

[29] Jose Paolo Talusan, Ayan Mukhopadhyay, Dan Freudberg, and Abhishek Dubey. 2022. On Designing Day Ahead and Same Day Ridership Level Prediction Models for City-Scale Transit Networks Using Noisy APC Data. http://arxiv.org/abs/2210.04989 arXiv:2210.04989 [cs].

[30] Alejandro Tirachini, David A. Hensher, and John M. Rose. 2013. Crowding in public transport systems: Effects on users, operation and implications for the estimation of demand. *Transportation Research Part A: Policy and Practice* 53 (July 2013), 36–52. https://doi.org/10.1016/j.tra.2013.06.005

[31] Stylianos I. Vagropoulos, G. I. Chouliaras, E. G. Kardakos, C. K. Simoglou, and A. G. Bakirtzis. 2016. Comparison of SARIMAX, SARIMA, modified SARIMA and ANN-based models for short-term PV generation forecasting. In *2016 IEEE International Energy Conference (ENERGYCON)*. 1–6. https://doi.org/10.1109/ENERGYCON.2016.7514029

[32] Michael L. Waskom. 2021. seaborn: statistical data visualization. *Journal of Open Source Software* 6, 60 (2021), 3021. https://doi.org/10.21105/joss.03021

[33] Jonathan Wood, Zhengyao Yu, and Vikash V. Gayah. 2022. Development and evaluation of frameworks for real-time bus passenger occupancy prediction. *International Journal of Transportation Science and Technology* (March 2022). https://doi.org/10.1016/j.ijtst.2022.03.005

[34] G. Xue, S. Liu, L. Ren, Y. Ma, and D. Gong. 2022. Forecasting the subway passenger flow under event occurrences with multivariate disturbances. *Expert Systems with Applications* 188 (2022). https://doi.org/10.1016/j.eswa.2021.116057

[35] E. Yao, J. Hong, L. Pan, B. Li, Y. Yang, and D. Guo. 2021. Forecasting Passenger Flow Distribution on Holidays for Urban Rail Transit Based on Destination Choice Behavior Analysis. *Journal of Advanced Transportation* 2021 (2021). https://doi.org/10.1155/2021/9922660

[36] Jiexia Ye, Juanjuan Zhao, Kejiang Ye, and Chengzhong Xu. 2020. Multi-STGCnet: A Graph Convolution Based Spatial-Temporal Framework for Subway Passenger Flow Forecasting. In *2020 International Joint Conference on Neural Networks (IJCNN)*. 1–8. https://doi.org/10.1109/IJCNN48605.2020.9207049 ISSN: 2161-4407.

[37] J. Ye, J. Zhao, K. Ye, and C. Xu. 2022. How to Build a Graph-Based Deep Learning Architecture in Traffic Domain: A Survey. *IEEE Transactions on Intelligent Transportation Systems* 23, 5 (2022), 3904–3924. https://doi.org/10.1109/TITS.2020.3043250

[38] Jiexia Ye, JuanJuan Zhao, Liutao Zhang, ChengZhong Xu, Jun Zhang, and Kejiang Ye. 2020. A Data-Driven Method for Dynamic OD Passenger Flow Matrix Estimation in Urban Metro Systems. In *Big Data – BigData 2020 (Lecture Notes in Computer Science)*, Surya Nepal, Wenqi Cao, Aziz Nasridinov, MD Zakirul Alam Bhuiyan, Xuan Guo, and Liang-Jie Zhang (Eds.). Springer International Publishing, Cham, 116–126. https://doi.org/10.1007/978-3-030-59612-5_9

[39] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. 2023. *Dive into Deep Learning*. Cambridge University Press. https://D2L.ai *Accessed 2023-10-17.*

[40] P. Zhang. 2023. Analysis of Optimum Time Granularity Selection in traffic prediction based on Pearson Correlation Coefficient, Vol. 12510. https://doi.org/10.1117/12.2656808 ISSN: 0277-786X.

