Improving the imperfect passenger flow at Eindhoven Airport

Brom, M.F.

Award date:
2019

Department of Industrial Engineering and Innovation Science
Department of Mathematics and Computer Science

Improving the imperfect passenger flow at Eindhoven Airport

M.F. Brom (0810623)

BSc in Industrial Engineering and Management Science

In partial fulfillment of the requirements for the degree of

Master of Science in Operations Management and Logistics
Master of Science in Business Information Systems

Supervisors:
Dr. Ir. H. (Rik) Eshuis - TU/e
Dr. D. (Dirk) Fahland - TU/e
S. (Sven) Bastianen MSc - EA

Eindhoven, Thursday 7th March, 2019

Abstract

Numerous airports across the globe suffer from a capacity crunch in which limited infrastructure capacity meets continuously expanding passenger numbers. At Eindhoven Airport, this capacity crunch results in an increasingly imperfect passenger flow that is subject to considerable queues and general crowdedness across the terminal. Our research focuses specifically on Eindhoven Airport's security filter, which can be considered to cause the most severe issues within the outbound passenger flow. The airport currently applies a reactive flow management style in which decision makers attempt to resolve issues within the passenger flow after they have already emerged. We propose to enable an proactive flow management style that instead focuses on issue prevention by means of a real-time decision-support model. The decision-support model is three-fold and provides the decision makers with a real-time three hour ahead forecast of passengers that will arrive at the security filter, an expected waiting time for the 12 time intervals of 15 minutes within these three hours and finally the model suggests actions that the decision makers can take to mitigate upcoming exploding waiting times. The short-term passenger flow forecast is obtained through a newly developed multi-source multivariate gated recurrent unit neural network forecasting model that relies on temporal links between the data sources. A pilot phase displayed the decision-support models usefulness in predicting upcoming issues and its ability to propose feasible mitigating actions. Finally, deployment of the model is expected to lead to an increasingly *perfect* passenger flow.

Executive summary

Problem context

This report describes research on improving the increasingly imperfect passenger flow at Eindhoven Airport (EA), in which the airport experiences an increased probability of bottlenecks whilst at the same time resolution is difficult. Examples of this imperfect passenger flow are intolerable long queues at the security filter or passenger congestion in pathways. A perfect passenger flow is seen as a seamless flow for passengers, without severe bottlenecks. This research focuses specifically on the outbound passenger flow (departing passengers). From a passenger perspective, the start of the outbound passenger flow is the moment when the departing passenger arrives at the airport terminal and it ends when the passenger boards the aircraft.

Two main causes are found that result in an increasingly imperfect passenger flow at EA. Firstly, the airport experiences a growing *capacity crunch*, in which limited infrastructural capacity meets continuously expanding passenger numbers. This increases the probability of bottlenecks and at the same time reduces the solution scope. Secondly, the negative effects of an expanding capacity crunch are found to be amplified by *inadequate passenger flow management*. Passenger flow management is defined as all actions and steps that are taken by an airport to manage the flow of passengers across the airport. This includes both long-term flow management in the form of resource planning (e.g. number of open security lanes) and short-term flow management in the day-to-day operation (e.g. ad-hoc staff changes due to delays). Moreover, the research finds that inadequate flow management on the one hand is a result of *inefficient resource planning* due to the limited usage of historical data and on the other hand is a result of a *reactive flow management style* in the day-to-day operation due to the limited usage of real-time or live data. A reactive flow management style is focused on issue resolving opposed to a more desirable proactive flow management style that is focused on issue prevention.

Research goals and questions

EA desires to have a perfect passenger flow with limited operational bottlenecks. For EA the value of this research therefore lies in addressing the issues as defined in the previous section. The research's practical goal is to provide insights on how to reduce the severity and number of bottlenecks experienced in the outbound passenger flow, that are a result of the capacity crunch and that are amplified by inadequate flow management.

The research finds that in order to address this goal, addressing the inadequate reactive flow management style by enabling a *proactive flow management style* is most relevant from both a scientific and business point of view. A proactive flow management style is seen as a style in which the main focus is on issue prevention so as to create a seamless passenger flow. The research finds that other options to address the practical goal like expanding infrastructure capacity, limiting growth or improving resource planning are either unfeasible from a business point of view or less interesting from a scientific point of view.

To enable a proactive flow management style, the stakeholders identify that they require real-time insight into the current state of the outbound passenger flow and more importantly require insight in the short-term future state of the flow. This future state can primarily be obtained by real-time forecasting of the short-term passenger flow. Subsequently, these forecasts can be used to provide additional decision-support to the stakeholders to finally enable a proactive flow management style. To illustrate, additional decision-support for example can be provided in the form of derived statistics (e.g. computing the expected queue length by combining the forecast with the planned capacity) or action proposals (e.g. provide the stakeholders with a range of possible mitigating actions for upcoming bottlenecks). Concluding, the practical goal can be addressed by enabling proactive flow management through short-term passenger flow forecasting and additional decision-support. This finally leads to a main research question:

How can proactive flow management be enabled by using real-time short-term passenger flow forecasting and providing additional decision-support in order to reduce the severity and number of bottlenecks experienced in an airport terminal environment?

In order to effectively and thoroughly answer the research question, the Cross Industry Standard Process for Data Mining (CRISP-DM) methodology is used to structure this research project. In this methodology, *business understanding* is the initial phase and focuses on establishing deeper understanding of the bottlenecks, their underlying causes and the related current flow management. The subsequent *data understanding* phase is concerned with identifying which data exists in the outbound passenger flow context, where the data is located and identifying the quality and availability of the data. Following, the *data preparation* phase focuses on constructing useful data sets for the forecasting model. In the *modelling phase* different techniques are discovered, selected and applied to build a short-term forecasting model and an additional decision-support model. These models are then tested by means of a pilot phase in the *evaluation phase.* It is important to indicate that although the CRISP-DM framework provides an idealized sequence of stages, these stages do not necessarily follow each other slavishly, but are often performed iteratively. The stages can be performed in a different order and it might be necessary to backtrack to previous stages and repeat certain actions.

In the business understanding phase, seven different bottlenecks are identified that are eminent in the outbound passenger flow: unacceptable crowdedness at the departure gates and intolerable queue lengths at the self-service baggage drop-off units, assistance counters, check-in counters, airline service counters, security filter and border control counters. The resulting scope when applying the main research question on this whole outbound passenger flow is too large for a single master thesis project. The research finds that solely resolving the research question for EA's security filter is adequate to accomplish the objectives. The security filter is found to be the most critical bottleneck, causing both the highest number and most severe flow problems. Concluding, the research focuses on enabling a proactive flow management style for EA's security filter by providing real-time short-term passenger flow forecasts and additional decision-support to the relevant stakeholders. These stakeholders can use this decision-support to proactively prevent issues and with that reduce the severity and number of experienced bottlenecks.

Forecast model

The forecast model's goal aims to predict the number of passengers that will arrive at the security filter within the near future in real-time. In discussions with the stakeholders it was identified that being able to look three hours ahead will suffice to enable a pro-active flow management style. Furthermore, the three hours are split into 12 time intervals of an equal predetermined length of 15 minutes. The model thus needs to be able to forecast the number of passengers that will arrive in each of the 12 upcoming time intervals.

In order to construct a model that is able to forecast this, the data features that are used as input and outcome in the model need to be identified. The data features are found through the use of expert knowledge and an analysis of all data sources that are related to the outbound passenger flow. Real-time data streams are required in order to use the forecasts in the day-to-day operation. Three data sources are identified that are most relevant and can also provide real-time data: AirDCS is the departure control systems that contains passenger information, SkyGuide is the flight information system and Parkbase contains all parking lot information.

Using the analysis of these data sources and additional expert knowledge, fourteen input data features are identified that are expected to be associated with the short-term expected number of passengers that arrive at the security filter. These fourteen data features are all aggregated to 15 minutes and are either calendar related (e.g. hour of day), passenger related (e.g. number of baggage items dropped off), transport related (e.g. number of car entries at car parks) or flight related (e.g. number of departing flights).

Furthermore, since the model aims to forecast the number of passengers that will arrive at the security filter, a related outcome variable needs to be identified. Analysis finds that passengers are required to scan their boarding card at the pre-security filter prior to arriving at the security filter. These scan events are captured within AirDCS in real-time and are considered to be the moment that a passenger arrives at the security filter and are therefore used as the outcome variable. To sum up, the forecast model uses fourteen input features to forecast the number of pre-security filter scan events in the 12 upcoming 15 minute time steps.

An elaborate literature study results in the conclusion that gated recurrent unit (GRU) neural networks are the most applicable forecast models for this research's purposes. Variations of this type of neural network and other forecast models are build, tested and compared. The research finds that a basic GRU model with a single hidden layer outperforms the other models and is able to forecast the short-term passenger flow with satisfactory accuracy.

Decision-support model

The real-time short-term forecasts are used in a decision-support model that has the form of a dashboard. The decision-support model is three-fold: it provides the decision makers with a three hour ahead forecast of passengers that will arrive at the security filter, an expected waiting time for the 12 time steps of 15 minutes within these three hours and finally the model suggests actions that the decision makers can take to mitigate upcoming exploding waiting times. A screen-shot of the dashboard that includes all three elements is provided in Figure 1.

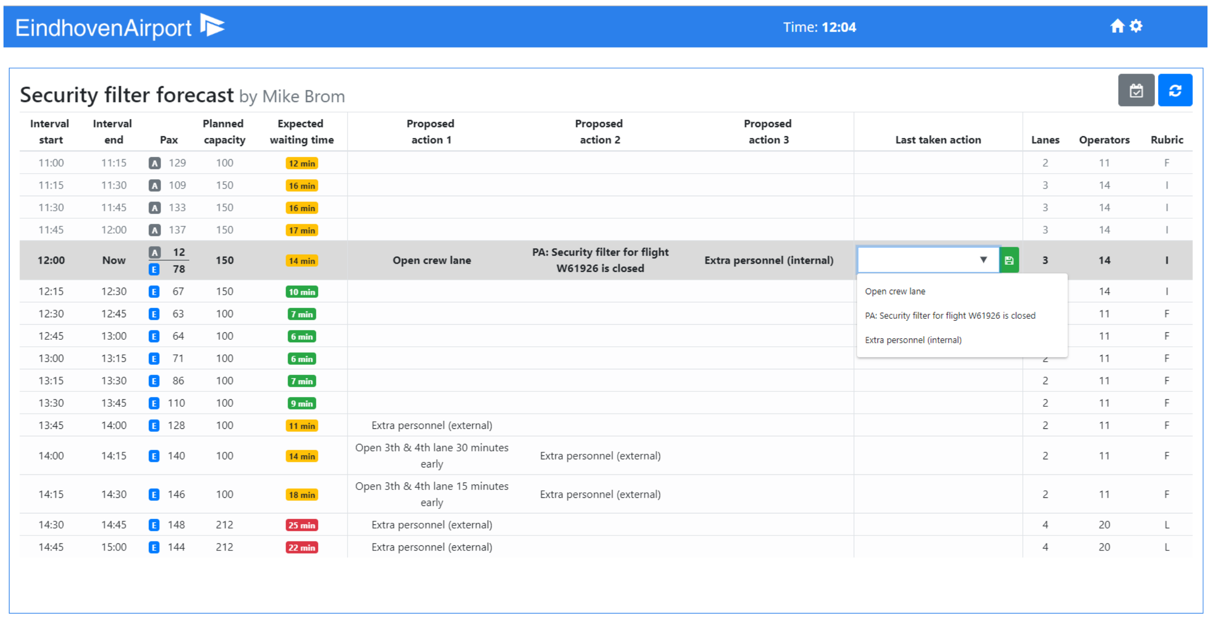
The expected waiting times are computed by offsetting the flow forecast to the planned security filter capacity. An *M(t)/G(t)/c(t)* queue approximation model is used in which the arrival rate, service time and number of lanes is different for each 15 minute time step. The basic idea behind the approximation relies on a loss queuing model in which the loss models of consecutive periods are connected via an artificial backlog. This backlog equals the loss rate of the preceding period. Passengers that are blocked/lost in one period are thus added to the queue of the succeeding period. Several parameters in this approximation allow for tuning in order to make the calculated expected waiting times match the actual waiting times.

Furthermore, the range of possible actions to mitigate expected undesirable waiting times is constructed for each of the 12 upcoming time steps. The range is different for every time step since not all actions can be executed within the remaining time between that time step and the current time. The range of actions that are actually proposed is dynamic and depends on the current conditions (e.g. one cannot open another security lane if six lanes are already open).

Conclusions and recommendations

The decision-support model has been deployed during a 2.5 week pilot phase. Through a quantitative and qualitative evaluation of this pilot phase the (expected) usefulness and (perceived) accuracy of the decision-support model are identified. The forecast model performed in similar fashion in terms of accuracy during the pilot phase as it did during model development. Furthermore, a qualitative evaluation with the users results in the conclusion that the model has a lot of potential and will definitely be used after the users establish additional confidence in the its accuracy. Although the model is not optimally tuned, during the pilot phase occurrences were already found during which it was able to predict upcoming bottlenecks and propose actions that were reasonably executable. Furthermore, the decision makers were adamant about using the model even after the pilot phase and it is therefore still available. Concluding, the positive impact of enabling a proactive flow management style at the security filter through the decision-support model is expected to be high. The research also finds that by carefully selecting the data features and model parameters, the development steps that were used for the security filter can be executed for other bottlenecks at EA. Ultimately, the decision-support model can be used to enable proactive flow management for the whole outbound passenger flow at EA in order to reduce the severity and number of bottlenecks experienced. Moreover, the research finds that similar models can be used at different airports, by taking the specific airport characteristics into account. Five recommendations for EA are found that can aid the goals.

First, the current decision-support model should be deployed and improved. User experience is required to improve the decision-support model by tuning the different parameters. This will lead to an increase in the confidence in the model's accuracy and that in turn will lead to greater use of the model (and its action proposals), thus enabling a proactive flow management. Secondly, the final forecast model that was constructed relies on a data set of 8.5 months and thus did not include a full year of data. When more data becomes available, the forecast model needs to be re-evaluated to determine if more data features can be included and the neural network needs to be retrained every month to increase its accuracy. Thirdly, to increase data availability and with that data usage, EA needs to continue its efforts in setting up an integration platform that controls the flow of data between all software systems. This will result in a decrease in its dependency on third parties and an increase in data control. Fourthly, EA needs to start the process of enabling proactive flow management by using a similar research process at other bottlenecks after the current model has been generally accepted and is actively used. Lastly, interconnecting the decision-support models is required so that the forecasts, action proposals and expected effects of taken actions can all be linked together to finally enable an all-embracing proactive flow management style. The model for example needs to make sure that actions that are proposed do not form bottlenecks at other locations in the passenger flow. Such an integrated system will ultimately have the capability to create a near perfect passenger flow at EA.


EindhovenAirport
Time: 12:04
Open crew lane
PA: Security filter for flight W61926 is closed
Extra personnel (internal)

Security filter forecast by Mike Brom

Interval start	Interval end	Pax	Planned capacity	Expected waiting time	Proposed action 1	Proposed action 2	Proposed action 3	Last taken action	Lanes	Operators	Rubric
11:00	11:15	A 129	100	12 min					2	11	F
11:15	11:30	A 109	150	16 min					3	14	I
11:30	11:45	A 133	150	16 min					3	14	I
11:45	12:00	A 137	150	17 min					3	14	I
12:00	**Now**	A **12** / E **78**	**150**	14 min	**Open crew lane**	**PA: Security filter for flight W61926 is closed**	**Extra personnel (internal)**		**3**	**14**	**I**
12:15	12:30	E 67	150	10 min					[illegible]	14	I
12:30	12:45	E 63	100	7 min					[illegible]	11	F
12:45	13:00	E 64	100	6 min					[illegible]	11	F
13:00	13:15	E 71	100	6 min					2	11	F
13:15	13:30	E 86	100	7 min					2	11	F
13:30	13:45	E 110	100	9 min					2	11	F
13:45	14:00	E 128	100	11 min	Extra personnel (external)				2	11	F
14:00	14:15	E 140	100	14 min	Open 3th & 4th lane 30 minutes early	Extra personnel (external)			2	11	F
14:15	14:30	E 146	100	18 min	Open 3th & 4th lane 15 minutes early	Extra personnel (external)			2	11	F
14:30	14:45	E 148	212	25 min	Extra personnel (external)				4	20	L
14:45	15:00	E 144	212	22 min	Extra personnel (external)				4	20	L

Figure 1: Screen-shot decision-support dashboard

Contents

List of Figures

List of Tables

List of Abbreviations

Adam	Adaptive moment estimation	IMF	Intrinsic mode function
AOM	Airport Operations Manager	IMM	Interactive multiple model
AR	Auto-regressive	KF	Kalman filter
ARIMA	AR integrated moving average	KMar	Koninklijke marechaussee
BHS	Baggage handling system	KPI	Key performance indicators
BP	Backward propagation	LOCF	Last observation carried forward
BPTT	Backpropagation through time	LS-SVM	Least squares support vector machine
DCS	Departure control systems	LSTM	Long-short term memory
DSS	Decision support systems	MA	Moving average
EA	Eindhoven Airport	MAE	Mean absolute error
ED	Encoder-decoder	MIMO	Multi-input multi-output
EI	Expected-improvement	MLP	Multi-layer perceptron
EM	Expectation-maximization	MSRBF	Multiscale radial basis function
EMD	Empirical mode decomposition	MTUNN	Multiple temporal units NN
FIDS	Flight information display system	NN	Neural network
FIFO	First come first serve	PENN	Parallel ensemble NN
FIS	Flight information system	PNL	Passenger-name list
FNN	Feedforward neural network	PRM	Passengers with restricted mobility
GBDT	Gradient boosting decision Tree	RBF	Radial basis function
GD	Gradient descent	RF	Random Forest
GRU	Gated recurrent unit	RMSE	Root mean squared error
GSVM	Grey support vector machine	RNN	Recurrent neural network
HA	Historic average	SARIMA	Seasonal ARIMA
HP	Hyper-parameter	SSBD	Self-service baggage drop-off
HRF	High risk flight	SVM	Support vector machine
IM	Information manager	VAR	Vector auto-regression

Chapter 1

Introduction

The capacity crunch at airports in which limited infrastructure capacity meets continuously expanding passenger numbers has been a subject of study for an extended period of time. Many airports around the world exhibit congestion problems at the airport passenger terminal as the number of passengers continue to increase (Hee and Zeph, 1998). Amsterdam Airport Schiphol for example identified that 'to be able to deal with the impacts of an expanding market and many other developments in time, it is necessary to evaluate critically and constantly the gearing between the current organization, the available infrastructural capacity and future needs' (Gatersleben, 1999). Numerous studies have identified the growing capacity crunch at airports all over the globe (e.g. Yanbing et al., 2007; Chiang and Taaffe, 2014; Alodhaibi et al., 2016). A study at Naples International Airport identified that the rapid increase in passenger traffic on the one hand and the slow expansion of airport capacity on the other limited the airports capability to maintain satisfactory service to the customer (Adacher et al., 2017).

Eindhoven Airport (EA), comparable in passenger numbers to Naples, foresees similar issues. This research revolves around EA and aims to identify the causes and effects of the looming capacity crunch and consequently design and implement an adequate solution.

1.1 Case setting

1.1.1 Company profile

Eindhoven Airport is the second largest airport in The Netherlands (after Amsterdam Airport Schiphol), serving over 6 million passengers per year. The airport connects the Brainport Region to over 75 destinations in Europe and the rest of the world, served by 5 different airlines. Furthermore, EA has the ambition to grow to 9 million passengers in 2025, while paying close attention to quality of life and sustainability within the region.

The airport serves both as a civilian airport (Eindhoven Airport) and as a military airbase (Airbase Eindhoven). Although the civilian part of the airport continues to grow, the runway, air traffic control and additional runway services (e.g. fire department) are still handled and owned by the Royal Netherlands Air Force. This research focuses solely on the civilian (Eindhoven Airport) part of the airport.

EA currently employs around 70 people, growing in number almost every year. The airport is able to handle millions of passengers annually with a limited number of employees due to the outsourcing of the handling activities: Viggo Eindhoven B.V. is responsible for most handling operations such as passenger check-in and aircraft pushbacks, G4S Security Services B.V. is responsible for most security operations and CSU B.V. performs all cleaning activities at the airport. Viggo, G4S and CSU are some of the key partners of EA, although there are numerous others that play an essential role. Furthermore, a number of governmental parties (e.g. Royal Netherlands Air Force) are involved. EA fulfills an orchestrating role between all those parties.

1.1.2 Stakeholders

Throughout this paper there are several mentions of the *stakeholders* of this research project. We identify two main stakeholders: the company supervisor of this project (EA's Information Manager) and the Airport Operations Managers (AOMs). During day-to-day operation the orcherstrator role as mentioned in the previous section is mainly executed by EA's 8 AOMs. An AOM is the first overall responsible person in the operation and supervises the coordination between the different parties in the airport. They are considered to be main stakeholders in this research project and are often consulted for their knowledge and experience. A more elaborate explanation of the responsibilities of the AOM's is provided in Section 3.3.

1.1.3 Scope

In its orchestrating role, EA oversees numerous activities and business areas to handle over 6 million passengers annually. Examples are aircraft handling, parking services and baggage handling. This research focuses specifically on the activities related to the outbound passenger flow (departing passengers). From a passenger perspective, the start of the outbound passenger flow is defined as the moment when the departing passenger arrives at the airport terminal and it ends when the passenger boards the aircraft.

1.2 Problem analysis

Extensive discussions with the relevant stakeholders at EA took place to identify the effects of the looming capacity crunch and identify related causes of these effects. A cause and effect diagram is provided in two-fold: Figure 1.1 depicts the causes and Figure 1.2 depicts the effects. To provide context for the analysis of the causes, the term *imperfect passenger flow* is introduced as a generic term for the identified effects. Examples of an imperfect passenger flow are intolerable long queues at the security filter or passenger congestion in pathways. Other (generalized) effects are discussed in Section 1.2.2. To illustrate, a perfect passenger flow is seen as a seamless flow for passengers, without severe bottlenecks. Establishing a seamless passenger flow is one of the cornerstones of EAs ambitions.

1.2.1 Causes

As found in relevant literature at other airports, one of the main causes of an imperfect passenger flow is the trade-off between capacity and growth; the airport is growing rapidly in passenger numbers whilst capacity is limited and mainly fixed. These two (*capacity* and *growth*) are therefore identified as main cause areas for the imperfect passenger flow. An important aspect concerning the growth cause area is the increase in peaks that the operation endures. Peaks are considered to be situations in which the airport runs at (close-to) full capacity, with a high number of aircraft and passengers at the airport. Full (or close-to full) capacity situations are most critical since there is an increased probability of bottlenecks whilst at the same time resolution is difficult: there is a reduced solution scope. The reduced solution scope during peaks is mainly a result of the limited options to upscale (e.g. all security lanes are already open) and the limited options to disperse the congestion that is a result of the bottlenecks (e.g. all areas of the terminal are full of passengers). The trade-off between capacity and growth is identified as the subsequent *capacity crunch* area in the cause and effect diagram.

At EA, the negative effects of an expanding capacity crunch are found to be amplified by inadequate passenger flow management. *Passenger flow management* is defined as all actions and steps that are taken by EA to manage the flow of passengers across the airport. This includes long-term flow management in the form of resource planning (e.g. number of open security lanes) and short-term flow management in the day-to-day operation (e.g. ad hoc staff changes due to delays). The stakeholders argue that inadequate flow management is a result of both a shortage in available data and a lack of usage of the data that is available: on the one hand there is inefficient resource planning due to the limited usage of historical data and on the other hand there exists a reactive flow management style in the day-to-day operation due to the limited usage of real-time or live data. A reactive flow management style is focused on issue

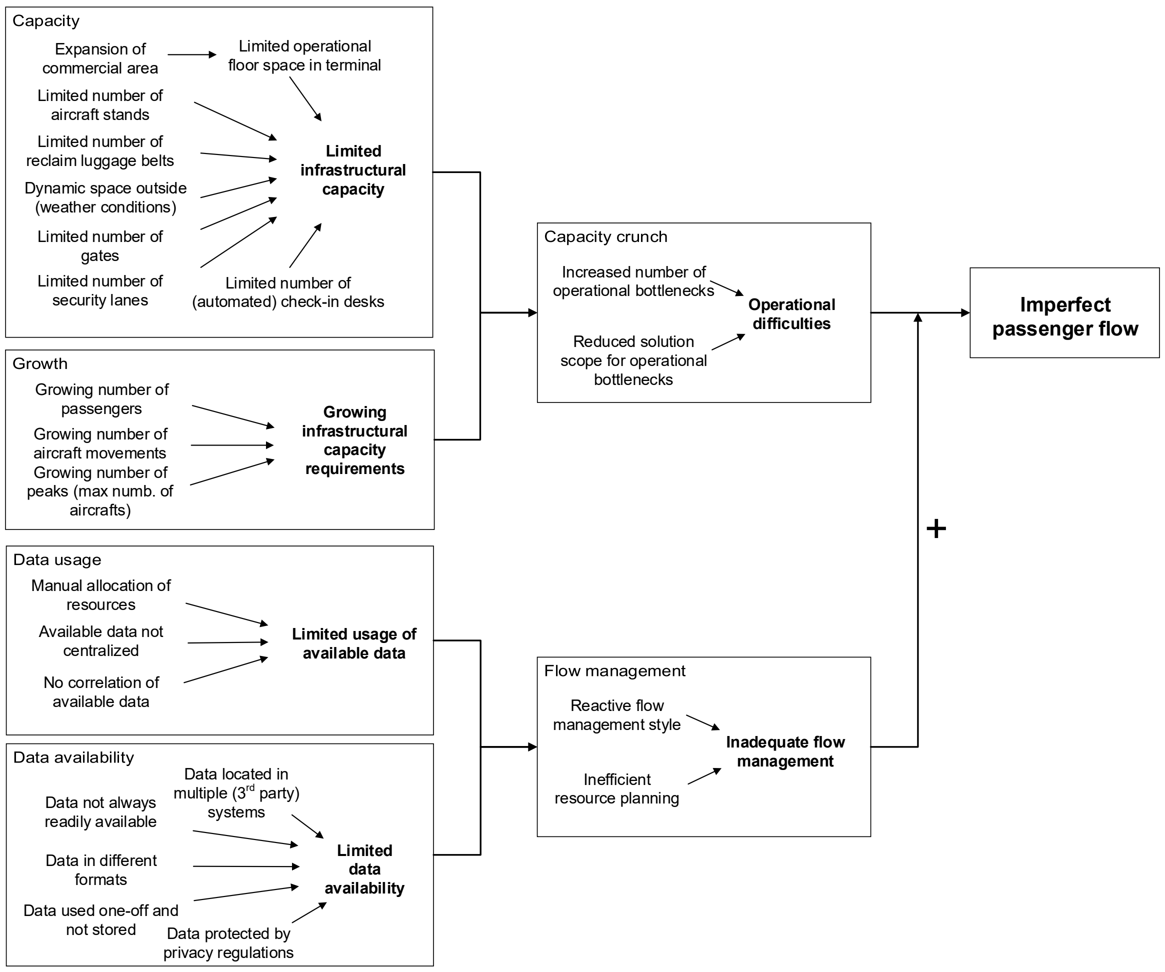

Capacity
Expansion of commercial area
Limited operational floor space in terminal
Limited number of aircraft stands
Limited number of reclaim luggage belts
Dynamic space outside (weather conditions)
Limited number of gates
Limited number of security lanes
Limited number of (automated) check-in desks
Limited infrastructural capacity
Growth
Growing number of passengers
Growing number of aircraft movements
Growing number of peaks (max numb. of aircrafts)
Growing infrastructural capacity requirements
Data usage
Manual allocation of resources
Available data not centralized
No correlation of available data
Limited usage of available data
Data availability
Data not always readily available
Data located in multiple (3rd party) systems
Data in different formats
Data used one-off and not stored
Data protected by privacy regulations
Limited data availability
Capacity crunch
Increased number of operational bottlenecks
Reduced solution scope for operational bottlenecks
Operational difficulties
Flow management
Reactive flow management style
Inefficient resource planning
Inadequate flow management
+
Imperfect passenger flow

Figure 1.1: Cause and Effect diagram: causes

resolving opposed to a more desirable proactive flow management style that is focused on issue prevention. This reactive flow management style results in problems in the outbound passenger flows that are difficult to resolve and that are already of considerable magnitude before action is taken to mitigate them. An example of proactive flow management is using real-time data on the number of online checked-in passengers for the upcoming flights to relocate staff members to the check-in counters before the queue explodes. A more elaborate example of the passenger flow and inadequate flow management is provided in Section 1.2.3. Concluding, in the cause and effect diagram, the moderator *flow management* cause area thus has two input areas: *data usage* and *data availability*. These two result in inefficient resource planning and a reactive flow management style that in turn amplify the negative effects of the looming capacity crunch.

1.2.2 Effects

The negative effects of the capacity crunch are captured in the term *imperfect passenger flow*. More specifically, the effects can be classified as being passenger and/or business related. The passenger related effects are summarized as being part of passenger dissatisfaction: long queues, limited freedom of movement and congestion in pathways result in irritation and frustration. Moreover, long queues could result in passengers missing their flight, which results in even higher passenger dissatisfaction. These different specific effects related to passenger dissatisfaction were gathered through discussions with stakeholders, but more importantly through continuous Net Promotor Score surveys at EA (Schalken and Nefkens, 2018). As an example of dissatisfaction,

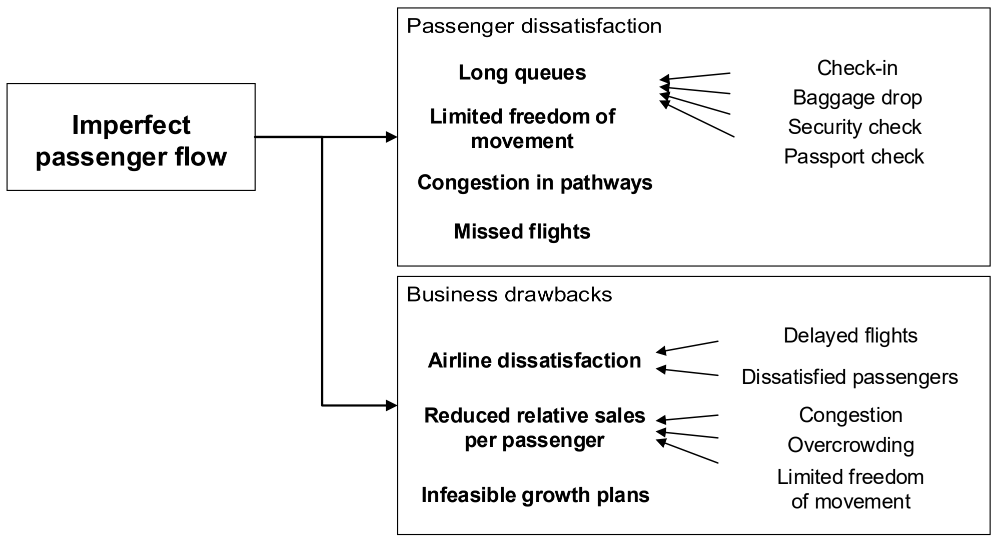


Figure 1.2: Cause and Effect diagram: effects

the surveys indicate 'waiting time at security filter' as having a significant negative effect on the NPS at the airport. Furthermore, throughout the surveys, negative associations with the airport such as *disorderly*, *busy* and *chaos* are frequently recurring.

The second set of effects concern the business drawbacks. Three negative effects are identified which are related to the business aspect of EA: airline dissatisfaction, reduced sales per passenger and unfeasible growth plans. The first two effects, airline dissatisfaction and reduced relative sales, are directly related to the airports revenue. To elaborate, the airport currently has three main sources of income: airlines, commercial areas and parking. Airline dissatisfaction strains the income from airlines and a decrease in sales per passenger is related to the commercial areas at the airport.

Two main causes for dissatisfied airlines are identified by the stakeholders. Firstly, flights can be delayed due to operational bottlenecks at EA, whilst these bottlenecks are mostly outside the airlines control. Secondly, a negative passenger experience at the airport is likely to be reflected on the airline with which the passenger travels. Airlines that are dissatisfied with the operations at EA could for example decide to discontinue flight routes, be unwilling to introduce new routes or demand reduction in fares. Furthermore, an overcrowded airport with congestion in multiple pathways and limited freedom of movement could negatively affect relative income: according to the airports commercial manager, passengers who cannot roam freely around the terminal will be less keen on shopping, resulting in lower sales per passenger.

As a last business drawback, an imperfect passenger flow could hinder growth and ultimately result in unfeasibility of current growth plans. Firstly, this could be due to operational difficulties: the airport is simply not able to accommodate more passengers (operation runs at maximum capacity). Secondly, as mentioned, if the airlines are dissatisfied with the situation at the airport they might decide for discontinuation of current routes and/or be unwilling to introduce additional routes.

1.2.3 Problem exemplification

In this section a fictional exemplification of the outbound passenger flow and related flow management is provided. This example serves as a clarification for what the outbound passenger flow and its bottlenecks entail, what flow management is and why it is currently deemed inadequate in a high demand low capacity situation by the stakeholders. A detailed explanation of the outbound passenger flow can be found in Section 4.1.1. For sake of clarity, a limited textual representation of the flow from a passenger perspective is now provided: a passenger either checks-in online or at the airport, possibly drops of baggage at the airport, goes through the security filter, possibly goes through border control and exits the terminal at one of the boarding gates and finally enters the aircraft.

It is Saturday morning 10.00, June 2 2018 and the start of the Dutch schools summer holidays. Whilst only two flights are scheduled to depart from Eindhoven Airport in the coming one and a half hours, a total of 21 flights is scheduled to depart after that between 11.30 and 14.30. Based on the maximum capacity of the aircraft, the expected number of departing passengers for the latter flights is over 4000.

At 10.15 a queue starts to form in front of the check-in desks. The two staff members that are assigned to the check-in desks are unable to process the large inflow of passengers in the queue. At 10.30 the staff members figure out that most passengers of three of the upcoming flights were not able to check-in online due to technical difficulties, thus needing to check-in at the airport. Word of this issue gets to the on-duty AOM who then requests at 10.35 to relocate staff members from the information counter to the check-in counters. The queue however is already of unsatisfactory length and it takes another 40 minutes to bring it back to an acceptable length.

At 11.15, the security filter is suddenly rapidly filling up, resulting in unacceptable queue times of over 30 minutes. Apparently, multiple buses got stuck in a traffic jam and all arrived around the same time at 10.45. Furthermore, due to the added capacity at the check-in desks, a lot of passengers are checked-in at the same time and directly proceed to the security filter afterwards. The security filter already runs at maximum capacity (6 open security lanes), so the on duty AOM decides to exploit the last mitigating solution and requests for the crew lane to be opened for passengers. The crew lane normally is only used for staff members of the airport and airline crew members. Due to the limited additional capacity that the crew lane brings, the unacceptable queue lengths persist for well over an hour.

Flight FR2576 to Málaga (Spain) is scheduled to depart at 12.15 from gate 7 and the 189 seats of the flight are completely sold-out. The gate is announced to the passengers at 11.30 and the passengers rush to the gate and stand in the queue, waiting to be allowed to go through the boarding scan gates. Since the flight is completely full and the gates queuing area only fits around 100 passengers, the remaining passengers queue up in the airport's main pathway and in front of the entrance to one of the shops. Unfortunately, the flight is delayed, meaning boarding does not start directly and the queue creates unwanted congestion at the airport. To mitigate the effect, the AOM requests part of the passengers to be let through the boarding scan gates prematurely (also called pre-boarding). These passengers then have to wait in the small space behind the scan gates or outdoors in the pathways that lead to the aircraft, which is presumably not ideal in terms of passenger satisfaction.

Three typical problems that arise in the outbound passenger flow have been illustrated (unacceptable check-in queue, unacceptable security filter queue, gate congestion), as well as the flow management that the on-duty AOM performs to mitigate these problems (staff relocation, resource purpose modification, pre-boarding). Flow management however is deemed inadequate by the stakeholders due to its reactive nature and the inefficient resource planning.