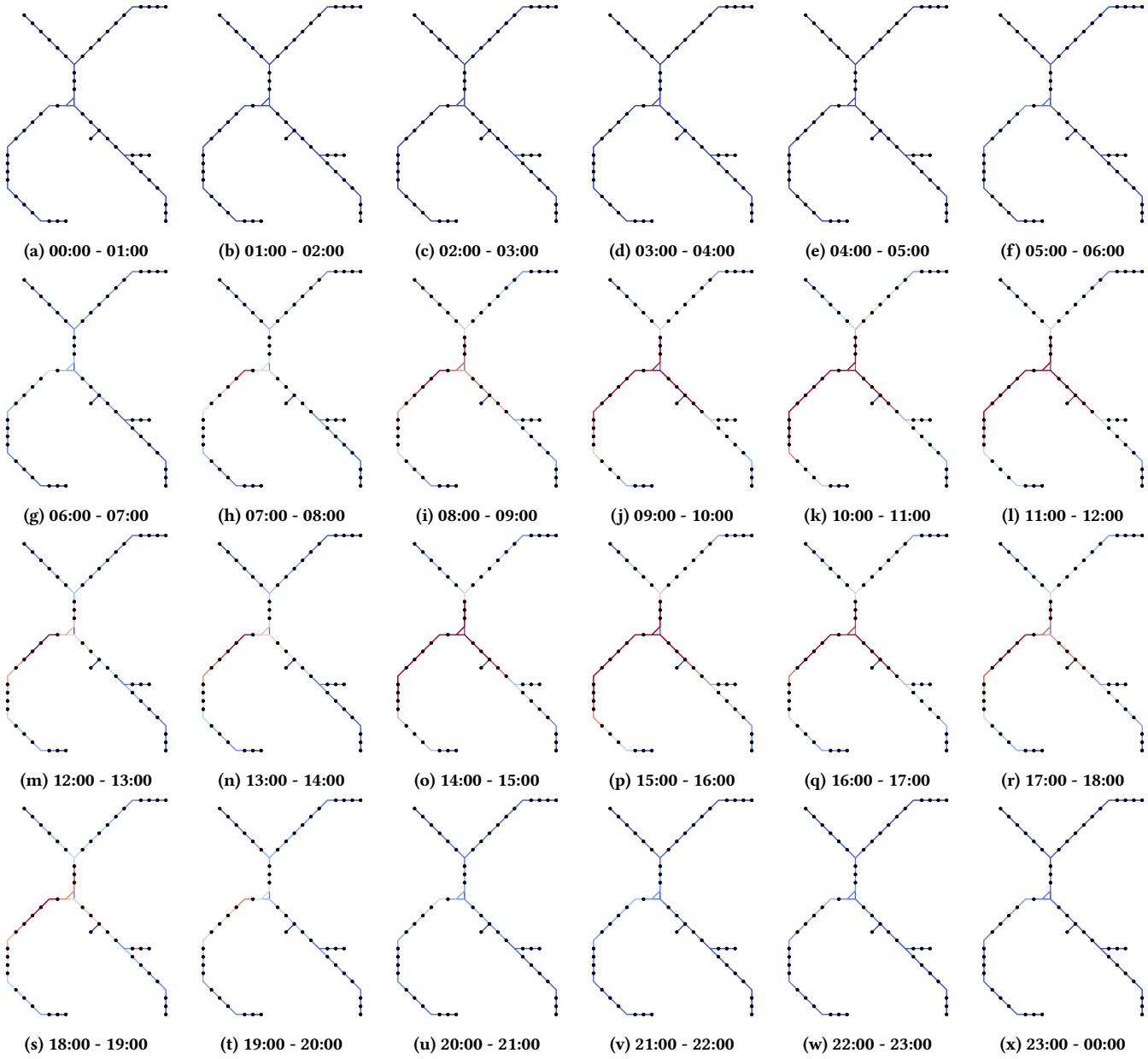
[41] Y. Zhao and Z. Ma. 2022. Naïve Bayes-Based Transition Model for Short-Term Metro Passenger Flow Prediction under Planned Events. *Transportation Research Record* 2676, 9 (2022), 309–324. https://doi.org/10.1177/03611981221086645

[42] Gaoxiang Zhou and Jinjin Tang. 2020. Forecast of Urban Rail Transit Passenger Flow in Holidays Based on Support Vector Machine Model. In *2020 5th International Conference on Electromechanical Control Technology and Transportation (ICECTT)*. 585–589. https://doi.org/10.1109/ICECTT50890.2020.00133

[43] Li Zhu, Fei Richard Yu, Yige Wang, Bin Ning, and Tao Tang. 2019. Big Data Analytics in Intelligent Transportation Systems: A Survey. *IEEE Transactions on Intelligent Transportation Systems* 20, 1 (Jan. 2019), 383–398. https://doi.org/10.1109/TITS.2018.2815678 Conference Name: IEEE Transactions on Intelligent Transportation Systems.

A VISUALIZED PASSENGER FLOW

The figure below, Figure A.1, visualizes passenger flow in the BART network throughout the day on *2022-06-20*:



(a) 00:00 - 01:00 (b) 01:00 - 02:00 (c) 02:00 - 03:00 (d) 03:00 - 04:00 (e) 04:00 - 05:00 (f) 05:00 - 06:00

(g) 06:00 - 07:00 (h) 07:00 - 08:00 (i) 08:00 - 09:00 (j) 09:00 - 10:00 (k) 10:00 - 11:00 (l) 11:00 - 12:00

(m) 12:00 - 13:00 (n) 13:00 - 14:00 (o) 14:00 - 15:00 (p) 15:00 - 16:00 (q) 16:00 - 17:00 (r) 17:00 - 18:00

(s) 18:00 - 19:00 (t) 19:00 - 20:00 (u) 20:00 - 21:00 (v) 21:00 - 22:00 (w) 22:00 - 23:00 (x) 23:00 - 00:00

Figure A.1: Visualized passenger flow throughout the day on *2022-06-20* in the BART dataset, where blue indicates very few passengers and red a lot of passengers.

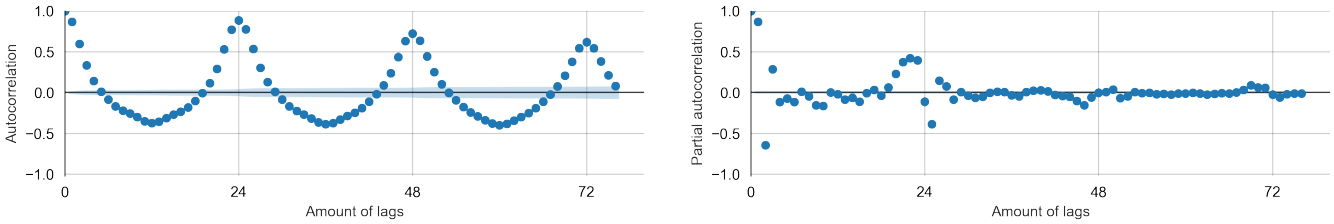
B SARIMA ORDER SELECTION

This appendix covers how the order $(p, d, q)(P, D, Q)_m$ for the SARIMA models is determined, namely by interpreting the data's stationarity, *autocorrelation* and *partial autocorrelation* as described by Hyndman and Athanasopoulos [10]. In this appendix, we focus on a single connection in the BART dataset in *2022*, namely between stations *Embarcadero* and *West Oakland*, and assume that the patterns on this connection are indicative of the rest of the network.

Stationarity. Whether the data is stationary is determined using the *Augmented Dickey-Fuller test* (ADF) at significance level $\alpha = 0.05$; the null-hypothesis of the ADF test is that the time series data has a unit root, and therefore would be non-stationary.

The null hypothesis is not rejected (p-value = 0.1226); therefore, the time series data is non-stationary, and we will select a differencing order of $d = 1$.

AR and MA order. The figure below (Figure B.1) shows the autocorrelation and partial autocorrelation: The autocorrelation gives for every lag k the correlation between y_t and y_{t-k} for all t in the dataset; and the partial autocorrelation, which is similar, but accounts for the influences of lags $1, 2, \ldots, k-1$ and is therefore a metric of the new information that lag k brings.



(a) Autocorrelation **(b) Partial autocorrelation**

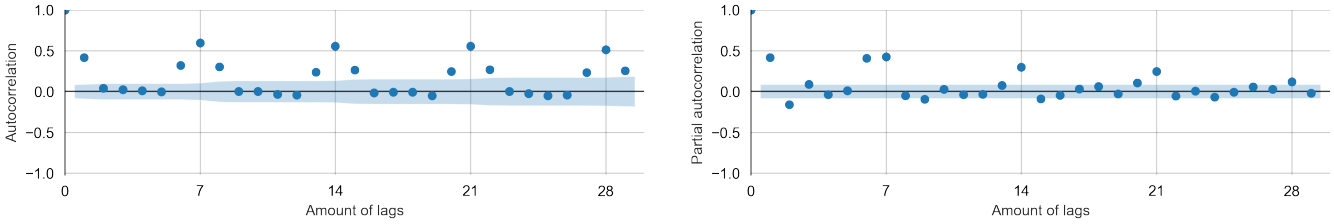
Figure B.1: Autocorrelation and partial autocorrelation on edge-level passenger flow from station *West Oakland* to *Embarcadero*.

Due to the sinusoidal resemblance of the autocorrelation plot (Figure B.1a), we will select an Auto-Regressive (AR) model over a Moving-Average (MA) model; therefore, we set $q = 0$. Then, observing the partial autocorrelation plot (in Figure B.1b), we select an auto-regressive order of $p = 23$ as it is the last significant partial correlation smaller than the seasonal lag (24) we will select in the next subsection.

B.1 Seasonal Order

As observed in Figure B.1b, there is a strong seasonal pattern with a wavelength of 24 lags; therefore, the seasonal lag of $m = 24$ has been selected for the SARIMA model. Then, similarly to what we did for the regular ARIMA order, we will determine the seasonal order, but this time on a set of every 24^{th} observation.