The reactive nature comes back clearly in all three examples, in which action is taken only after the problem already exists. The AOMs argue that the usage of real-time data can address this reactive flow management style: firstly pro-activity (issue prevention) could for example have been achieved by using real-time data on the number of online checked-in passengers for the upcoming flights to relocate staff members to the check-in counters before the queue explodes. Secondly, real-time forecasting of the short-term passenger numbers at the security filter can for example be used to spread passengers across time to reduce sudden spikes (e.g. announce a flight via intercom to persuade passengers to go to the security filter early). Lastly, dynamic gate announcement/assignment based on updated expected departure times can for example be used to overcome congestion in front of the gates due to delays.

The inefficient resource planning can be mainly seen in the gate congestion problem. The stakeholders argue that increased usage of historical data can enhance the current planning methods at EA. For example, from historical data, it can be found that the specific flight FR2576 to Málaga has a high average load factor (percentage of filled seats). It would have been better to assign this flight to a gate at the outskirts of the terminal (e.g. 1-2 or 11-12), since the high number of passengers on this flight would result in less congestion problems there (there are no pathways/shops). A flight with a historical low average load factor should then be assigned to gate 7.

1.2.4 Problem statement

As illustrated in the previous sections, there are multiple cause areas that result in an imperfect passenger flow when combined. The causes and effects were gathered through discussions with several stakeholders. This leads to the following problem statement for this research project:

Limited infrastructure capacity and continuously expanding passenger numbers lead to an increase in operational bottlenecks and a decrease in the associated solution scope. Combining this with the amplifying effect of an inadequate flow management style resulting from the unavailability and non-usage of relevant data leads to an increasingly imperfect passenger flow at Eindhoven Airport.

1.3 Research goals

1.3.1 Practical goal

The practical (business) goal of this research is derived directly from the problem statement and business scope. EA desires to have a perfect passenger flow with limited operational bottlenecks. For EA the value of this research therefore lies in reducing the number and severity of these bottlenecks. The practical goal is formulated as follows:

The research should provide insights on how to reduce the severity and number of bottlenecks experienced in the outbound passenger flow, that are a result of the capacity crunch and that are amplified by inadequate flow management.

1.3.2 Scientific goal

It is apparent that there are multiple research and solution directions that address the identified practical goal. As the example from 1.2.3 shows, the problems are profoundly complex and span numerous facets of the outbound passenger flow at EA. Elaborate business understanding is required to formulate a scientific goal that fits within the scope of one research project, that addresses the practical goal and is also relevant from a scientific perspective. This business understanding is established in Chapter 4 and identifies the different bottlenecks and the current state of flow management within the outbound passenger flow. Subsequently, the chapter ends with the conclusion that in order to address the practical goal, addressing the inadequate (reactive) flow management style by enabling a *proactive flow management style* is most relevant from both a scientific and business point of view. A proactive flow management style is seen as a style in which the main focus is on issue prevention so as to create a seamless passenger flow. We find that other options to address the practical goal like expanding infrastructure capacity, limiting growth or improving resource planning are either unfeasible from a business point of view or less interesting from a scientific point of view.

To enable a proactive flow management style, the stakeholders identify that they require real-time insight into the current state of the outbound passenger flow and more importantly require insight in the short-term future state of the flow. This future state can primarily be obtained by real-time forecasting of the short-term passenger flow. To illustrate, part of the real-time future state insights could for example be a short-term forecast for the number of passengers that will arrive at the check-in counters in the near future. With this information, proactive measures can be taken to ensure that no extreme queues will form. A first scientific goal thus lies in providing scientific insights in developing, using and analyzing a real-time short-term passenger flow forecasting model within an airport terminal.

Since this is very broad, an elaborate literature review is conducted in Chapter 2 that results in a more detailed explanation of the scientific contribution and relevance of this paper (Section 2.3). In short, firstly we find that the field of short-term passenger flow forecasting in the specific airport context is completely undiscovered. This research paper aims to fill that gap. Secondly, a broader literature search that also includes the field of public transport finds that the vast majority of the proposed forecasting models merely use a single source time-series and its derivatives (e.g. day of week) as input for the forecasting model. Instead, we propose a multi-source multivariate forecasting model with clear temporal links between the data sources (e.g. also use recent bus arrivals to forecast the short-term passenger flow).

Subsequently, the forecasts can be used to provide additional decision-support to the stakeholders to better aid a proactive flow management style. To illustrate, additional decision-support for example can be provided in the form of derived statistics (e.g. computing the expected queue waiting time by combining the forecast with the planned capacity) or action proposals (e.g. provide the stakeholders with a range of possible mitigating actions for upcoming bottlenecks). Since this additional decision-support could result in a complete research project on its own when fully addressed scientifically, it will primarily be addressed from a business point of view by using expert knowledge. Nonetheless, the scientific contribution lies in the lessons that are learned during development of a decision-support model that aims to enable a proactive flow management style.

Concluding, the practical goal can be addressed by enabling proactive flow management through short-term passenger flow forecasting and additional decision-support. This eventually leads to a scientific goal:

Provide scientific insights in the usage of real-time short-term passenger flow forecasting in an airport terminal environment and provide more general insights in additional decision-support that collectively enable a proactive flow management style.

1.4 Research questions

Based on the above discussion a set of research questions is defined. The research questions are in line with both the practical and scientific goal of this research. First, to guide this research, a main research question is defined:

How can proactive flow management be enabled by using real-time short-term passenger flow forecasting and providing additional decision-support in order to reduce the severity and number of bottlenecks experienced in an airport terminal environment?

As indicated, this research focuses specifically on the outbound passenger flow (departing passengers). The resulting scope when applying the main research question on the whole outbound passenger flow is too large for a single research project. We argue in Section 4.3.2 that solely resolving the research question for EA's security filter is adequate to accomplish the objectives that are set. We find that the security filter is the most critical bottleneck within the outbound passenger flow, causing both the highest number and most severe flow problems. Moreover, we believe that the findings that will be obtained when developing the necessary models to enable proactive flow management at the security filter can almost directly be used for most of the other identified bottlenecks of Section 4.2. This scoping step results in an adjusted main research question:

How can proactive flow management be supported for an airport's security filter by using real-time short-term passenger flow forecasting and providing additional decision-support in order to reduce the severity and number of bottlenecks experienced?

In support of the main research question, 5 sub-questions are identified.

1. What is the current situation regarding bottlenecks and flow management within the context of the outbound passenger flow at Eindhoven Airport?
2. What is the current situation regarding (real-time) data within the context of the outbound passenger flow at Eindhoven Airport?
3. How to design a model to forecast the short-term passenger flow at Eindhoven Airport's security filter in real-time?
4. How to design a model that uses real-time data and short-term forecasts to provide decision-support and enable proactive flow management at Eindhoven Airport's security filter?
5. What is the expected impact of implementing a real-time data driven proactive flow management style?

To complement these research questions, Figure 1.3 is introduced to provide the reader with an outlook of the final deliverable of this research project. Further explanation of the choices behind this final deliverable is provided in Section 4.3.1.

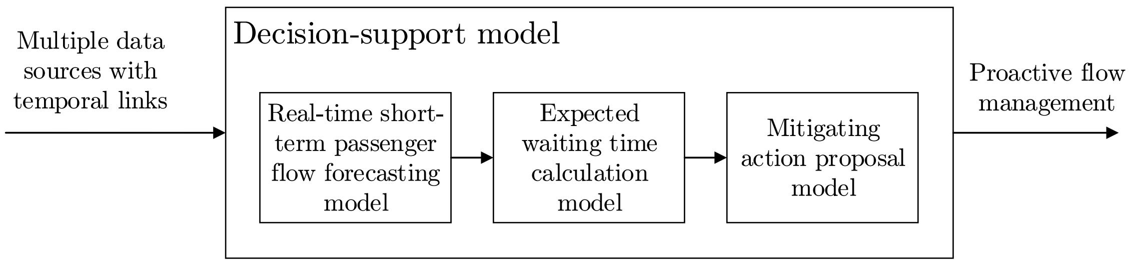

Multiple data sources with temporal links
Decision-support model
Real-time short-term passenger flow forecasting model
Expected waiting time calculation model
Mitigating action proposal model
Proactive flow management

Figure 1.3: Final deliverable: decision-support model

1.5 Outline

The structure of the thesis is now introduced. In Chapter 2 a literature study is performed that focuses on the methods for short-term passenger flow forecasting that can be found throughout literature. This study is used to choose the most appropriate forecasting method for this thesis, after which an in-depth introduction is provided to this method. The chapter ends with a more elaborate explanation of the scientific contribution of this paper and a deepening of third sub-question. Chapter 3 introduces the methodology that is used to structure the research. The rest of this research follows the resulting structure and is primarily based on the problem solving cycle (van Aken et al., 2012) and the CRISP-DM framework (IBM, 2011). Chapter 4 deepens the business understanding of Chapter 1 and this subsequently leads to answering the first sub-question. Chapter 5 answers the second sub-question and focuses on data understanding. This chapter is concerned with identifying which data exists at EA, where the data is located and identifying the quality and availability of the data. In Chapter 6 (data preparation) and Chapter 7 (forecast modeling) the short-term passenger flow forecast model is developed, collectively answering the third sub-question. Chapter 8 is concerned with constructing the model that uses real-time data and short-term forecasts to provide decision-support and enable proactive flow management, thus answering the fourth sub-question. The fifth sub-question is answered in Chapter 9 by evaluating the constructed models by means of a pilot-phase and a consecutive quantitative and qualitative evaluation. Finally, the conclusions, future research and recommendations are provided in Chapter 10.

Chapter 2

Literature review

This chapter first discusses relevant literature regarding short-term passenger flow forecasting within airport terminals. Specifically, the literature review tries to identify the methods that are used throughout the literature to forecast the short-term passenger flow. Secondly, the constructed list of methods is used to determine the most appropriate forecasting model for this research project. Following, the chosen method is explained in more detail and finally the scientific contribution of our proposed forecasting model is clarified.

2.1 Short-term passenger flow forecasting literature

For the course of this research project, we deem a forecast to be short-term if it has a time window between 0-3 hours. An extensive literature review specifically aimed at the airport context did not yield any relevant results. Apparently, the topic of real-time passenger flow forecasting within an airport context is unexplored in scientific literature. We argue that *public transport* can serve as a valuable research context alternative, with sufficient research readily available.

Public transport are buses, trains, and other forms of transport that are available to the public, charge set fares, and run on fixed routes (Oxford University Press, 2018). Airlines are part of the public transport network. Although there are some essential differences between airlines and other modes of transport, they share a lot of characteristics. Differences can be found for example in the timeliness of ticket purchases (in advance vs. on the spot) or security requirements, whilst similarities can be found in that airlines are also available to the public, charge set fares and run on fixed routes. Subsequently, e.g. bus, train, tram and metro terminals share a lot of characteristics with airport terminals. We therefore argue that literature that concerns short-term passenger flow forecasting in the broader context of public transport is relevant in this literature review.

In order to provide context, the relevant forecasting methods are categorized into four groups: naïve, parametric, non-parametric and hybrid (van Lint and van Hinsbergen, 2012). This categorization is used by the authors for short-term traffic forecasting. However, the same categorization can be used in this research project since short-term passenger flow forecasting is deemed to be a subcategory of short-term traffic forecasting (Ni et al., 2017). Scientific literature related to traffic forecasting is excluded in this literature review since we deem it to be considerably different from literature related to passenger flow forecasting. Differences can for example be found in speed of movement (high vs. low) and arrival flow capacity (restricted vs. unrestricted).

No naïve approaches have been found in the literature review. Naïve approaches do not deduct a forecasting model structure or parameter set from (real-time) data, but rather use direct relationships (e.g. historical average) as a predictor for the upcoming time periods. Although computational requirements are low and these methods are easy to implement, they are not suitable in most forecasting settings due to their low accuracy and are therefore not found in the literature and thus excluded from this review. Furthermore, only limited literature can be found in the application of parametric models within the short-term passenger flow forecasting field. As indicated, short-term passenger flow forecasting is a subcategory of short-term traffic forecasting. The latter has been studied for an extensive amount of time, whilst the passenger flow field spun off later and has only emerged in recent years. This field therefore skipped the basic approaches and almost directly started with the more promising and accurate approaches (non-parametric and hybrid).

2.1.1 Parametric models

Parametric approaches rely on analytic models. The parameters of the model are found through (real-time) data whilst the structure of the model is predetermined and is based on expert knowledge and theoretical considerations. As explained, a limited number of only two papers has been found that employ parametric models.

Exponential smoothing

In his attempt to create a fuzzy clustering approach to real-time demand-responsive bus dispatching control, Sheu (2005) introduced a simple passenger demand forecasting method. The method is an exponential smoothing technique that relies on real-time information regarding passenger number changes. The passenger demand for the upcoming time interval is calculated by multiplying the passenger number change in the last two time intervals with a predetermined factor α. In this case α is determined by searching for the optimal value using historical data with the objective of minimizing the sum of squared forecasting errors.

Sugiyama et al. (2010) introduced two possible adaptions to this exponential smoothing method. The first adaption is an auto-regressive method (AR) that forecasts the passenger flow in the coming time interval by combining real-time information of passenger flow in the previous time interval together with historical reference data on passenger flow on the same time/day of the week. The second adaption is a pattern matching (PM) method that uses the same real-time information but searches for a comparable flow pattern in previous data and forecasts the passenger flow based on the most similar data. They conclude that a combination of the two, based on the time of day, is most accurate.

2.1.2 Non-parametric and hybrid models

Non-parametric approaches are comparable but different from parametric approaches in that the number and nature of the parameters is flexible rather than fixed in advance. Furthermore, hybrid approaches combine two or three of the proposed categories (naïve, parametric and non-parametric) in order to obtain the forecasts. Eight different non-parametric and hybrid model categories are found in the relevant literature.

Feedforward neural networks

Feedforward neural networks (FNN) are artificial neural networks wherein connections between the nodes do not form a cycle. Nodes are arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers have no connection with the external world, and are therefore called hidden layers. Each node in one layer is connected to every node on the next layer, resulting in information being *fed forward* from one layer to the next. There is no connection among nodes in the same layer.

In order to improve revenue management in railway operations, Tsai et al. (2009), introduce a feedforward neural network approach for short-term passenger flow forecasting. In their paper they examine three different network structures: conventional multi-layer perceptron (MLP), multiple temporal units neural network (MTUNN) and parallel ensemble neural network (PENN). Tsai et al. (2009) first extract five data features from their Taiwan railway data set that are used in the input layer(s): daily trend, day-of-the-week pattern, vacation day, monthly trend and month-of-the-year pattern. The fully forward connected MLP structure has only three layers: input, hidden and output. The input layer consists of the extracted data features.

The single hidden layer combines information from the input layer and generates new features for network learning, thus can be seen as a feature extractor. Finally, the output layer generates forecasts and propagates errors for parameter estimation.

The MTUNN structure differs from MLP in that the hidden layer does not mix all (five) input features but rather combines them within groups. Tsai et al. (2009) define two feature groups: daily and weekly. Also, another layer (called the integrating layer) is added. This layer combines the two outcomes from the hidden layer in order to finally produce the results in the output layer. The last model in their paper follows the PENN structure. This method combines the MLP and MTUNN approach. Similarly to MTUNN, it splits the data features into two groups (daily and weekly). However, these groups are not only split within the hidden layer, but across all layers. PENN, in fact, contains two individual models: a daily model and weekly model. Both models are formulated according to the MLP structure (with three layers). The outputs of the two models are combined to obtain a final forecast. In this case the study simply multiplies the network outputs from the daily model and the weekly model to obtain the forecasts. They finally conclude that there is no significant difference between MTUNN and PENN in terms of forecasting performance. However, PENN is preferred due to better model parsimony.

Following on this research, Wei and Chen (2012) combine the use of empirical mode decomposition (EMD) with an MLP network to forecast the short-term passenger flow in metro systems. Their method is divided into three stages. First the short-term passenger flow series is decomposed into a number of intrinsic mode function (IMF) components (EMD Stage). The second stage (Component Identification Stage) identifies the meaningful IMFs as inputs for MLP. Finally, in the third stage, the MLP is applied to perform the passenger flow forecasting. The first two stages are comparable to the data feature extraction in Tsai et al. (2009). The EMD Stage decomposes the original passenger flow series data into a number of IMFs. The extracted IMF components represent a range of frequencies within the data (high to low) and reveal various periodic patterns of passenger flow. Each IMF component can represent the local characteristic time scale by itself. An example of an IMF component is one that represents patterns during the day. In stage two the IMFs are filtered using correlation analysis. The highly correlated IMFs are used as individual inputs in stage three (neural network) whilst the lower correlated IMFs are used as aggregated inputs. An example of a low correlated IMF component is a frequency with low oscillation that represents multiple years. Stage three produces the passenger flow forecasts by applying an MLP neural network with three layers (input, hidden and output). They finally conclude that the proposed EMD-MLP approach outperforms other models such as sole MLP and Seasonal Auto-regressive Integrated Moving Average (SARIMA).

A sole MLP has been studied more extensively by Jin et al. (2013). They aim to forecast the short-term passenger flow in urban rail transit stations in Beijing. Their method uses passenger flow data on the last three consecutive periods as input data in order to forecast passenger flow in the upcoming period. They conclude that their method meets their forecast requirements. However, they do not compare the method to others. As indicated previously by Wei and Chen (2012), a sole MLP network can be outperformed by using IMF components as input variables instead of basic passenger flow data.

Another approach was introduced by Teng and Chen (2015) whom integrate the auto-regressive integrated moving average (ARIMA) method and an MLP neural network model. They consider their bus passenger flow data series to be composed of a linear auto-correlation structure and a nonlinear structure. ARIMA is applied to the linear component whilst the neural network model is applied to the nonlinear component. By combining the outputs of both models, a forecast can be made that considers both linear and nonlinear characteristics. The authors compare their method to a sole ARIMA model and a sole MLP model and conclude that the proposed method has higher relative forecasting accuracy.

As a last feedforward neural network, Li et al. (2017) proposed a method based on multiscale radial basis function (MSRBF) networks in order to forecast passenger flows under special events scenarios. These special events may have a disruptive impact on public transportation systems and result in irregular fluctuation of subway passenger flows. The proposed method is centered around radial basis functions (RBFs). RBFs are means to approximate multivariable (also called multivariate) functions by linear combinations of terms based on a single univariate function (the radial basis function). The method is compared method to conventional techniques such as regression trees and support vector machines and conclude that their method outperforms those techniques.

Most recently, the upcoming field of deep learning methods expanded to short-term passenger flow forecasting. An MLP neural network containing multiple hidden layers is considered to be a deep learning structure. Based on the big data of rail transit in Shanghai, Zhu et al. (2018) use different dynamic factors as input vector: weather data, atmospheric temperature data, holiday and festival data, ground index data, and elevated road data. They vary the number of hidden layers between 4 and 13. They aim to predict the entrance and exit passenger flow of individual rail transit stations. By means of experiment, the authors claim that the training error and prediction error and accuracy of the model are higher than those of currently known prediction methods and its prediction error is lower than 4.1%. However, they only depict the numbers which are used to compare deep learning to multiple linear regression.

Recurrent neural networks

More recently, Toque et al. (2017) introduced a long-short term memory (LSTM) model. Such an LSTM model is a type of recurrent neural network (RNN). Unlike standard feedforward neural networks like the MLP structure, RNNs can use their internal state (memory) to process sequences of inputs. They are networks with loops in them, allowing information regarding previous data and forecasts to persist. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Toque et al. (2017) use time slice windows of the passenger counts at each station and exogenous features including the calendar information as input for the network. They compare the LSTM model to two different Random Forest (RF) techniques. They conclude that both RF methods slightly outperform the LSTM model, but attribute this to the fact that the performance levels were obtained with one multivariate LSTM model that simultaneously predicts the passengers entering/boarding at all tram/bus stops, whilst for the RF models there are as many models as there are stations or stops.

From the same authors, a follow-up study introduces a gated recurrent unit (GRU) network structure (Toque et al., 2018). This type of RNN is closely related to LSTM, as it also has gating units that modulate the flow of information inside the unit. The difference is that GRU does not have separate memory cells. The performance of both types of RNNs is comparable, but a GRU network is deemed computationally more efficient due to it's less complex structure (Chung et al., 2014). The effectiveness of GRU in short-term passenger flow forecasting is demonstrated in Toque et al. (2018). They conclude that GRU models are less sensitive to error propagation along the different time steps, contrary to other models such as vector autoregressive or random forest.

Ji and Hou (2018) use a similar GRU network architecture to forecast the short-term number of passengers in different bus station in Shenzhen, China. They develop ARIMA models and GRU neural networks for different intervals, including the 30-minute, 1-hour, 2-hour and 4-hour intervals. They conclude that the GRU models outperform the ARIMA models for all intervals.

Interactive multiple model

In their paper Ma et al. (2014) make use of an Interactive Multiple Model-based Pattern Hybrid (IMM-PH) approach to predict short-term passenger demand on bus routes in Jinan, China.

The approach dynamically combines different pattern models using an IMM algorithm. Pattern models capture relevant patterns in their respective time series. Based on correlation analysis, three of those relevant temporal time series are generated, namely, a weekly, daily and hourly pattern time series. Based on real-time data, the IMM algorithm then determines the most appropriate combination of pattern models that can be used to predict the passenger demand in the next time interval. The proposed IMM-PH model is validated by comparing with statistical methods and an artificial neural network based hybrid model. Ma et al. (2014) conclude that the results suggest that the IMM-PH model provides a better forecast performance than its alternatives, including prediction accuracy, robustness, explanatory power and model complexity.

Following on this research, Xue et al. (2015) aspire to improve the model proposed in Ma et al. (2014) by including seasonal fluctuations and volatility of time series data. They argue that the lack of fluctuations and volatility characteristics leads to serious prediction lag in the previous model. Similarly, they also include three time series models. However, they use a shorter time interval (15 minutes vs hourly) for the third time series model. According to their analysis, the adaptions to the previous model make their method more accurate.

Kalman filter

A Kalman filter is an algorithm that produces estimates of unknown variables by combining multiple sources of data and estimating a joint probability distribution over the variables of each time frame. The filter is designed to cope with statistical noise and other inaccuracies in order to improve forecasts.

Gong et al. (2014) introduced a short-term passenger flow prediction framework that mainly relies on a Kalman filter. The framework consists of three sequential stages. In the first stage, based on historical boarding count data, an initial forecast is made using an ARIMA model. Since the paper also covers passengers who depart at a bus stop, the second stage is an event-based prediction of departing passengers. Finally, in stage three, a Kalman filter-based algorithm is developed to combine the information of the first two stages and finally predict the passenger count. The authors compare the proposed three stage method to a simple direct-addition method and conclude that their method is more accurate in forecasting short-term passenger flow.

Also based on Kalman filters, Jiao et al. (2016) propose three different methods for real-time prediction of the passenger flow in Beijing, China. Their first method introduces the historical prediction error into the measurement equation and formulates a revised Kalman filtering model based on the Error Correction Coefficient (KF-ECC). The second method uses the deviation between real-time passenger flow and corresponding historical data as a state variable (called Historical Deviation) and is named the KF-HD method. Their final method includes Non-parametric Regression and Bayesian Combination within a revised Kalman filter (KF-BCNR). This method combines the prediction outcomes of a basic Kalman filter (KF) model and a nonparametric regression (NR) model by using a weighted average Bayesian method. The K-nearest neighbor nonparametric regression (KNNNR) method is employed as NR model. In the Bayesian combination framework, the weights of the KF and NR model are adjusted dynamically according to the forecasting errors of the two single models.

According to Jiao et al. (2016) the reported prediction results show that KF-ECC improves the applicability to historical trend, KF-HD achieves excellent accuracy and stability, and KF-BCNR yields the best performances. According to the authors, all three revised models are accurate and stable enough for on-line predictions, especially during the peak periods.

Support vector machine

Support vector machine (SVM) is a supervised machine learning algorithm that analyzes data for classification and regression analysis. It looks at data and sorts it into one of two categories.

An SVM outputs a map of the sorted data with the margins between the two as far apart as possible.

As indicated, Wei and Chen (2012), use IMFs as input for their MLP neural networks. A similar approach was proposed by Jiang et al. (2014) in which they use the derived IMFs as input in a grey support vector machine (GSVM). GSVMs are an extension of standard SVMs and are supervised learning models belonging to the machine learning research field. These models are able to accurately forecast time series data when the underlying system processes are typically nonlinear, nonstationary and not defined a priori. They compare their method to standard SVMs and ARIMA methods and conclude that their mean absolute percentage errors are a lot lower.

Ma et al. (2015) instead use a Least Squares Support Vector Machine (LS-SVM) in order to forecast short-term bus passenger flow in Huhhot, China. LS-SVM is different from classic SVM in that it finds the solution by solving a set of linear equations instead of a convex quadratic programming problem. Training the model to solve these linear equations and determine the correct internal parameters results in a set of equations which can be used for prediction.

In the same year Sun et al. (2015) expand LS-SVM with Wavelet theory in order to predict short-term passenger flow in the Beijing subway system. They define three stages for this prediction model. In the first stage the original passenger flow data is regarded as a signal sequence and is decomposed into different high frequency and low frequency series by wavelet. This convolves a signal (passenger flow sequence) with a predefined wavelet to decompose a signal. The choice of the predefined wavelet is dependent on the type of data that is used as input. In this case, Sun et al. (2015) use Daubechies orthogonal wavelets as the predefined wavelets. In the second stage, the least squares support vector machines (LS-SVMs) are built to predict the target high frequency and low frequency sequences according to the decomposed sequences from the first stage. In the third and last stage, the decomposition of stage 1 is reversed by reconstructing the predicted sequences and finally obtaining the short-term passenger flow forecast. The authors conclude that the proposed Wavelet-SVM method has better forecasting performance compared to state-of-the-art prediction algorithms (Wavelet-NN and EMD-BPN).

Decision tree

Decision (or probability) trees are a common and broadly used in prediction. Decision trees build classifications or regression models in the form of a tree structure. Leng et al. (2013) use a probability tree based passenger flow model that is based on historical flow information in order to predict passenger flow in the Beijing subway. The results of their experiments show that the proposed method has lower performance than existing prediction approaches.

Following on this, in order to include the influence of external and internal factors on the prediction of short-term passenger flow, Ding et al. (2016) propose to use a Gradient Boosting Decision Tree (GBDT) when predicting short-term passenger flow in the Beijing subway system. Specifically, the authors consider the access trips generated from adjacent bus stops (passengers arriving by bus at the subway station) in short-term subway ridership forecasting. A GDBT is able to capture the subtle and sudden changes of short-term subway ridership based on a series of influential factors. These factors thus include the adjacent bus stop activity and furthermore include the passenger demand in previous time intervals, the time of day, the day of the week and the day of the month characteristics. GBDT is different from classic decision trees in that it can iteratively construct and combine several simple tree models to achieve optimized prediction performance while interpreting model results by identifying the key explanatory variables. The authors find that their proposed method outperforms back propagation neural networks, support vector machines and random forests.

Box-Jenkins ARMA

Anvari et al. (2016) propose a forecasting framework for public transportation systems that is based on Box-Jenkins method. Their method uses a three-stage modeling approach. Firstly they identify the appropriate model and decide on auto-regressive (AR) and moving average (MA) components of the model. As a second step computation algorithms are used for parameter estimation of the AR and MA coefficients in the model. Lastly the model is tested for its accuracy and good fitness. In basic Box-Jenkins methods these steps are executed in sequence by the researcher. However, Anvari et al. (2016) propose a method for iterative automated execution of these steps. The authors indicate that their method outperforms several other methods such as Holt-Winters, sine wave, simple AR and linear regression.

Bayesian network

Another recent method in short-term passenger prediction is the use of dynamic Bayesian networks. Bayesian networks rely on probability distributions and use the laws of probability for prediction. In their paper, Roos et al. (2017) use a Dynamic Bayesian network that is designed to perform even in the case of incomplete data. In the latter case, an Expectation-Maximization (EM) algorithm is used to learn both the structure and the parameters of the model to fill the gap. In order to extract a real-time passenger flow forecast out of the Bayesian network, a so called bootstrap filter is applied. The authors conclude that their method outperforms the simple historical average and last observation carried forward (LOCF) methods.

2.1.3 Discussion

In the previous section the literature that has been proposed to forecast the short-term passenger flow in a public transport context has been highlighted. The constructed list of methods is used to determine the forecasting model that will be employed throughout this research project within an airport context. We argue that a recurrent neural network, and more specifically a gated recurrent unit RNN, is an appropriate choice.

We consider the parametric exponential smoothing methods to be too restrictive and simple to be used in the complex airport environment. Moreover, neural networks are by far the most commonly used methods in short-term passenger flow forecasting in the public transport context, indicating that neural networks provide high applicability in different settings. Most of the listed methods use a single input source (the historical passenger flow) and extract features from that input source (e.g. time-of-day or day-of-the-week). However, an elaborate list of data features originating from different data sources will be used in this research project. The relationships between these inputs are expected to be non-linear and complex. Neural networks have the ability to learn and model these non-linear and complex relationships. Therefore, we consider neural networks to be the appropriate choice to obtain a passenger flow forecast within the airport context.

Two different classes of neural networks have been found in the public transport literature: feedforward neural networks and recurrent neural networks. Both network structures are used in time-series prediction. However, RNN's ability to allow information to persist over time and recognize patterns in sequences of data is seen as an advantage for the course of this research project. Also, RNNs are able to produce multi-step ahead forecasts and use the predicted value of one time step to determine the value in the next time step. Feedforward neural networks do not normally have this property and rely on independent value prediction for each time-step. Independent value prediction methods have more difficulty in learning the true model and do not smooth out the effect of noise unlike multi-step prediction methods (Cheng et al., 2006). Therefore, RNNs are preferred over FNNs in this research project.

As indicated, there are two main types of RNNs: LSTMs and GRUs. The less complex

GRU structure has proven to outperform or be on par with LSTM by respectively Jozefowicz and Zaremba (2015) and Chung et al. (2014). This leads to the preliminary conclusion that a GRU RNN is the appropriate forecast model structure for this research project. However, in order to be thorough, the results of the GRU model will also be offset to an LSTM model.

2.2 Gated recurrent unit literature

In order to understand the ideas behind GRUs, a short introduction to artificial neural networks, RNNs and LSTMs is provided. Subsequently, a detailed explanation of GRUs and its model architecture will be provided after which the position of this research in literature is explained.

2.2.1 Neural networks

Artificial neural networks are inspired on the human brain. It simulates the network of neurons that make up a brain so that the computer will be able to learn things and make decisions in a way that is similar to humans. The neurons in an artificial neural network perform mathematical processing to transform input data into useful output data. The neurons are grouped into different layers. The first layer is called the input layer and is the layer that receives various forms of information from the outside world. From the input layer, this information flows through one or more hidden layers, during which these hidden layers try to transform the input into something the final output later can use. The basic mathematical processing that individual neurons in the layers perform is depicted in Figure 2.1. A neuron receives a set of inputs, multiplies these inputs with a set of weights, sums these multiplications together with a bias, and finally applies an activation function on the sum to produce a single non-linear output.

The feedforward neural network with one hidden layer is the traditional and simplest type of neural network. In an FNN the neurons of a layer are connected to all the neurons in the next layer. The output of all neurons is calculated when the information moves from the input to the output layer. A basic representation can be found in Figure 2.2.

The neural network has to be trained so that it can learn the value of the weights and biases in the model. During training, the network is fed with matching input and output data. The network processes the input records one at a time and learns by comparing the network output with the known actual output. The errors from the initial output of the first record is fed back into the network, and used to modify the networks parameters the second time around, and so on for many iterations. The errors are obtained through a *loss function*, which measures the distance between network output and the desired output. In regression problems the Root

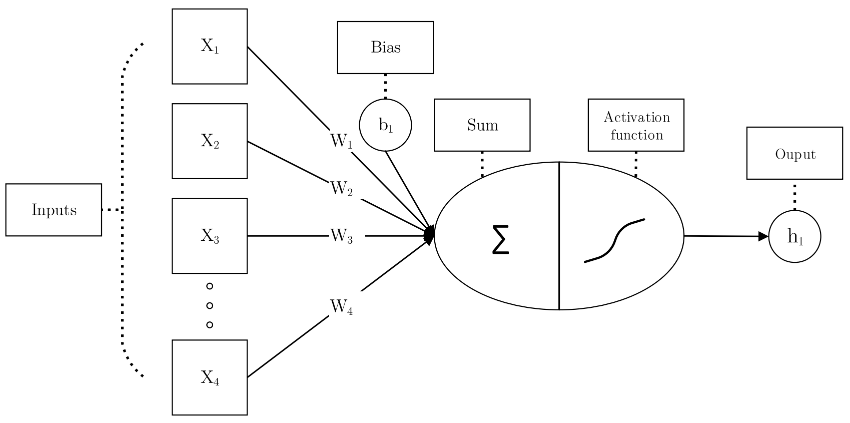


Figure 2.1: Basic neuron layout

Mean Squared Error (RMSE) and Mean Absolute Error (MAE) are well known loss functions.

The optimization technique that uses the outcome of the loss function after an iteration and improves the networks parameters is called *gradient descent*. The goal of gradient descent is to minimize the chosen loss function. It does this by calculating gradients, which are measures of how much the loss function changes when the model parameters are slightly altered. The higher the gradient, the steeper the slope and the faster a model can learn. The gradients are usually calculated through the *backward propagation* algorithm (BP), which is essentially an algorithm used to calculate derivatives quickly (Nielsen, 2015). These calculations take place after every *epoch*, which is defined as one forward pass and one backward pass of all the training examples. After each epoch the model parameters are updated and should become more accurate and thus decrease the loss function. However, due to computational restrictions, the data set is often too large to pass through completely and calculate the gradients over the entire set. In these cases the data is divided into *batches*. The gradients are calculated during the passing of every batch and the model parameters are updated accordingly.

The number of epochs and the batch size are *hyper-parameters*. Hyper-parameters are set by the researcher and are often found through elaborate experimentation. Other examples of hyper-parameters are the *learning rate* and *dropout*. The learning rate tells the optimizer how far to move the weights in the direction of the gradient whilst the dropout is a fixed percentage of connections between neurons that are removed randomly in each training epoch to prevent overfitting (Srivastava et al., 2014). Since it makes a huge impact on the learned model, choosing appropriate hyper-parameters plays a crucial role in the success of a neural network architecture.

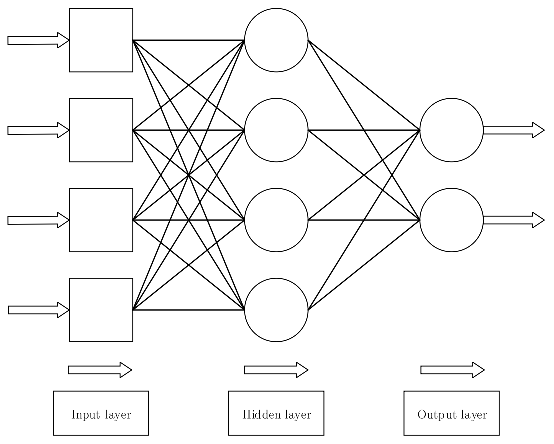


Figure 2.2: Feedforward neural network

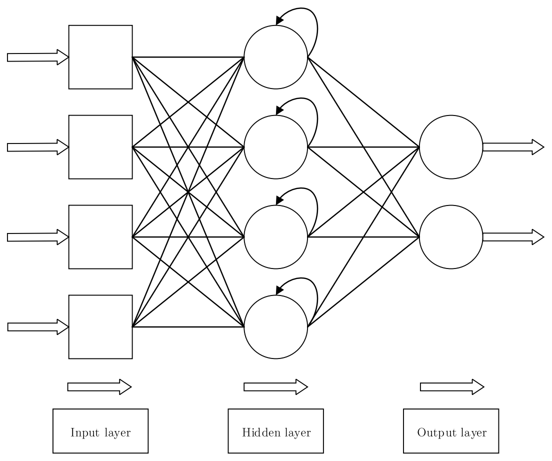


Figure 2.3: Recurrent neural network

2.2.2 Recurrent neural networks

Before diving into the structure of RNNs it is important to understand the shortcoming's of traditional NNs and the following need for RNNs when processing sequential data. When processing sequential data, the ordering of the data samples matters. A regularly used example of sequential data is that of natural language: the order of the words in a sentence matters in order to understand the context. However, traditional NNs assume independence between data samples. When using a traditional NN to predict the next word in a sentence, the network would have a hard time grasping the dependencies of the previous words because it treats each data sample individually. One could use a fixed sliding window in order to use a set of the preceding words as input, but as demonstrated in Frank et al. (2001), finding the optimal windows size is vital but often particularly challenging and sometimes impossible. Furthermore, traditional NNs do not support dynamic sliding windows, which are often required when processing sequential data (e.g. varying sentence length).