This time, the null hypothesis of the ADF test is rejected (p-value = 0.0232); therefore, the seasonal data is stationary, and we select a seasonal differencing order of $D = 0$.



(a) Seasonal autocorrelation **(b) Seasonal partial autocorrelation**

Figure B.2: Seasonal autocorrelation and partial autocorrelation on edge-level passenger flow from station *West Oakland* to *Embarcadero* for a seasonality of 24.

Similarly, from observing Figure B.2, we set $Q = 0$ and $P = 7$. So, with these final two parameters, we have selected order $(23, 1, 0)(7, 0, 0)_{24}$ for SARIMA in the BART dataset; the order of the NS dataset is found using the same methodology and is equal to $(47, 1, 0)(7, 0, 0)_{48}$.

C TRAINING TIMES

The table below (Table C.1) shows the training times for the various forecasting algorithms using an *Intel(R) Core(TM) i7-12800H* with a *NVIDIA GeForce MX550* GPU:

	# Epochs	Time per epoch	Total training time
SARIMA	10	~ 7.4 min	~ 1.2 hrs
SARIMAX	10	~ 9.6 min	~ 1.6 hrs
LR	25	~ 9.4 min	~ 3.9 hrs
MLP	25	~ 9.5 min	~ 4.0 hrs
Hybrid LR	25	~ 9.6 min	~ 4.0 hrs
Hybrid MLP	25	~ 9.5 min	~ 4.0 hrs
GNN	50	~ 3.3 min	~ 2.7 hrs

Table C.1: Training times for the forecasting algorithms.

D LEARNING CURVES

Table D.1 (on the next page) gives an overview of the improvement in the performance of the various algorithms on the training and validation datasets during training.

All lines appear to have converged to their (horizontal) asymptote, suggesting that each forecasting algorithm has reached as much as there is to learn, given the distribution of the dataset and the selected hyperparameters. For most models, all five lines (for the training or validation set separately) converge to the same point, except for the GNN algorithm, for which at least one run has converged to a local optimum.

Furthermore, there was no sudden increase in validation errors as the training errors decreased. So, since the training set and validation set are strictly separated (temporally), this suggests that none of the forecasting algorithms overfitted on the training data.

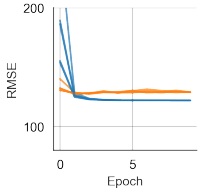
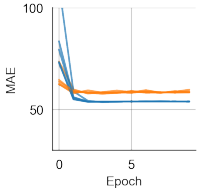
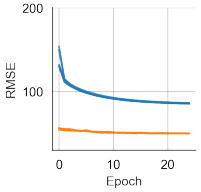
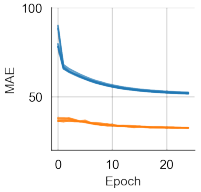
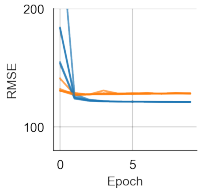
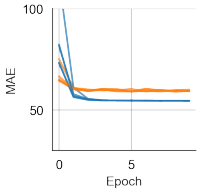
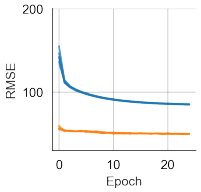
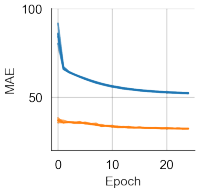
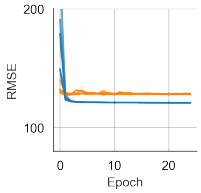
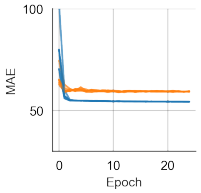
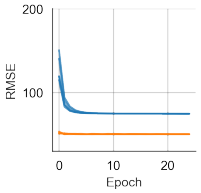
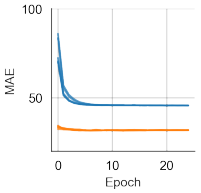
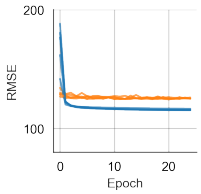
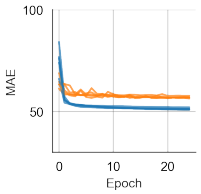
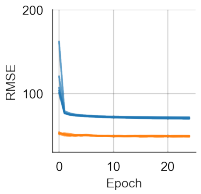
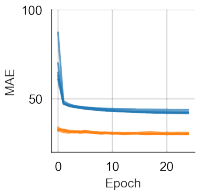
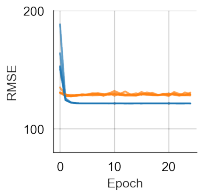
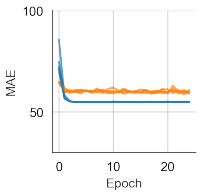
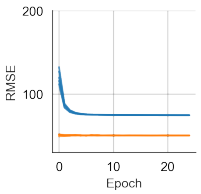
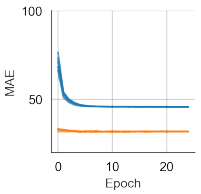
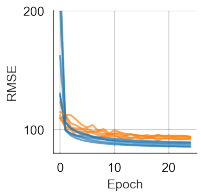
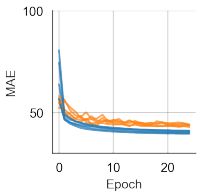
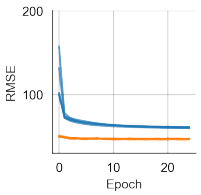
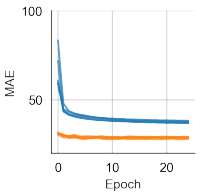
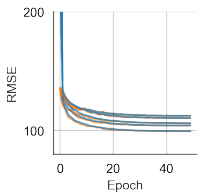
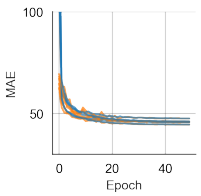
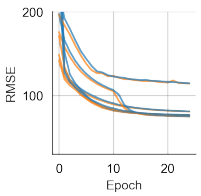
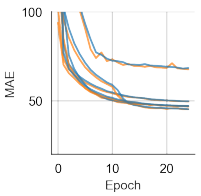
Algorithm	BART RMSE	BART MAE	NS RMSE	NS MAE
SARIMA				
SARIMAX				
Hybrid LR				
Hybrid MLP				
LR				
MLP				
GNN				