Recurrent networks address these shortcomings. They take as their input not just the current input data, but also what has been perceived previously in time. They maintain a hidden state vector that acts as memory and preserves information about the sequence. The hidden state memorizes long-term dependencies between events separated by many moments. This allows the RNN to use both current input and past information when making predictions. They are designed to recognize a data's sequential characteristics and use patterns to predict the next sequence(s).

A basic RNN architecture is depicted in Figure 2.3. The obvious difference with FNNs is the feedback loop that connects the neurons in the hidden layer to their past decisions, ingesting their own outputs moment after moment as input. Because this feedback loop occurs at every time step, each hidden state contains traces not only of the previous hidden state, but also of all those that precede it. To train the model, RNNs rely on an adaption of the backpropagation algorithm called *backpropagation through time* (BPTT). BPTT works by unrolling all input time-steps. Each time-step has one input time-step, one copy of the network, and one output. The loss is calculated for each time-step and then accumulated. The network is rolled back up and the weights are updated accordingly. The chain-like structure of an unrolled recurrent neural network is shown in Figure 2.4.

However, classical RNNs are known to experience the vanishing gradient problem, which was first mentioned in Hochreiter (1991). The problem is caused by the feedback loops inside the hidden layers and limits the capability of classic RNNs to learn dependencies across long intervals. The values within the loops are multiplied by a certain weight factor once for every time-step. Due to the many time-steps and thus stages of multiplication, whenever the factor is smaller than one, the gradient becomes extremely small and thus vanishes. The opposite, exploding gradient problem, is less common but occurs when the weight factor is bigger than one or smaller than minus one. Six years after discovery of the problem, Hochreiter came up with a solution in the form of an LSTM network (Hochreiter and Urgen Schmidhuber, 1997).

2.2.3 Long short-term memory

Since introduction, Long Short-Term Memory networks have evolved considerably and are now the most popular type of RNN. LSTMs are better able to learn long term dependencies over substantial long time intervals without experiencing the vanishing or exploding gradient problem. Several architecture variations have been proposed throughout the years, but through large-scale analysis of eight LSTM variants, Greff et al. (2017) show that none of the variants

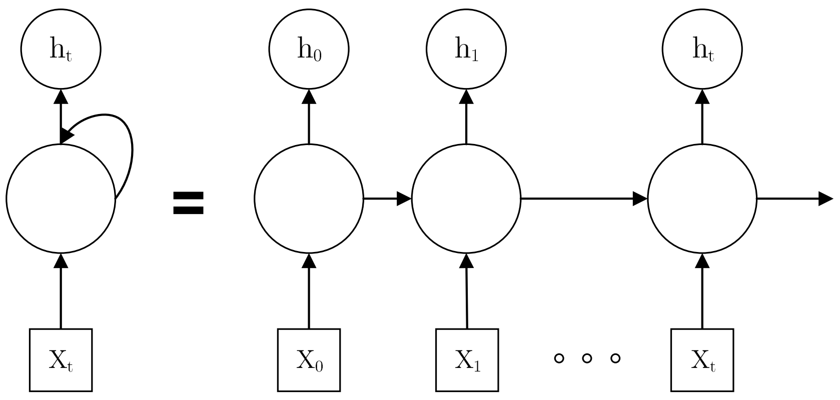


Figure 2.4: Unrolled RNN layout

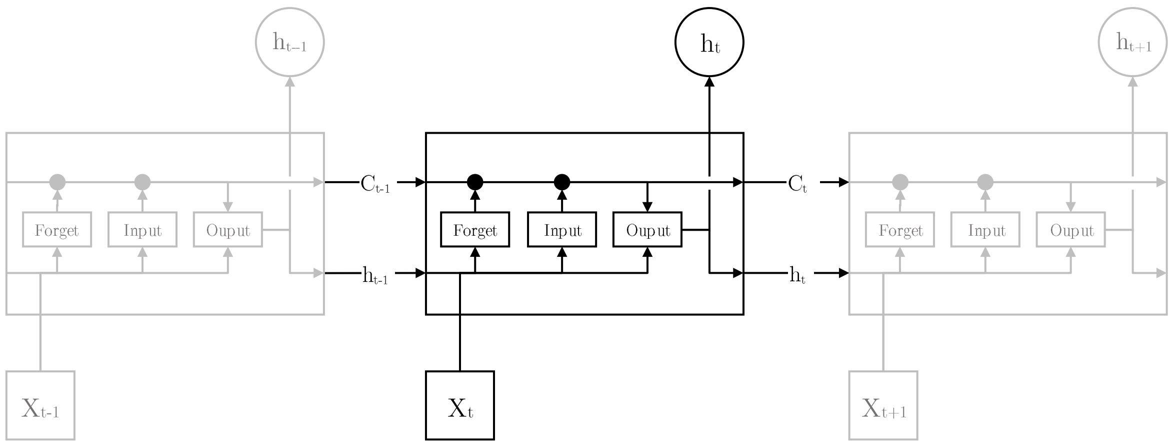


Figure 2.5: LSTM unit

can improve upon the standard LSTM architecture significantly. This section therefore focuses on the standard LSTM architecture as of Gers et al. (2000).

To obtain an LSTM architecture, the neuron and it's feedback loop in the classic RNN are replaced with an LSTM unit. An LSTM unit is a gated cell that contains information outside the flow of the RNN. Much like data in a computers memory, information can be stored in, written to, or read from an LSTM unit. The memory of an LSTM is the cell state. The LSTM unit decides what to store and when to allow reads, writes or deletions on the cell state via several gates that open and close. Those gates act on the signals they receive and block or pass information based on the signal's strength. The weights associated with the gates are adjusted and optimized during the BPPT learning process, so that the network learns when to allow the reading, writing or deletion of information. In support of the simplified representation of an LSTM unit provided in Figure 2.5, a step-by-step walk through of an LSTM unit is provided.

The first layer in a unit is called the *forget layer*. The forget layer takes as input the new information from the current time-step (X_t) and the output of the previous time-step (h_{t-1}). With this input it then decides what information to forget from the cell state of the previous time-step (C_{t-1}) through the *forget gate* and with that compute its own cell state (C_t). The next step is to decide what new information it's going to store in its own cell state, which takes place in the *input layer*. This layer decides what values to update and with how much the values have to be updated through the *input gate*. Finally, the unit has to decide what it's going to output (h_t), which takes place in the *output layer*. The output is a filtered version of the updated cell state and the current input, and is computed through the *output gate*. Summarizing, the forget gate controls the extent to which a value remains in the cell state, the input gate controls the extent to which a new value flows into the cell state and the output gate controls the extent to which the value in the cell state is used to compute the output of the LSTM unit.

2.2.4 Gated recurrent unit

The GRU was proposed by Cho et al. (2014) as a new recurrent unit that is easy to train while also avoiding the vanishing gradients problem. GRU is related to LSTM as both are utilizing different ways of using internal gates, but GRUs are found to be less computationally expensive due having fewer number of those gates. It combines the forget and input gates of the LSTM into a single *update gate*. Also, GRUs do not maintain separate cell states but instead expose the complete memory by merging the cell state and hidden state. Figure 2.6 shows the architecture of a GRU unit. Furthermore, similarly to the LSTM unit, a step-by-step walk through of the

GRU unit is provided. Since the GRU architecture is central to the forecasting component of this research project, the mathematical expressions behind it are also included (Ian Goodfellow, 2017).

The update gate takes as input the new information from the current time-step (X_t) and the output of the previous time-step (h_{t-1}). The update gate is calculated by multiplying both inputs with their weight factors, $W^{(z)}$ and $U^{(z)}$ respectively, and summing them. The sigmoid activation function is applied to the sum in order to squash the result between 0 and 1. Equation 2.1 is a general sigmoid function and equation 2.2 shows the equation that calculates the update gate. The update gate determines how much information from previous time-steps is preserved and passed along to the future.

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2.1}$$

$$z_t = \sigma\left(W^{(z)}X_t + U^{(z)}h_{t-1}\right) \tag{2.2}$$

The other gate within a GRU unit is the *reset gate*. Equation 2.3, which belongs to the reset gate, has the same format as the update gate equation. However, the weights and gate's usage are different.

$$r_t = \sigma\left(W^{(r)}X_t + U^{(r)}h_{t-1}\right) \tag{2.3}$$

After calculation of both gates, a new temporary memory variable h'_t is introduced (equation 2.5). This variable uses the result from the reset gate (r_t) to store the relevant information from the past. To calculate h'_t, first h_{t-1} is multiplied with a weight U, after which the element-wise product of this multiplication and r_t is determined. The element-wise product is then summed with a multiplication of a weight W and the input X_t. This sum is finally squashed between -1 and 1 through the *tanh* activation function (2.4).

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.4}$$

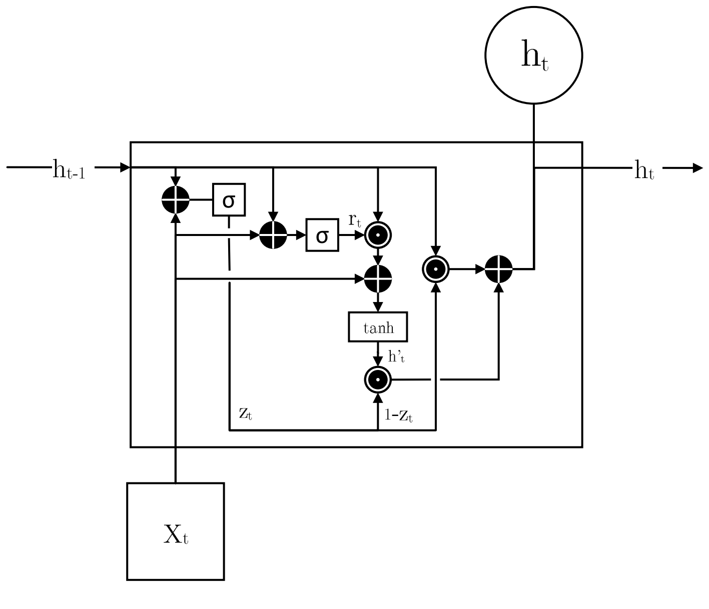


Figure 2.6: GRU unit

$$h'_t = \tanh\left(Wx_t + r_t \odot Uh_{t-1}\right) \tag{2.5}$$

Finally, the network needs to calculate the output vector h_t. To do so it uses the temporal current memory content h'_t, the output of the previous step h_{t-1} and the result of the update gate calculation z_t. Corresponding equation 2.6 shows the element-wise multiplication between z_t and h_{t-1} which is summed with the element-wise multiplication between $1 - z_t$ and h'_t.

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t \tag{2.6}$$

As can be seen, GRUs are able to store and filter information using the update and reset gates. A GRU unit does not wash out new input every single time step like in a classic RNN unit but rather keeps relevant information and passes it down the next time steps. This eliminates the vanishing gradient problem with a relatively simple architecture. In order to teach a GRU model when to allow updates and resets, the weights associated with the gates are adjusted and optimized during a BPPT learning process.

2.2.5 GRU in time-series forecasting

The GRU structure was introduced in 2014 and thus is relatively new. It is still a partly undiscovered field of research in several sectors that deal with time-series. Shen et al. (2018) for example conclude that, although GRU networks have great potential in the financial time series prediction, they are seldom used in the field. More closely related to this research project, Fu et al. (2016) concluded only two years ago that their paper is the first time that GRU is applied in traffic flow prediction. Only a handful of papers concerning traffic flow prediction with GRUs have been published since (e.g. Zhang and Kabuka, 2018; Song and Lu, 2018; Fandango and Wiegand, 2018). After an extensive literature review, the latter conclude that there are very few papers for applying the GRU architecture in the more specific *short-term* traffic flow forecasting context. As mentioned, this research project focuses on a sub-category of short-term traffic flow forecasting, called short-term *passenger* flow forecasting.

Literature regarding short-term passenger flow forecasting using GRU networks is sparse. As indicated in Section 2.1.2, only two relevant articles have been found within the public transport sector. Additional research with an even broader scope does currently not yield any other relevant literature. Both articles that were found are further analyzed in order to determine the scientific contribution of this research project.

Ji and Hou (2018) use bus entry data to construct models that are able to predict different time interval lengths. The only data feature that they use concerns the aggregated amount of people that enter a bus during the chosen time interval. Furthermore, their model is only able to generate a single-step ahead forecast.

Toque et al. (2018) use larger models that use more data features and that are able to generate multi-step ahead forecasts. The additional data features are all related to calendar information such as month, day, holiday, or extended weekend. By using the past historical observations and additional calendar features, they try to predict the number of passengers entering the 30 different studied railway stations at the 8 next time steps (data is aggregated with 15 minute time steps). Both univariate and multivariate models are considered to forecast passenger flows. The univariate model uses the past observations of one station's time series and the calendar information to predict the next value of the same time series. The multivariate model uses the past observations of all the station's time series to predict the next values of all the stations. We argue that the multivariate models are only weakly applicable in their context since the different time-series have a limited temporal link between each other. In their paper, they use the number of passengers entering the 30 stations of the multimodal transport network of the business district La Defense of Paris in France. These numbers are gathered through smart card data: passengers swiping their smart card when entering the station. Passengers entering one station will usually not *enter* another station within the multi-step ahead time-window, but will rather *depart* from another station. We therefore consider the temporal link between the different passenger entry time-series to be minimal. Arguably, their multivariate models mainly perform well due to the high capability to generalize and not due to the temporal links between the different time-series. These findings leave room for additional research, which is addressed in the next section.

2.3 Scientific contribution

First and foremost, as indicated, an extensive literature review specifically aimed at short-term passenger flow forecasting within the airport context did not yield any relevant results. Thus, at least from a scientific point of view, this field is totally undiscovered. A first scientific contribution of this research project therefore lies in the insights found when developing, using and analyzing a short-term passenger flow forecast model within an airport context.

Secondly, as argued in the previous section, there is room for additional research in general short-term passenger flow prediction using GRU networks. Both found articles use a single source of time-series as their main data feature (historical bus/train station entries). We deem the additional calendar data features used in Toque et al. (2018) to be characteristics of the main data feature time-series and are thus not considered to be a separate data source. Furthermore, the multivariate models used in that article are argued to be weakly applicable due to limited temporal relationships among the different time-series. Contrary, in this research paper, a multi-source multivariate GRU forecasting model with clear temporal relations is proposed. Multi-source refers to the usage of additional types of time-series as data feature next to the type of time-series the model tries to predict. An example is that of also using the recent bus arrivals at the airport terminal to forecast the upcoming passenger flow within the terminal. Furthermore, the literature review in Section 2.1 yielded only one paper that considers multi-source data features (Ding et al. (2016) use bus transfer activities to predict short-term subway ridership). The second scientific contribution therefore lies in the insights found when developing, using and analyzing a multi-source multivariate GRU short-term passenger flow forecasting model with clear temporal links between the data sources. Furthermore, this model will be compared to several other short-term forecasting models (e.g. vector auto-regression and LSTM). The insights gained when comparing these models can be considered to be a third scientific contribution.

The scientific contribution of this study can be summarized in the three following points:

1. Insights found when developing, using and analyzing a short-term passenger flow forecast model within an airport context.

2. Insights found when developing, using and analyzing a multi-source multivariate GRU short-term passenger flow forecasting model with temporal links between the data sources.

3. Comparison of a multi-source multivariate GRU short-term passenger flow forecasting model with other short-term forecasting models.

2.4 Conclusion

At the start of this chapter, we have identified nine different forecasting method categories that are used throughout 22 separate scientific papers to forecast the short-term passenger flow. Of those nine categories, we find that RNN's are the most promising technique for the goals of this research project. More specifically, we consider GRU based RNNs to be the most appropriate method.

Furthermore, the literature review results in the conclusion that the field of short-term passenger flow forecasting in the specific airport context is completely undiscovered. This research paper aims to fill that gap. Also, a broader literature search that also includes the field of public transport finds that the vast majority of the proposed forecasting models merely use a single source time-series and its derivatives (e.g. day of week) as input for the forecasting model. Instead, we propose a multi-source multivariate forecasting model with clear temporal links between the data sources. Filling the gap in the airport context and proposing a multi-source model are seen as main scientific contributions of this research paper.

Finally, in Chapter 1 we introduced five sub-questions to answer the main research question. Using the results of this chapter, we can update the third sub-question so that it is less broad. The third sub-question as of Chapter 1:

3. How to design a model to forecast the short-term passenger flow at Eindhoven Airport's security filter in real-time?

With the newly obtained knowledge from this chapter, the sub-question is updated:

3. How to design a multi-source multivariate gated recurrent unit neural network forecasting model that relies on temporal links between the data sources in order to forecast the short-term passenger flow at Eindhoven Airport's security filter in real-time?

Chapter 3

Methodology

In order to effectively and thoroughly investigate and solve a specific matter or problem the research has to be performed systematically (van Aken et al., 2012). This chapter therefore introduces the research methodology that is used throughout the research project to structure the overall research process. Furthermore, for each of the five sub-questions this chapter indicates how they will be answered. Finally, the methods for information gathering that are used throughout this research project are identified.

3.1 Problem solving cycle

The research follows the problem solving cycle, which is a design-oriented and theory-informed methodology aimed at developing solutions for field problems (van Aken et al., 2012). The problem solving cycle methodology is particularly useful when a business problem arises within a company. Business problems are often a set of interrelated problems, also known as the concept of a 'problem mess' (Ackoff, 1981). To formulate a clear business problem, this problem mess has been identified and structured in the preliminary research proposal and summarized in Chapter 1. This structuring process is the first step of the problem solving cycle and resulted in a problem definition. This step is followed by four more steps, which will eventually lead to a problem solution that is implemented and finally evaluated. The cycle can be depicted as in Figure 3.1. During this research project the two following steps, *analysis and diagnosis of the problem* and *solution design* are executed. These two steps are broad and not specific for the research project at Eindhoven Airport. Therefore, in order to further structure the research project, these two steps are broken down according to the CRISP-DM methodology in the next section. The other two phases (intervention, learning and evaluation) will only be partly addressed and will primarily serve as an initial evaluation of the solution design. We deem full execution of the problem solving cycle to be in-feasible due to the imposed time constraints on this project.