Table D.1: 1-step forecasting performance of algorithms during training with training set in ***blue*** and validation set in ***orange.***

E COMPARISON OF FORECASTING PERFORMANCE

This appendix gives more detailed performance results for the averaged sets of weights per forecasting algorithm. It is meant to complement this thesis's results section (Section 6). The mean μ and standard deviation σ of the aggregated metrics per forecasting algorithm is given as $\mu \pm \sigma$ throughout this appendix.

First up is the overall forecasting performance of the various forecasting algorithms in Table E.1; looking at the MAE, RMSE and R^2 metrics, all forecasting algorithms have very stable results, except for the GNN approaches, which have significantly more variance between different sets of weights.

		MAE	**MAPE (%)**	**RMSE**	R^2
Without event data	SARIMA	67.02 ± 0.67	101.76 ± 6.12	143.83 ± 0.24	0.9616 ± 0.0001
	SARIMAX	67.36 ± 0.26	110.49 ± 2.60	144.03 ± 0.24	0.9615 ± 0.0001
	LR	67.59 ± 0.57	110.64 ± 8.47	144.72 ± 0.72	0.9611 ± 0.0004
	MLP	50.94 ± 0.71	49.90 ± 2.24	103.55 ± 1.70	0.9801 ± 0.0007
	Hybrid LR	67.29 ± 0.11	111.36 ± 1.62	143.95 ± 0.23	0.9615 ± 0.0001
	Hybrid MLP	66.08 ± 0.11	78.94 ± 7.53	142.46 ± 0.28	0.9623 ± 0.0002
	GNN	58.44 ± 1.11	66.26 ± 4.03	129.40 ± 5.37	0.9689 ± 0.0026
With event data	SARIMA	-	-	-	-
	SARIMAX	67.37 ± 0.25	110.50 ± 2.59	144.04 ± 0.23	0.9615 ± 0.0001
	LR	67.72 ± 0.57	110.67 ± 8.47	144.80 ± 0.65	0.9611 ± 0.0004
	MLP	51.14 ± 0.69	49.94 ± 2.24	103.90 ± 1.71	0.9799 ± 0.0007
	Hybrid LR	67.35 ± 0.10	111.38 ± 1.62	143.96 ± 0.21	0.9615 ± 0.0001
	Hybrid MLP	66.10 ± 0.12	78.97 ± 7.53	142.37 ± 0.34	0.9624 ± 0.0002
	GNN	58.42 ± 1.12	66.27 ± 4.03	129.28 ± 5.36	0.9689 ± 0.0026

Table E.1: Difference in overall 1-step forecasting performance in the BART dataset.