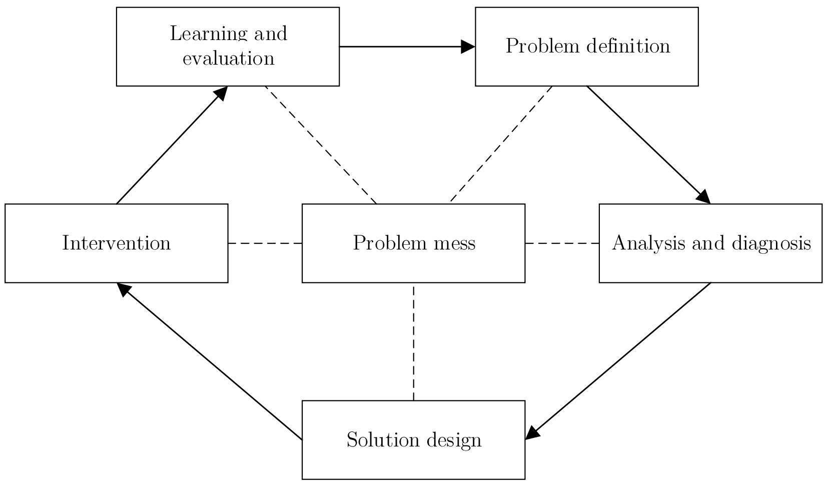


Figure 3.1: Problem solving cycle

3.2 CRISP-DM

CRISP-DM (Cross Industry Standard Process for Data Mining) methodology provides a structured approach to planning and executing a data mining project (IBM, 2011). Even though this research is more of a data science project opposed to a pure data mining project, as argued by Vorhies (2016), CRISP-DM provides strong guidance for even the most advanced of todays data science activities since the basic constructs are the same. In the generic CRISP-DM reference model, the life cycle of a data science project is broken down into six phases as depicted in Figure 3.2.

In this framework, business understanding is the initial phase and focuses on the project objectives and requirements from a business perspective and converting this knowledge into a data science project. The subsequent data understanding phase starts with initial data collection and activities in order to get familiar with the data (quality, first insights, detect interesting subsets etc.). The data preparation phase focuses on constructing useful data sets. In the modelling phase different techniques are discovered, selected and applied. These models are then tested in the evaluation phase and finally the champion model is deployed in the deployment phase (Wirth and Hipp, 2017).

These phases overlap with the phases in the problem solving cycle: *business understanding* and *data understanding* are covered by the analysis and diagnosis phase while *data preparation*, *modelling* and *evaluation* are captured in solution design. *Deployment*, which is identified as the last phase in the CRISP-DM framework, has a close match with the intervention step in the problem solving cycle, and is only partly addressed in this research project.

It is important to indicate that although the CRISP-DM framework provides an idealized sequence of stages, these stages do not necessarily follow each other slavishly, but are often performed iteratively. The stages can be performed in a different order and it might be necessary to backtrack to previous stages and repeat certain actions. Documenting the complete process (going back and forth between stages) in this research project would result in a confusing and incoherent report. Therefore, in this paper, only the final outcomes of each step are documented and structured according to the ideal sequence of stages. The stages and their corresponding sub-questions are now introduced.

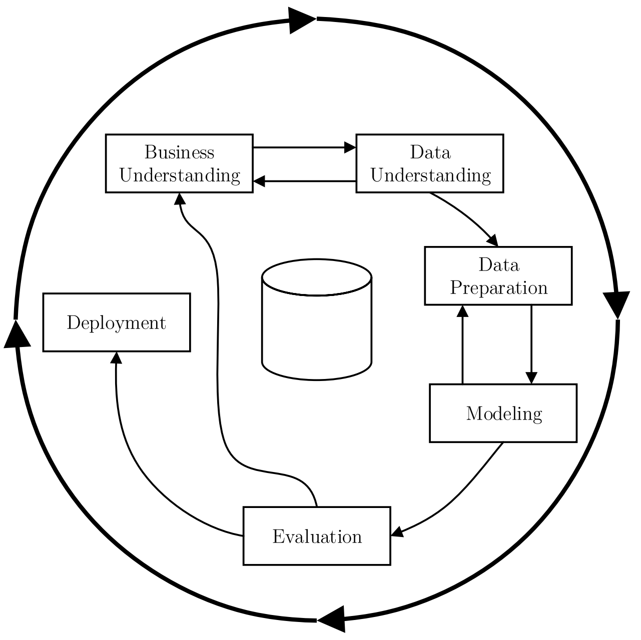


Figure 3.2: CRISP-DM framework

3.2.1 Business understanding

A large part of the business understanding phase has been addressed in Chapter 1. In that chapter the business problem has been analyzed and it's root causes and possible solution directions were identified. As a business goal it is identified that the research should provide insights on how to reduce the severity and number of bottlenecks experienced in the outbound passenger flow, that are a result of the capacity crunch and amplified by inadequate flow management. Another part of the business understanding phase lies in assessing the current business situation. This assessment is addressed in Chapter 4 and provides a more detailed description of the as-is outbound passenger flow situation and it's bottlenecks and corresponding flow management. This chapter provides the answer to the first sub-question.

3.2.2 Data understanding

The data understanding phase of the CRISP-DM is addressed in Chapter 5 and involves taking a closer look at the as-is data situation of Eindhoven Airport. The phase identifies which data exists, where the data is located and identifying the quality and availability of the data. Firstly, every data source that falls within the scope of the outbound passenger flow at Eindhoven Airport is identified and listed. Secondly, the list is evaluated in order to identify the data sources that would ideally be accessible for the duration of this research project. Subsequently, the ideal situation is converted into a workable situation that only includes data sources that are realistically accessible for this research project. Finally, data quality from the latter data sources is verified. The results from this phase provide the answer to the second sub-question.

3.2.3 Data preparation

Using the information gathered through the business and data understanding phases, a set of data features is constructed in Chapter 6. This chapter indicates which data features will be used in the model, how they are constructed and what the reasoning behind their selection is. This data is then cleaned and a final data set of the chosen data features is constructed.

3.2.4 Modelling

In the modelling phase of the CRISP-DM framework the final forecasting model is constructed. As indicated, the main model that is to be tested is a GRU based RNN. Furthermore, this model will be offset to other models to compare its performance. The plan for training, testing and evaluation is outlined in this phase of the project. The different models are ranked and compared with pre-defined evaluation techniques. Finally, the best performing forecasting model is chosen. The results from the previous data preparation phase and the results from this modelling step provide the answer to the updated third sub-question (see Section 2.4). Although this modelling step already includes an evaluation of the forecasting models to find the best model, this evaluation does not serve as the evaluation phase in the CRISP-DM framework. The CRISP-DM evaluation phase is explained in the next section.

Subsequently, this model is used as the basis for a decision-support model that provides the AOMs with the real-time forecasted passenger flow at the security filter, the computed expected waiting time and the possibilities of mitigation for upcoming bottlenecks. Development of this decision-support model is documented in Chapter 8 and provides the answer to the fourth sub-question. We thus employ two modelling steps: first a short-term forecasting model is developed and secondly a decision-support model is constructed that integrates the forecast model with a waiting time model and an action proposal model.

3.2.5 Evaluation

The evaluation phase is concerned with assessing the degree to which the model meets the business objectives and seeks to determine if there is some business reason why this model is deficient. The dashboard constructed in the modelling phase is piloted for several weeks and used by the relevant stakeholders. A quantitative and a qualitative evaluation is performed. The first is concerned with forecast accuracy whilst the latter is concerned with the model's expected usefulness and perceived accurateness by the stakeholders. The results of this evaluation are presented in Chapter 9 and provide the answer to the last sub-question.

The last phase of the CRISP-DM framework (deployment) is only partly executed during the pilot of the model, and does not have a separate chapter in this research project.

3.3 Information gathering

A considerable portion of this research relies on knowledge that resides in the company. For example, the identification of the problems and their causes and effects of Chapter 1 was based on multiple interviews with the relevant stakeholders. As indicated in the company profile (1.1.1), these relevant stakeholders are predominantly airport operation managers. In this report, there will be numerous mentions of these stakeholders and the information they provided. This information will be mainly gathered through (short) semi-structured interviews with these stakeholders (van Aken et al., 2012), in which a few specific questions will be constructed but sufficient room is left for additional information. Furthermore, continuous discussions will take place with the company supervisor of this research project (Information Manager). The AOMs can be considered to all have a different view on the processes in place and can therefore be seen as separate stakeholders instead of a group of homogeneous stakeholders with the same views. Statements made by one AOM can therefore be verified or possibly challenged by other AOMs in order to finally collect the correct information. A more detailed description of the responsibilities of the AOMs is now provided.

Eindhoven Airport currently employs eight AOMs. As indicated, an AOM is the first overall responsible person in the day-to-day operation and is responsible for supervising the coordination between the processes in the airport and making sure that all the operations are running smoothly. The AOM oversees the security needs and helps in enforcement of rules and regulations in the airport. Hence the position of an AOM is vital in the airport to keep the airport functioning properly. AOMs need to have extensive knowledge of the functioning of the airport and should have good management skills as well. During all operating hours of the airport (04.00 – 00.30) and on every day of the week, one of the eight AOMs is on duty. Weekdays are currently divided into three shifts whilst Saturdays and Sundays have two shifts. Playing an essential and central role in the management of the operations, the AOMs are key stakeholders in this research project. Most of these AOMs have worked at the airport for an extensive period of time and have a lot of experience. Also, all AOMs have secondary responsibilities that they attend to when they are off-duty (e.g. security policy or capacity planning). Finally, this means that they all have a different view on the processes in place and they can therefore be seen as separate stakeholders instead of a group of homogeneous stakeholders with the same views.

Chapter 4

Business understanding

In Chapter 1 the business problem has been defined with the aid of a cause-and-effect diagram. Furthermore, the outbound passenger flow has been defined as the initial business process scope. Subsequently, this chapter addresses the business understanding phase of the CRISP-DM framework and complements the business problem and its scope. Firstly, a detailed overview of the outbound passenger flow and the physical layout of the airport is provided. Secondly, the bottlenecks in the outbound passenger flow are identified and finally the final research scope is set using the complete business understanding.

4.1 Overview

4.1.1 Passenger flow

As indicated, the projects scope captures the outbound passenger flow at EA. From a passenger perspective the start of the outbound passenger flow is defined as the moment when the departing passenger arrives at the airport terminal. Departing passengers have 3 main ways of arriving at the terminal: by bus, drop-off or car park. The number of passengers that arrive at the terminal with other means of transport (e.g. by bike or foot) is negligible (MarktEffect, 2018). After arrival the passenger enters the terminal and follows the flow as defined in Figure 4.1. It must be noted that this is a typical scenario and does not include exceptions (e.g. flight cancellation).

In this high level overview four different types of processes have been defined. A process that includes a touch-point is defined as a process during which the passenger interacts with one or more of EA's staff members (e.g. check-in) or directly interacts with one of EA's physical devices (e.g. pre-security filter gate). Some processes are mandatory for all passengers whilst others are optional and depend on the passengers' or airlines' preferences. Eight different processes that include a touch-point are found to be optional:

Firstly, depending on the airline, passengers can skip check-in at the terminal by checking-in online. Additionally, not all passengers have luggage to drop off and also only a select group of passengers require additional assistance or airline service. Furthermore, an additional High

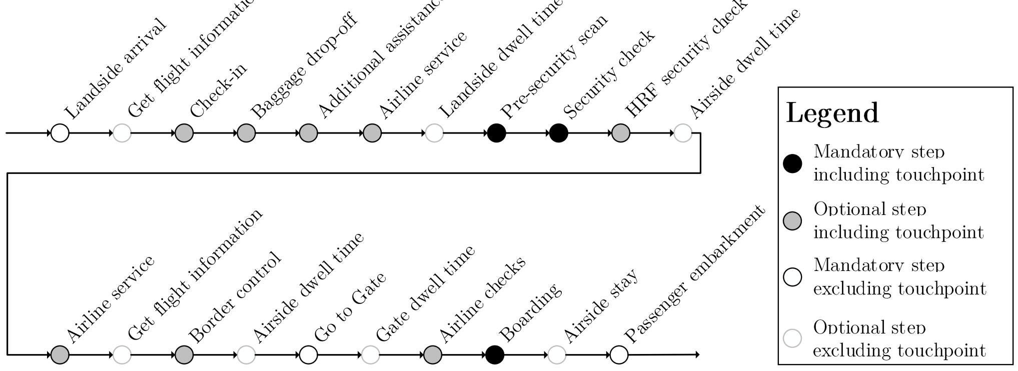


Figure 4.1: Outbound passenger flow

Risk Flight (HRF) security check is only required for certain flights that are deemed high risk by the national counter-terrorism unit and border control is only necessary for flights to countries outside the Schengen border treaty. Finally, airlines can require extra checks prior to boarding (e.g. passenger name check or carry-on baggage size check).

Next to these optional processes, three processes have been identified to be mandatory and also serve as a touch-point. Firstly, during the pre-security scan process the passengers scan their boarding card at one of the pre-security filter gates. These gates determine if the passenger is allowed to enter the security filter (e.g. flight date is required to be today). Secondly, after passing through the pre-security filter gates, passengers go to the security filter. During the subsequent security check the passengers and their valuables are scanned by an x-ray scanner and potentially searched manually by a security officer. Lastly, to be allowed to embark on the aircraft, passengers are required to scan their boarding card at one of the boarding gates that belongs to their flight. This is the last touch-point during which passengers directly interact with EA's staff members or physical devices.

The other processes do not serve as a touch-point and their execution is mainly passenger specific. These processes are less relevant from a passenger flow management point of view since there is no direct interaction with the passengers and thus allow for limited control.

4.1.2 Layout

A representation of the physical lay-out of the terminal is provided in Figure 4.2. The terminal is divided into a *land-side* area and an *air-side* area, which are represented with a white and black floor respectively. The distinction is made to determine what part of the airport is authorized access only (air-side). The self-service baggage drop-off units (SSBD) are located on the ground floor on the left of the escalator in the land-side area. Next to the SSBD units, on the right side of the escalator, the assistance and check-in counters are located. The airline service counters are located just in front of the entrance to the pre-security filter. This pre-security filter is located on the far right of the ground floor, and serves as the cross-over from land-side (white) to air-side (black). The security filter (queue area and security lanes), are located directly behind the pre-security filter. On the top far right of the ground floor gate 1 can be found and the last gate (gate 12) can be found on the far left of the 1st floor. Finally, border control is located right next to the escalator on the ground floor of the air-side area. Border control allows access to gate 8-10.

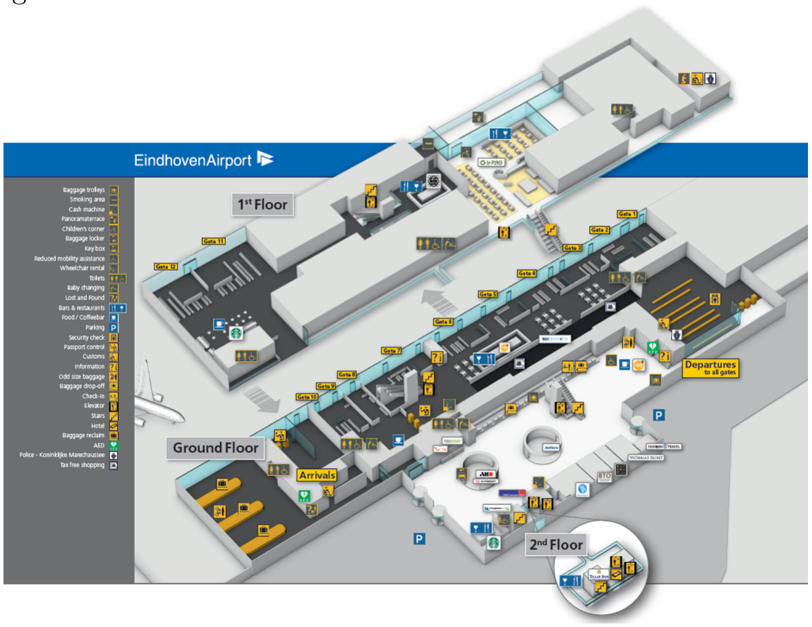


Figure 4.2: Eindhoven Airport layout

4.2 Bottlenecks

As summarized in the problem statement of Section 1.2.4, we find that there is an increasing risk of operational bottlenecks and a decrease in their associated solution scope in the outbound passenger flow. It is therefore important to identify and analyze the different bottlenecks in order to further scope the business problem. A total of seven bottlenecks are identified through interviews with the relevant stakeholders, monthly customer survey results (MarktEffect, 2018) and EA's key performance indicators (KPI). A basic overview of the flow management actions that the AOMs take to mitigate the problems is provided, as well as the resources that are planned for the specific bottlenecks. The identified actions are currently mostly executed after the problem already exists, since the AOMs currently have limited options to foresee problems.

4.2.1 Self-service baggage drop-off units

EA has 12 SSBD units. Passengers that check-in online are able to use the units to drop-off their baggage without staff assistance. Naturally, during busy operating hours, the 12 units can form a bottleneck and result in unacceptable queue times. EA employs a KPI that tracks the number of times the queue time exceeds 5 minutes. Exceeding of this queue time is tracked by Viggo and was recorded only 8 times over the last year. However, there is some uncertainty about the accuracy of this statistic since the stakeholders believe that KPI is only recorded during a lot longer queues (over 10 minutes). Nonetheless, the stakeholders indicate that the number of SSBD units is currently sufficient to deal with peaks and they only become a bottleneck during technical failures in high demand periods. However, the stakeholders see an upward trend in number of passengers that use the SSBD units due to changing airline regulations and also foresee higher peak demands due to an increasing number of flights. Therefore, although the SSBD units do currently not result in a lot of problems, this could change in the future.

The most prominent mitigating option that the AOMs have in the case it does get crowded at the SSBD units, is to request a relocation of staff members that are performing a different task in the near proximity. These staff members can then assist passengers that experience difficulties with the baggage drop-off procedure. This has the potential to speed up the flow significantly because those type of passengers would block the unit for a long time without assistance. Moreover, the SBDD units can be assigned to specific flights (indicated by a digital display above the unit): in case of long queues, more units can be assigned to flights that are leaving in the near future so as to prevent delays or passengers missing their flights. As per resource planning, the SBDD units are commonly assigned to specific airlines and their flights (e.g. 4 units for Ryanair, 4 for Wizzair and 4 for Transavia), without any elaborate scheduling behind this. Furthermore, one staff member is scheduled to assist at the SBDD units throughout the day, thus also lacking any elaborate scheduling procedure.

4.2.2 Assistance counters

Some of the airlines obligate passengers to check-in online and drop-off their luggage through the SSBD units. Whenever passengers traveling with these airlines are not able to check-in online or experience problems when using the SSBD units, they can attend one of the additional assistance counters. These counters are staffed and depending on staff-planning a maximum of 4 counters is open. Passengers that require assistance line-up in a designated queue area that is cordoned off with banklining. As per one of EA's KPIs, this queue length is deemed to be unsatisfactory when the queue physically passes the start of the marked queue area. Number of exceedances of this queue length shows an erratic pattern: most months only have 2 occurrences while the highest number of occurrences within a month is 19. The stakeholders indicate that these exceedances mainly occur when big groups of passengers are not able to check-in online due to technical difficulties on their airlines website, which could partially explain the erratic pattern. With an expected upcoming increase in flights and passengers, the bottleneck that is caused by the assistance counters is likely to become more salient. However, the stakeholders deem predictability and proactive flow management difficult due to the external factors that are the main cause for severe bottlenecks at the assistance counters.

Similar to the flow management options for the SBDD units, the most prominent mitigating option that the AOMs have is to request a relocation of staff members that are performing a different task in the near proximity so as to open additional assistance counters. Furthermore, passengers that are on a flight that is departing in the near future can be requested to come forward so that they can be assisted first. The assistant counters are common-use and thus not assigned to a specific airline. The planning for the number of open counters is done manually and the planner roughly takes into account how many flights are leaving in a certain time window, without the use of any refined planning or forecasting techniques.

4.2.3 Check-in counters

Opposed to the airlines that only focus on online check-in, some airlines require passengers to check-in at the airport through one of 9 conventional check-in counters. The related queue length KPI works in the same fashion as the assistance counter KPI, in which the exceedance of the queue past a predefined physical point is recorded. Similar to the baggage-drop off counters, a limited number of these exceedances has been recorded over the past year, averaging 1.5 occurrences per month. According to the stakeholders this low number is mainly due to the limited number of airlines that require their passengers to check-in on the airport. This could however easily change in the future due to new airlines or an increase in flights for the airlines that currently employ this requirement. Due to their comparative nature, the main flow management options that AOMs have to alleviate problems at the check-in counters are the same as for the assistance counters. The planning of these counters is however slightly different since the counters are assigned to specific flights. This planning mostly is a result of contracts with the airlines that state how many and when to open the check-in counters for which flights.

4.2.4 Airline service counters

EA also has general airline service counters where a variety of services are provided to the passengers. Examples of the provided services are the payment of excess baggage or a name change and reprinting of a boarding card. Two staff members are scheduled to operate the service counters during the day. The airport does not currently track the queue length in front of the service counters by means of a KPI. According to the stakeholders, the service counters do not form a bottleneck in normal operation and only form a bottleneck during abnormal situations when multiple flights are promptly cancelled (e.g. due to weather conditions). During these situations, numerous of these passengers will request service at the counters in order to for example reschedule their flight or demand an overnight stay at a nearby hotel. The mitigating actions that an AOM can take are limited to requesting a relocation of staff members or requesting direct colleagues to assist at the service counters.

4.2.5 Pre-security filter and security filter

All outbound passengers enter the pre-security filter during which they scan their boarding card in order to be allowed to enter the security filter. The pre-security filter has four self-service e-gates and one staffed scan counter. The gates only open when the boarding card passes a certain set of conditions, e.g. the flight departs in the near future and the passenger paid for their excess baggage. Once the passenger has passed the pre-security filter, he directly enters the security filter queuing area that is cordoned off with banklining in a 'zig-zag' shape. When this queuing area is completely full, a queue can form in front of the pre-security filter. In normal operation (no breakdowns), the four e-gates and one manual scan counter are sufficient to process all passengers directly, meaning that virtually no situations occur in which the security filter queue area is only partially filled and yet there is still a queue in front of the pre-security filter. Analysis has shown that in the last year a maximum of 413 passengers passed through the pre-security filter in a 15 minutes time interval, whilst the theoretical maximum capacity of the e-gates alone (excluding manual scan counter) is already at a combined 600 passengers. An important assumption throughout this research project is therefore that the security filter queue and the pre-security filter queue are directly connected and thus operate as a single queue. With other words, the pre-security filter on itself is assumed to not be a possible bottleneck.

This leaves the security filter as a bottleneck. The security filter consists of 6 security lanes that are located directly next to each other. Passengers wait in a single queue and are only referred to one of the open security lanes at the end of the queue. A planning of the lane configuration is constructed by one of EA's AOMs every week. This lane-configuration planning states exactly how many lanes will be open during every 15 minute time interval in a day and also indicates the number of staff and their respective required skills (e.g. security officer or queue assistant). The specific configuration for a 15 minute time interval corresponds to a contracted rubric code which relates to a contracted throughput level. A total of 21 different configurations are agreed upon (A to U) with G4S (the company that delivers the staff for the security lanes). To illustrate, the rubrics letter I for example corresponds to three out of six lanes being open, with one supervisor, ten security officers, three security assistants and one queue assistant. This in turn should lead to an average throughput of 3.3 passengers per lane per minute, provided that there is a constant influx of passengers.

Similar to the assistance and check-in counters, the maximum acceptable queue length is measured by the exceedance of the queue past a predetermined physical point in the queue area. This physical point is located before the pre-security filter. Over the past year an average of 27 occurrences have been recorded per month during in which the queue length and thus the passengers queue time is deemed to be unacceptable. It becomes clear that this is a severe bottleneck within the outbound passenger flow at EA. According to the stakeholders the majority of the unacceptable queue lengths are due to a discrepancy between the planned lane-configuration and erratic arrival spikes of passengers. The planning is currently mainly based on educated guesses concerning the number of passengers on the individual flights during a day and applying a Poisson arrival rate to those guesses. With this, the expected number of passengers that will arrive at the security filter during every 15 minute time interval is computed. These numbers are then matched with an appropriate lane-configuration that has an adequate contracted throughput level. The high number of unacceptable queue lengths suggests that this method is not fully able to grasp the complexity of passenger arrivals. Furthermore, with an increase in passengers and a higher constant flow of passengers throughout the day, this problem is expected to grow in the future.

As illustrated in the problem exemplification (1.2.3), the main mitigating action that the AOM can perform is to request the crew-lane to be opened to passengers. Furthermore, the airport has a public address system with which passengers can be informed throughout the terminal (e.g. to urge missing passengers to go to their gate). It can also be used to make an announcement stating that the security filter is currently closed for a specific flight so as to prevent the existing queue from growing even further. Another option is to request direct colleagues (e.g. off-duty AOMs) to assist at the security filter.

4.2.6 Border control filter

Passengers travelling to a destination that is outside of the Schengen area have to pass through border control after having passed the security filter. The border control filter is located at a different area of the airport and consists of 6 counters that are staffed by employees of the Royal Netherlands Marechaussee (KMar). Depending on the number of non-Schengen flights during the day, a request is made in advance to the KMar that details the required number of staffed counters during a certain time-interval. It is however up to the KMar to decide if they adhere to the request, with very limited influence of EA. When multiple non-Schengen flights are scheduled during the same time interval and the KMar does not employ full capacity, this bottleneck can lead to severe queue lengths. Due to the limited control over this bottleneck, EA does currently not employ a KPI that is related to the queues in front of the border control filter. This is seen as an influential bottleneck in the passenger flow from both a passenger and a business perspective, thus EA is keen on finding mitigation options to prevent bigger problems resulting from the looming capacity crunch.

4.2.7 Gates

Contrary to the other bottlenecks, the last identified bottleneck is not directly involved with unacceptable queue lengths but more with general unacceptable crowdedness. EA has 12 gates that are all located next to each other, where gate 1 is located on the far right and gate 12 is located on the far left (albeit gate 11 and 12 are on a different floor). The gates provide a queuing area for passengers before boarding their flight. However, the queuing areas of these gates are not spacious enough for all passengers to reside inside. Approximately half of the passengers of a full flight (e.g. 100 out of 200) fit in the queuing area. This means that the rest of the passengers have to queue outside the designated gate area and thus occupy the aisle that connects all areas of the airport (e.g. gates, shops, restaurants), which creates an obstruction for passengers of other flights that are moving between areas. This problem is more stringent for the gates that are in the middle of the airport (e.g. 5-7) compared to the gates on the far end of the airport (e.g. 1-2 and 11-12). These bottlenecks, created by passengers waiting in the main aisle, are a serious issue that also comes forward prominently in the customer surveys.

According to the stakeholders, gate planning (which flight departs from which gate) is very important in alleviating the problem. Currently, the planning is made manually and is mainly based on the experience of the planner. Factors that could however for example be incorporated in planning are the number of passengers on a flight, the announcement time of gates to passengers, number of passengers with restricted mobility (PRM), flight delays and flight destinations (different types of travelers). These factors are currently barely taken into account due to the lack of available data and the manual construction of the gate planning. This is a clear example of the capacity crunch and the problem is expected to become worse with a growing number of flights. However, because general crowdedness is difficult to measure, there currently is no KPI to track when and where the problems occur.

4.3 Scoping

4.3.1 Focus on proactive flow management

The problem analysis of Chapter 1 resulted in a problem statement, indicating that the looming capacity crunch combined with the amplifying effect of an inadequate flow management style lead to an increase in operational bottlenecks and a decrease in the associated solution scope. These problems have been further elaborated on in this chapter by zooming in on the actual bottlenecks, the related reactive flow management style and the resource planning in the outbound passenger flow. Furthermore, the identified practical goal indicates that this research should provide insights on how to reduce the severity and number of these bottlenecks that are experienced in the outbound passenger flow. Since a problem is best dealt with by addressing one of its main causes, we look at the identified causes (1.2.1) for possible solution directions. Two main cause areas have been identified that are both deemed to be two-fold and that together result in an imperfect passenger flow: the capacity crunch (capacity vs. growth) and inadequate flow management (insufficient resource planning and reactive flow management). In this section we argue that addressing the latter (reactive flow management) is the most interesting from a business and scientific point of view.

Firstly, concerning *capacity*, in order to reduce the severity and number of bottlenecks experienced in the outbound passenger flow, one could simply increase the infrastructural capacity at the airport. However, relevant stakeholders have indicated that infrastructural capacity will not grow in coming years, in fact, it will presumably slightly decrease due to the expansion of commercial areas. For the course of this project, we therefore assumed that the infrastructural situation will remain as it is now.

Secondly, concerning *growth*, the increasing number of problems resulting from the capacity crunch could simply be halted by limiting growth or even allow for a decline in passenger numbers. However, the stakeholders indicate that EA does not wish to limit growth to overcome operational difficulties.

Thirdly, the combination of these two, which is indicated as the *capacity crunch*, has been a subject of study for an extended period of time and has been explored extensively at different airports (e.g. Hee and Zeph, 1998; Gatersleben, 1999; Yanbing et al., 2007; Chiang and Taaffe, 2014; Alodhaibi et al., 2016; Nmmik and Antov, 2017; Adacher et al., 2017). The studies for example use simulation models, based on historical data, to gain insights into the presence of bottlenecks and their causes, and to evaluate the key performance measures of the system. Through optimization, these studies then determine how different resources can be reassigned or infrastructure rearranged in order to increase the utilization of facilities and improve the service level. Due to the abundant literature this is not an appropriate research direction for this master thesis. To sum up, we argued that the *capacity*, *growth* and *capacity crunch* cause areas are unsuitable to serve as research directions.

This leaves three causes that can be addressed to provide the insights as required in the practical goal: *insufficient resource planning*, *reactive flow management* and the combination of the two (*inadequate flow management*). Concerning the latter, despite the close relation between the two issues (resource planning and flow management), they are two separate research directions of which the combined scope is too large for one research project since they require a fundamentally different theoretical approach.

Concerning *resource planning*, we find that this has been a long studied subject in literature, covering countless studies. Also, as indicated in previous paragraphs, numerous airport terminal specific studies have already been published that primarily focus on the resource planning to address the capacity crunch. Furthermore, the airport is a highly static, but at the same time a highly dynamic environment. To illustrate, the airport computes static data weeks in advance concerning the expected number of passengers that will flow through the airport during a given day. However, the passenger flow during that day is highly dynamic and subject to change. Therefore, even with enhanced resource planning which is based on extensive (historical) data, dynamics during the day are expected to be difficult to capture beforehand. This problem clearly comes forward when looking at the security filter bottleneck in the previous section.

We finally argue that addressing the *reactive flow management style* is the most relevant research direction from both a scientific and business point of view. We propose to address this root cause by enabling a proactive flow management. A proactive flow management style is seen as a style in which the main focus is on issue prevention opposed to issue resolving. To enable a proactive flow management style, the stakeholders require real-time insight into the current state of the outbound passenger flow and more importantly require insight in the short-term future state of the flow. This future state can primarily be obtained by real-time forecasting of the short-term passenger flow. The usage of real-time short-term forecasting in flow management is believed to be a relatively new subject in scientific literature, emerging from the increasing availability of real-time data (e.g. due to Internet of Things).

Subsequently, these forecasts can be used to provide additional decision-support to the stakeholders in order to further enable proactive flow management. To illustrate, additional decision-support for example can be provided in the form of derived statistics (e.g. computing the expected queue length by combining the forecast with the planned capacity) or action proposals (e.g. provide the stakeholders with a range of possible mitigating actions for upcoming bottlenecks). By preventing issues instead of reacting to them, the airport is able to get one step closer to the desired seamless passenger flow.

4.3.2 Focus on security filter

Enabling proactive flow management in all aspects of the outbound passenger flow is not feasible within the time constraints that are imposed on a single master thesis project. The analysis of the different bottlenecks in the previous section clearly shows that the security filter is the most critical bottleneck. According to the stakeholders it has both the most severe and the most occurrences of unacceptable queue lengths, which is also supported by the KPI recordings. Furthermore, the customer surveys clearly show that the queue lengths at the security filter negatively influence the overall satisfaction score. Also, it is the only mandatory process step including a touchpoint that is accessible to theoretical models due to its clear queue structure. This research will therefore focus solely on enabling proactive flow management within the scope of EA's security filter.

Moreover, we believe that the findings that will be obtained when developing the necessary models at the security filter can almost directly be used for most of the other identified bottlenecks. A short-term passenger flow forecast model for the security filter is expected to be highly comparable to a short-term passenger flow forecast model for the self-service baggage drop-off units, assistance counters, check-in counters, airline service counters and the border control filter. Similarly, developed additional decision-support models are also expected to be equivalent and applicable to other bottlenecks. Only the gate congestion bottleneck would require extensive additional research. Concluding, we argue that solely resolving the research question for EAs security filter is adequate to accomplish the objectives that are set.

4.4 Conclusion

In this chapter, we have further detailed what the outbound passenger flow entails from a passenger and physical layout perspective. Consequently, seven different bottlenecks are identified that are eminent in the outbound passenger flow: unacceptable crowdedness at the departure gates and intolerable queue lengths at the self-service baggage drop-off units, assistance counters, check-in counters, airline service counters, security filter and border control counters.

Furthermore, in order to reduce the number and severity of these bottlenecks, we find that addressing the inadequate (reactive) flow management style by enabling a *proactive flow management style* is most relevant from both a scientific and business point of view. A proactive flow management style is seen as a style in which the main focus is on issue prevention so as to create a seamless passenger flow. We find that other options to address the practical goal like expanding infrastructure capacity, limiting growth or improving resource planning are either unfeasible from a business point of view or less interesting from a scientific point of view. We also find that, to enable a proactive flow management style, short-term passenger flow forecasts and additional decision-support need to be provided to the AOMs.

Finally, an additional scoping step is taken in which we consider that solely resolving the research question for EA's security filter is adequate to accomplish the objectives. The security filter is found to be the most critical bottleneck, causing both the highest number and most severe flow problems. Concluding, the upcoming chapters focus on enabling a proactive flow management style for EA's security filter by providing real-time short-term passenger flow forecasts and additional decision-support to the relevant stakeholders. These stakeholders can use this decision-support to proactively prevent issues and with that reduce the severity and number of experienced bottlenecks.

Chapter 5

Data understanding

This section addresses the data understanding phase of CRISP-DM framework and is concerned with identifying which data exists, where the data is located and identifying the quality and availability of the data. Firstly, every data source that falls within the scope of the outbound passenger flow at Eindhoven Airport and that has a relation to flow management is identified and listed. We purposefully use the scope of the whole outbound passenger flow opposed to focusing on the security filter during this first step, because this list can later be used by EA when addressing other parts of the outbound passenger flow. In the second step the smaller scope (see Section 4.3) is used and the list is evaluated in order to identify the data sources that are realistically accessible for this research project, are of acceptable quality and are relevant to the research problem. For each of these data sources, the available data and its quality are analyzed.

Methodology

In order to identify all the relevant data sources we use the obtained knowledge of the flow and its bottlenecks from the previous chapter, the expertise from the AOMs and more importantly the expertise of the airport's information manager (IM). The IM is responsible for most IT-projects within the airport and oversees the process from initial business problem to final IT solution implementation. Furthermore, the IM is responsible for the implementation of standardized data models and technologies. To illustrate, an example of one of the projects is the implementation of a standardized homogeneous IT communication environment in the form of an integration platform that sits between several software systems.

For every step in the passenger process (4.1.1) and for every bottleneck (4.2) an analysis is made as to which data sources are related. This analysis has been performed in cooperation with both AOMs and the IM in separate meetings to cross-check all obtained information. This finally leads to a complete and thorough list of data sources. In the next sections we analyze these data sources to determine which data sources are relevant and which sources contain accessible historic and/or real-time data. The findings of this analysis are summarized in Table 5.1. The table also contains a column that indicates the mode of connection of the three data sources that are ultimately used in this research project.