Then, Table E.2 shows a different subset of the testing data; therefore, the results in this table cannot be directly compared to the previous table. Since this table zooms in on event conditions in the test dataset, the differences in performance of all the models are more significant as the entire goal of this table is to show how the forecasting performance changes when including event data in the forecast. Surprisingly, the variance of the performance metrics between the forecasts with and without event data is very similar.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	125.07 ± 1.54	14.93 ± 0.39	211.19 ± 1.00	0.9588 ± 0.0004
	SARIMAX	121.42 ± 2.44	14.66 ± 0.15	205.76 ± 4.44	0.9608 ± 0.0013
	LR	115.51 ± 1.16	15.73 ± 0.12	197.66 ± 2.24	0.9635 ± 0.0007
	MLP	91.78 ± 1.37	13.41 ± 0.25	161.11 ± 2.43	0.9765 ± 0.0005
	Hybrid LR	115.57 ± 1.26	15.05 ± 0.16	198.08 ± 2.38	0.9619 ± 0.0006
	Hybrid MLP	114.07 ± 0.38	15.54 ± 0.21	196.46 ± 1.29	0.9621 ± 0.0007
	GNN	102.05 ± 2.11	14.44 ± 0.24	182.42 ± 4.57	0.9714 ± 0.0014
With event data	SARIMA	-	-	-	-
	SARIMAX	121.51 ± 2.47	14.68 ± 0.15	205.95 ± 4.49	0.9607 ± 0.0014
	LR	118.24 ± 1.13	16.37 ± 0.19	198.96 ± 1.81	0.9630 ± 0.0006
	MLP	96.48 ± 0.90	14.26 ± 0.14	166.55 ± 2.44	0.9749 ± 0.0005
	Hybrid LR	116.83 ± 1.28	15.35 ± 0.16	198.36 ± 2.39	0.9618 ± 0.0006
	Hybrid MLP	116.12 ± 0.50	16.20 ± 0.20	197.14 ± 1.72	0.9619 ± 0.0009
	GNN	102.66 ± 2.18	14.57 ± 0.18	182.77 ± 4.77	0.9713 ± 0.0014

Table E.2: 1-step forecasting performance under event conditions in the BART dataset.

The final table for the BART dataset is Table E.3, which shows the MAE of the forecasts of the various forecasting algorithms as the forecast goes further into the future. Unsurprisingly, for most models, the standard deviation (and thus the variance) increases for forecasts further into the future.

	1 hour	6 hours	72 hours
SARIMA	67.02 ± 0.67	75.28 ± 1.10	80.07 ± 1.84
SARIMAX	67.37 ± 0.25	76.10 ± 0.45	81.26 ± 0.97
LR	67.72 ± 0.57	76.55 ± 0.82	82.12 ± 1.08
MLP	51.14 ± 0.69	63.28 ± 1.19	67.41 ± 1.85
Hybrid LR	67.35 ± 0.10	75.58 ± 0.08	80.09 ± 0.41
Hybrid MLP	66.10 ± 0.12	73.37 ± 0.16	77.48 ± 0.63
GNN	58.42 ± 1.12	66.79 ± 1.18	69.59 ± 1.37

Table E.3: Comparison of multi-step rolling forecasting performance in the BART dataset measured with MAE over the entire network.

E.1 NS Dataset

Continuing to the NS dataset, Table E.4 gives the overall forecasting performance of the forecasting algorithms. One thing that stands out is that – in contrast to the BART dataset – the SARIMA model has the lowest standard deviation across most performance metrics compared to the rest of the forecasting algorithms. The standard deviation of the GNN algorithm is the biggest, which is quite logical, considering this approach converged to many different local optima (as seen in Appendix D).

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	82.79 ± 0.17	28.51 ± 0.20	125.88 ± 0.23	0.7897 ± 0.0008
	SARIMAX	80.00 ± 0.24	26.90 ± 0.04	122.62 ± 0.29	0.8004 ± 0.0010
	LR	78.85 ± 0.26	24.89 ± 0.02	126.37 ± 0.33	0.7880 ± 0.0011
	MLP	72.95 ± 1.89	23.43 ± 1.00	114.11 ± 2.68	0.8271 ± 0.0082
	Hybrid LR	78.62 ± 0.42	24.92 ± 0.03	126.06 ± 0.51	0.7890 ± 0.0017
	Hybrid MLP	75.15 ± 0.97	24.45 ± 0.32	117.91 ± 1.83	0.8154 ± 0.0057
	GNN	82.87 ± 8.95	27.57 ± 4.19	127.92 ± 14.84	0.7804 ± 0.0541
With event data	SARIMA	-	-	-	-
	SARIMAX	80.00 ± 0.24	26.91 ± 0.04	122.62 ± 0.30	0.8004 ± 0.0010
	LR	79.49 ± 0.29	25.21 ± 0.04	126.92 ± 0.37	0.7861 ± 0.0012
	MLP	73.62 ± 1.91	23.72 ± 0.99	114.72 ± 2.67	0.8252 ± 0.0082
	Hybrid LR	79.27 ± 0.46	25.23 ± 0.04	126.60 ± 0.55	0.7872 ± 0.0019
	Hybrid MLP	76.09 ± 0.88	24.90 ± 0.34	118.90 ± 1.69	0.8123 ± 0.0053
	GNN	82.87 ± 8.95	27.57 ± 4.19	127.93 ± 14.84	0.7804 ± 0.0541

Table E.4: Difference in overall 1-step (30 minutes) forecasting performance on the NS dataset.

The situation concerning the standard deviation of the forecasting metrics is slightly different under event conditions, as seen in Table E.5. However, the GNN algorithm still stands out, with a significantly larger standard deviation across all metrics.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	72.33 ± 1.01	28.15 ± 0.43	92.84 ± 1.30	0.7703 ± 0.0065
	SARIMAX	66.53 ± 0.24	25.46 ± 0.12	86.48 ± 0.44	0.8006 ± 0.0020
	LR	54.04 ± 0.61	20.53 ± 0.37	70.12 ± 0.70	0.8723 ± 0.0061
	MLP	50.91 ± 2.07	18.99 ± 1.06	67.17 ± 2.01	0.8873 ± 0.0068
	Hybrid LR	54.12 ± 0.45	20.88 ± 0.11	70.19 ± 0.52	0.8683 ± 0.0019
	Hybrid MLP	53.75 ± 0.83	21.51 ± 0.52	69.60 ± 0.96	0.8705 ± 0.0036
	GNN	62.99 ± 16.72	23.31 ± 6.11	83.49 ± 22.78	0.8167 ± 0.1095
With event data	SARIMA	-	-	-	-
	SARIMAX	66.55 ± 0.34	25.60 ± 0.26	86.47 ± 0.57	0.8006 ± 0.0028
	LR	63.87 ± 1.05	25.30 ± 0.55	83.92 ± 1.63	0.8171 ± 0.0080
	MLP	61.40 ± 2.28	23.49 ± 0.97	81.85 ± 3.74	0.8326 ± 0.0147
	Hybrid LR	63.66 ± 1.21	25.57 ± 0.55	83.48 ± 1.79	0.8137 ± 0.0080
	Hybrid MLP	67.64 ± 1.92	28.12 ± 1.12	91.17 ± 3.01	0.7776 ± 0.0148
	GNN	63.03 ± 16.70	23.31 ± 6.11	83.54 ± 22.75	0.8165 ± 0.1094

Table E.5: 1-step (30 minutes) forecasting performance under event conditions around station Q in the NS dataset.

The same trend continues in the final table of this subsection, Table E.6. Moreover, just like for the BART dataset, the standard deviation of the MAE increases as the forecasts go further into the future.

	30 minutes	1 hour	6 hours	72 hours
SARIMA	82.79 ± 0.17	99.25 ± 0.18	119.13 ± 0.38	139.30 ± 0.60
SARIMAX	80.00 ± 0.24	95.37 ± 0.35	116.46 ± 0.78	153.91 ± 1.48
LR	79.49 ± 0.29	95.77 ± 0.39	122.46 ± 1.50	179.99 ± 8.28
MLP	73.62 ± 1.91	83.90 ± 3.22	119.15 ± 14.79	160.33 ± 16.17
Hybrid LR	79.27 ± 0.46	95.45 ± 0.63	121.24 ± 2.25	173.86 ± 12.85
Hybrid MLP	76.09 ± 0.88	90.84 ± 1.99	127.40 ± 7.24	164.70 ± 21.86
GNN	82.87 ± 8.95	99.67 ± 7.88	126.63 ± 10.39	182.35 ± 37.23

Table E.6: Comparison of multi-step rolling forecasting performance measured with MAE over the entire network in the NS dataset.

E.2 Transferred Weights

Finally, this subsection shows the out-of-the-box forecasting performance on the NS dataset of all forecasting algorithms with weights trained on the BART dataset. Similarly to what was concluded in Section 6, even though the results lie within the same ballpark, the out-of-the-box performance of the forecasting algorithms with transferred weights is measurably worse than the performance of the forecasting algorithms trained on the NS dataset and tested on the NS dataset. This becomes abundantly clear when looking at the results in Tables E.7 and E.8.

However, one thing that stands out is how the out-of-the-box long-term multi-step rolling forecasts for the forecasting algorithms with transferred weights (in Table E.9) are significantly better than the forecasts from the weights trained on the NS dataset (in Table E.6). Presumably, the weights from the BART network have learned to properly take both daily and weekly seasonalities into account (as can be seen in Figure 18), whereas the weights trained on the NS dataset leave some more significant correlations in residuals around 24 hours (48 lags) and 7 days (336 lags), as can be seen in Figure 25.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	91.88 ± 0.60	35.90 ± 0.33	130.26 ± 0.76	0.7822 ± 0.0025
	SARIMAX	91.62 ± 0.54	35.62 ± 0.17	129.76 ± 0.66	0.7839 ± 0.0022
	LR	92.19 ± 1.13	35.69 ± 0.22	130.59 ± 1.42	0.7811 ± 0.0048
	MLP	80.92 ± 1.14	28.21 ± 0.11	118.94 ± 3.83	0.8183 ± 0.0117
	Hybrid LR	92.16 ± 0.53	35.81 ± 0.14	130.41 ± 0.61	0.7817 ± 0.0020
	Hybrid MLP	91.34 ± 0.47	35.18 ± 0.19	130.53 ± 0.75	0.7813 ± 0.0025
	GNN	91.48 ± 1.34	32.36 ± 0.89	133.68 ± 3.80	0.7705 ± 0.0131
With event data	SARIMA	-	-	-	-
	SARIMAX	91.61 ± 0.53	35.62 ± 0.17	129.75 ± 0.65	0.7839 ± 0.0022
	LR	92.23 ± 1.10	35.72 ± 0.23	130.63 ± 1.39	0.7810 ± 0.0047
	MLP	80.96 ± 1.14	28.21 ± 0.14	118.98 ± 3.87	0.8182 ± 0.0118
	Hybrid LR	92.19 ± 0.53	35.83 ± 0.14	130.43 ± 0.61	0.7817 ± 0.0020
	Hybrid MLP	91.32 ± 0.46	35.14 ± 0.19	130.53 ± 0.74	0.7813 ± 0.0025
	GNN	91.49 ± 1.35	32.36 ± 0.90	133.70 ± 3.81	0.7704 ± 0.0131

Table E.7: Difference in overall 1-step out-of-the-box forecasting performance for weights trained on the BART dataset and tested on the NS dataset.

		MAE	MAPE (%)	RMSE	R^2
Without event data	SARIMA	83.13 ± 0.68	32.35 ± 0.16	105.71 ± 0.84	0.7333 ± 0.0042
	SARIMAX	83.19 ± 0.73	32.07 ± 0.20	106.43 ± 0.63	0.7296 ± 0.0032
	LR	83.62 ± 1.54	32.05 ± 0.23	107.04 ± 2.02	0.7265 ± 0.0103
	MLP	78.04 ± 5.33	28.83 ± 1.79	100.88 ± 4.29	0.7569 ± 0.0207
	Hybrid LR	83.62 ± 0.59	32.13 ± 0.12	106.82 ± 0.70	0.7277 ± 0.0036
	Hybrid MLP	81.29 ± 0.69	30.22 ± 0.27	103.50 ± 0.86	0.7444 ± 0.0043
	GNN	84.12 ± 2.96	30.74 ± 0.51	108.92 ± 3.26	0.7167 ± 0.0169
With event data	SARIMA	-	-	-	-
	SARIMAX	83.10 ± 0.70	32.08 ± 0.20	106.26 ± 0.60	0.7305 ± 0.0031
	LR	84.54 ± 0.97	32.62 ± 0.26	108.10 ± 1.28	0.7211 ± 0.0066
	MLP	78.30 ± 5.08	28.67 ± 1.17	101.48 ± 3.20	0.7541 ± 0.0155
	Hybrid LR	84.33 ± 0.57	32.43 ± 0.14	107.45 ± 0.62	0.7245 ± 0.0032
	Hybrid MLP	81.17 ± 0.77	29.73 ± 0.26	103.71 ± 1.24	0.7433 ± 0.0061
	GNN	84.31 ± 2.88	30.73 ± 0.47	109.30 ± 3.23	0.7147 ± 0.0168

Table E.8: 1**-step out-of-the-box forecasting performance under event conditions around station *Q* for weights trained on the BART dataset and tested on the NS dataset.**

	1 hour	6 hours	72 hours
SARIMA	91.88 ± 0.60	105.96 ± 0.69	105.56 ± 1.37
SARIMAX	91.61 ± 0.53	105.31 ± 0.78	105.38 ± 1.21
LR	92.23 ± 1.10	106.35 ± 1.46	107.02 ± 2.39
MLP	80.96 ± 1.14	105.82 ± 1.03	108.30 ± 0.25
Hybrid LR	92.19 ± 0.53	106.32 ± 0.69	106.80 ± 1.20
Hybrid MLP	91.32 ± 0.46	106.54 ± 0.64	107.92 ± 0.90
GNN	91.49 ± 1.35	103.48 ± 1.07	108.65 ± 2.53

Table E.9: Comparison of out-of-the-box multi-step rolling forecasting performance measured with MAE over the entire network for weights trained on the BART dataset and tested on the NS dataset.