Data source	Mode of connection	LA	GFI	CI	BDO	AA	AS	LD	PSC	SC	HRF	AD	AC	B
AirDCS	Real-time data stream posted to API			**B**	**B**	H	H		**B**		I		I	I
iPort				I	I	I	I		I		I		I	I
VIBES					I									
SkyGuide	Scheduled calls to API		**B**								I			I
Parkbase	Scheduled direct database query	**B**												
MSCompact					H					H				
G4S Data										H				
PFM Intelligence								H				I		

- H: *Historic data available*, R: *Real-time data available*, B: *Both historic and real-time data available*, I: *Irrelevant data*
- LA: *Landside arrival*, GFI: *Get flight information*, CI: *Check-in*, BDO: *Baggage drop-off*, AS: *Additional assistance*, LD: *Landside dwell time*, PSC: *Pre-security scan*, SC: *Security check*, HRF: *HRF security check*, AD: *Airside dwell time*, AC: *Airline checks*, B: *Boarding*

Table 5.1: Data sources and their relation to the outbound passenger flow

5.1 Untapped data sources

The list of all identified data sources can be found in Appendix A. The list includes both data sources that are readily accessible for our purposes and data sources that can reasonably become accessible in the future through (limited) implementation of additional hardware and/or software. The data sources that are included in the latter category are either easy to implement using current Eindhoven Airport IT facilities or are already included in the airports planned IT projects. Two departure control systems (DCS) have been identified (AirDCS and iPort) that contain passenger information (e.g. booking references and baggage drop-off timestamps). A baggage handling system (BHS) called VIBES contains all baggage information (e.g. baggage destination and clearance scan timestamps). All flight information (e.g. expected arrival time and assigned gate) resides inside the flight information system (FIS) called SkyGuide. Parkbase contains all parking lot information (e.g. reservations and barrier motion timestamps). All x-ray scanners at the airport are controlled through the x-ray management system called MSCompact that for example contains information on x-ray timestamps and scan results. G4S, the security partner of EA, manually records the throughput of each security lane (passengers per 15 minutes) at the security filter by reading the throughput value from the metal detectors and inputting this on a self-designed web-app. Finally, the PFM Intelligence system uses sensors in a few toilet groups to count the number of people entering and leaving the groups.

5.1.1 Irrelevant sources

The complete list of data sources is evaluated in order to identify the sources that would ideally be accessible for this research project. We assume that, as in most prediction and process management projects, the usability and accurateness of the outcomes increases with the number of relevant and related data sources that are used. Thus, just two data sources from the complete list are identified as irrelevant for the ideal data source situation of this research project.

iPort

The airport uses two departure control systems, AirDCS and iPort. A DCS is essential in passenger processing since it is the central data source for all information related to individual passengers. The system for example handles information required for airport check-in, printing boarding cards, baggage acceptance, boarding and load control. Within Eindhoven Airport, AirDCS is used for over 97% of the departing passengers whilst the remaining passengers are handled by iPort. This rather peculiar distribution originates from TUI Netherlands and TUI fly Belgium, which are airlines that operates from Eindhoven Airport. TUI requires Eindhoven Airport to use iPort for their flights. However, since TUI accommodates only a very small percentage of the total amount of passengers and their flights have a particular seasonal trend (they primarily operate during summer), iPort is excluded from the list of ideally available data sources for the execution of this research project.

VIBES

Eindhoven Airport uses the VIBES software suite developed by Vanderlande to support the baggage handling operations for departing passengers. VIBES monitors and manages these operations, starting at bag-drop and ending after final reconciliation and loading on the aircraft. The only touch-point of this system with the passenger flow at Eindhoven Airport is the bag-drop/check-in process. During this process the gathered information is stored in both VIBES and the DCS. This information contains baggage information (e.g. destination, weight, route of baggage) as well as timestamps. Because of the overlap in information between VIBES and the DCS, we deem the baggage handling system to be irrelevant for the course of this research project.

5.1.2 Unavailable sources

One of the main aspects of the project is the development of a real-time passenger flow prediction model. Such a prediction model requires both historical data in order to train the model and real-time data for actual adoption of the model. The latter is usually more difficult to obtain since it often requires (considerable) changes to the data sources. Analysis of the six remaining data sources shows that for the course of this research project, only three will be realistically available. This is mainly due to the challenges and extensive timeliness that are often inherent to the development of real-time data streams. For the three data sources that will not be included into this project, the reasons behind the exclusion and an analysis of the impact on the project are provided.

MSCompact

The airport has several security lanes in place that each passenger has to go through. These security lanes consist of an x-ray scanner (for baggage and other personal items) and a metal detector (for the passengers themselves). MSCompact is the software suite that controls the x-ray scanners. This system is highly secure and contains encrypted information about the items that pass the scanners. Furthermore, the system runs in a fully isolated network in order to enhance security. This makes MSCompact unfit to serve as a real-time data source.

MSCompact could provide meaningful information on the throughput of the individual x-ray scanners, if the data was accessible. However, these potential throughput statistics would only provide information on the number of items that pass through the scanner. Information on the number of passengers that flow through the security lane is more useful (see G4S Data). Furthermore, the throughput of a scanner is heavily influenced by the number of items that a passenger carries, the number of items that get re-scanned and the speed of the operator of the scanner. Therefore, although no actual data has been observed, we assume that the information from MSCompact will be dispensable.

G4S Data

As indicated, the security lanes at Eindhoven Airport consist of x-ray scanners and metal detectors. The latter are not centrally managed by a software application but instead are standalone instruments. The metal detectors do however have small displays that show a counter. This counter indicates how many people have passed through the metal detector since the last time the counter was set to zero. Employees of G4S are required to read these displays every 15 minutes and input the values into a specifically designed web-app. These values can then be used to calculate the throughput and evaluate if G4S complies to the contracted service levels. Historical data from this system is available. However, G4S, is currently not willing to invest in the system so that the data becomes available in real-time. This is mainly due to planned changes in the hardware and software environment of G4S that would render the current web-app obsolete. Therefore, the information from the metal detector logs are not available for the course of this project.

Real-time information from the logs could provide insights in the throughput of the last time-intervals. Furthermore, with prediction models, the information could be used to forecast the throughput of upcoming periods. The ability to do so and the quality of the predictions depends heavily on the quality of the data. To assess this quality, logs of the year 2017 are analyzed. We find that over 10% of the entries contain values that are out of realistic bounds and are thus a definite entry error. Furthermore, fluctuations in throughput between succeeding time intervals (15 minutes) is found to be improbably high. Although not thoroughly evaluated, these first statistics indicate that the G4S data is prone to human error and contains high fluctuations that make the data quality questionable and thus not completely essential for this research project.

Further research is required to determine if the throughput information at the security lanes is accurate enough to be useful in case G4S decides to set-up a real-time data stream.

PFM Intelligence

In order to increase sanitation cleanliness across the airport, some toilet clusters are equipped with people counting devices. The information from these devices is centrally collected and stored to support the contracted cleaning company CSU in determining when to actuate a new cleaning job. The PFM Intelligence information is very basic and provides insights in the number of toilet visits in a certain time frame (since last time cleaning). Although the data is available for review by CSU employees in real-time, it is not directly available in real-time for the course of this project. The sensors are connected to a central logger through an analogue interface. Therefore, extensive hardware and software changes need to be made to remotely access the real-time information over the internet. This is an investment that Eindhoven Airport is not currently willing to make. Furthermore, the land-side area of the terminal (before the security filter) is open to visitors (non-passengers). These visitors for example work in the area or come to pick-up an inbound passenger and make use of the shops/restaurants in the terminal. These visitors are expected to cloud the people count numbers at the toilet groups, rendering them less useful for passenger flow forecasting.

5.2 Used data sources

The above analysis leaves three available sources for this research project: AirDCS, SkyGuide and Parkbase, which we consider to also be the most important. A detailed description of the available data in those systems is provided, as well as an analysis of the quality of the data.

5.2.1 AirDCS

AirDCS is the airport's main departure control system. It is managed, maintained and primarily used by Viggo: the handler that Eindhoven Airport has contracted to perform most of its operational activities. Furthermore, AirDCS is split into a passenger management component and an operational flight management component. The latter is for example used for load control (ensuring that cargo loads are evenly distributed in the aircraft) and aircraft checks management. We focus solely on the passenger management component since the flight component is considered to be outside of our scope. Furthermore, all flight time related information will be extracted from SkyGuide, since for that type of information SkyGuide is the single point of truth within the organization.

The passenger management component of the DCS manages all information related to the passenger process across the airport. It for example includes the management of airport check-in, printing boarding cards, baggage acceptance and boarding. It enables staff members to manage the check-in process, collect the passengers' information, choose seats, generate and issue IATA standard boarding passes and baggage tags, and generate the manifest after the check-in. The DCS uses service oriented architecture including database management that allows for consistent views of passenger information.

Airlines send a Passenger-Name List (PNL) to Eindhoven Airport at least three hours before the scheduled departure time of a flight, which is then uploaded into AirDCS. A PNL contains all the relevant information for the passengers on a flight (e.g. name, baggage allowance, required assistance). From this moment of initial creation, all actions that are taken for a specific passenger are recorded in the DCS. Examples include the drop-off of their baggage, a change in the booked seat number and the final scan of their boarding card before embarking the aircraft.

Log files are maintained and stored for 6 months of every event that takes place within AirDCS. The log files contain a lot of information, such as staff member log-in events and passenger boarding scan events. A log file that covers one month of data usually contains over 2 million events. Logs are available from the 14th of March 2018 to the 17th of December 2018 for this analysis. The following information is available for every event:

Identifier:	Event identifier and timestamp of the event
Location:	Machine that initiated the event (name of machine, IP address and which button was clicked)
Type:	Type of the event (e.g. *pre_security_in_through_egate* event)
Value:	In case a change was made, the old and new value are provided (e.g. seat change)
Passenger info:	In case the event concerns a passenger, the passenger details are provided (passenger identifier and booking reference)
Flight info:	In case the event concerns a passenger and/or a flight, the flight details are provided (flight identifier and public flight date)
Baggage info:	In case the event also concerns a baggage item, the baggage identifier is provided (e.g. baggage drop-off)

Over 200 types of events have been identified, most of which we deem useless for our goals: some events have nothing to do with the outbound passenger flow (e.g. staff log-in), some events take place after the security filter (e.g. boarding scans) and others have such a low occurrence that they are not expected to be of any explanatory value in the forecast (e.g. seat change). Therefore, only four event types are identified to be interesting for our models: passenger creation, passenger baggage drop-off, passenger check-in, and passenger pre-security filter scan.

The log files are however always at least one day old and are not accessible in real-time. Therefore, the French manufacturer of AirDCS called Tisys was requested to set up a real-time event stream that posts these four event types to an API. This API in turn logs the events instantaneously into a database. Since this required substantial changes in their system, the real-time event stream was only set up on the 17th of December 2018, nearing the end of this research project. This means that analysis on the completeness and accuracy of the data is merely performed on the log files (until the 17th of December) and the real-time event stream is assumed to have the same quality. Analysis concerning the timeliness of the data is however performed on the real-time event stream. Timeliness indicates what the time difference is between data capture and the real world event being captured. We find that the maximum time difference that is found is equal to 10 seconds, which we deem sufficient for our purposes. The completeness and accuracy of the four identified event types is now analyzed by using the log files.

Passenger creation events

The passenger creation event is triggered when AirDCS receives the PNL from the airline. This is the first time that the system receives information about a passenger. Although the passenger creation events are identified as required event types, they are not actually recorded in the historical log files. However, the real-time stream does include these events. The passenger creation events will be merely used to compute the actual number of passengers that are expected to depart on a flight (see 6.1.3). Since we can extract the actual number of passengers that departed on a historical flight from SkyGuide, we are able to construct artificial passenger creation events to complement the historical log files. For every flight with a mention in the historical log files, artificial events are created with a timestamp 3 hours before that flights public departure time. The number of passenger creation events that need to be created per flight is set equal to the actual number of passengers that departed on that flight (extracted from SkyGuide). This is expected to closely match reality: only no-show passengers that would have had a passenger creation event if they were logged will not have one now, since they are not in the actual passenger count.

Passenger baggage drop-off

The passenger baggage drop-off events are recorded for all passengers that drop-off one or more pieces of baggage at the SSBD units or at one of the assistance or check-in counters. We find that there are two subsequent days without any baggage drop-off events (1st and 2nd of April 2018). Further investigation shows that this is not an anomaly due to operational problems and are a definite data error. The other days do not show any anomalies and follow a stable pattern. The data completeness is checked against data from VIBES, the baggage handling system from Vanderlande that processes all baggage that is dropped off. Computing the tallies of baggage drop-off events and comparing them to the counts of processed bags by VIBES results in the conclusion that every bag that is dropped also has an associated event. Apart from some small anomalies (e.g. a difference of two bags on a whole day), there are no major differences.

Passenger check-in

The passenger check-in events are available for all passengers that check-in at one of the check-in counters. Similar to the baggage drop-off events, we have no events for the same two days (1st and 2nd of April 2018). To check if all passengers that check-in on the airport also have an event, we cross-check the percentage of departing passengers that have a check-in event per airline with the percentages provided by these airlines. These are similar and we thus conclude that the check-in event logs are complete, except for the two missing days.

Passenger pre-security filter scan

The passenger pre-security filter scan events are recorded for all the passengers that scan their boarding card at one of the four self-service e-gates or at the manual scan counter. Every departing passenger is expected to have one pre-security filter scan event. However, during first analysis we find that there are way more pre-security filter scans than departing passengers. We find that three out of four self-service e-gates log a scan twice, approximately 5 seconds apart. Data cleaning is performed to delete the duplicates and only keep the first events. We then again compare the numbers of actually departed passengers (from SkyGuide) with the numbers of pre-security filter scans. The vast majority of days do not show any significant anomalies (average differences of just over 3 passengers per day). However, there are a few days with more pre-security filter scans than actually departed passengers. Further analysis shows that this is due to last-minute cancellation of flights: these passengers already passed pre-security but left the airport through the exit door instead of an aircraft. Since these events are still important from a forecasting point of view (the passengers actually went through the security filter), they are maintained in the data set. We conclude that all passenger pre-security filter scans are complete and accurate in the historical logs from AirDCS.

5.2.2 SkyGuide

SkyGuide is Eindhoven Airport's flight information system. It is the source system for all flight related information. On a flight level, the system for example provides information about planned, expected and actual arrival/departure times. Other information for example includes aircraft type, parking position on the platform, flight status and assigned luggage belt in the arrival hall. After departure or arrival of a flight, the actual number of passengers is recorded in SkyGuide. These numbers, among others, are then used to send invoices directly to the airlines. Furthermore, it also serves as a data source for external partners like NS (Dutch Railways) and EA's flight information display system (FIDS), that in turn use the information to supply factual current flight information on websites and digital displays. This all indicates why SkyGuide serves as the single point of truth for flight related information.

SkyGuide will be used to determine which, and more specifically how many, flights are departing in the near future. This data is extracted from SkyGuide in real-time and will always have a high data quality, with a low timeliness and high accuracy and completeness. This data is provided through an API that is part of an integration platform that is connected to SkyGuide. We do not provide an overview of the available data in SkyGuide since it does not concern event data (as in AirDCS and Parkbase) and since the number of data features in SkyGuide is huge and spread over several databases. Furthermore, we merely use SkyGuide to obtain the number of flights that are expected to depart in the near future (see Chapter 6). This makes it unreasonable and irrelevant to provide a complete overview of available data in this section.

5.2.3 Parkbase

Parkbase is the software platform that is used to manage the different car parks that are owned by the airport. The airport has one short-term car-park (*P1*) and five long-term car-parks that are located in close proximity to the airport terminal. Passengers who desire to park their car before departure can make online reservations for the car-parks or try to find an available spot on arrival at the terminal. All vehicles need to enter and exit the car-parks through electronic barriers and every barrier movement is recorded in real-time within Parkbase. Public transport buses that arrive at the airport terminal all drive through a designated barrier at the short-term car-park *P1*. The associated barrier movements are also captured in real-time within Parkbase.

The data that resides in Parkbase is stored across multiple databases (e.g. reservation database, barrier movement database) and needs to be linked together in order to be useful to us. From all the data that is available within Parkbase, for our purposes we identify the following information to be required to exist of each barrier movement:

Identifier:	Barrier movement identifier and timestamp of the movement
Location:	Barrier that initiated the event (barrier identifier and car-park identifier)
Type:	Movement type (e.g. reservation, staff member, bus or regular)

The movement type indicates on what grounds the vehicle is allowed to pass through that barrier. This can for example be a reservation, staff member subscription (free parking), public transport bus or a regular parking vehicle that arrives without a reservation. We only focus on bus entries, reservation entries and regular parking entries since the others (e.g. staff member subscription) are not interesting from a passenger flow forecasting point of view. A system is developed that executes queries directly on the Parkbase databases and is able to retrieve this information for a predetermined time frame. The movements are recorded in the Parkbase databases almost instantaneously and the query to retrieve for example the movements of the last 15 minutes takes a maximum of 10 seconds. We therefore consider the timeliness to be adequate for our purposes. The completeness and accurateness of the data is now considered.

Movement type: bus

Eindhoven Airport can only be reached by a single mode of public transport: buses. Buses are therefore an important mode of transport for EA's passengers. There are two dedicated routes between Eindhoven central station and Eindhoven Airport (routes 400 and 401), one including intermediate stops and one non-stop transfer. These bus routes have a high frequency and alternate every 5 minutes. Next to these two there is another bus route (20), that visits nearby villages and arrives at the airport every half hour. These bus lines have predetermined schedules that are the same for a full year. The arrival logs of the buses in Parkbase should therefore closely match these schedules. Analysis shows that on most days, the number of buses that are logged within each hour closely match the number of buses that were expected to arrive in that hour. We do see a consistent slight discrepancy during early morning and late evening (start and end of the schedule) between the logs and the schedule, in which there are more logs than actually planned buses. Further investigation shows that this is due to empty buses arriving at the terminal to start their shift from the airport. During normal operation (during the day), these buses will arrive with passengers and also leave with passengers. Since less than 3% of the buses during a day are found to be starting their shift from the airport, this is assumed to not be a significant data inaccuracy. Furthermore, Parkbase is analyzed to have no missing data: we find that there are three days with a significant lower amount of buses than were scheduled, but further research shows that during these days the bus drivers went on strike.

Movement type: subscription

Passengers are able to reserve a parking spot on EA's website. These reservations are stored in Parkbase and in a separate sales system. To check for data accuracy and completeness, the number of reservations for a certain day in the sales system should match with the number of reservation barrier movements in Parkbase. Analysis shows that for most days this is however not the case: there are more reservations than reservation barrier movements. Further investigation shows that a significant amount of passengers, although they have a reserved spot, still get a regular entry ticket and are thus registered as such. These passengers have to pay this ticket on exit of the car-park and are able to get a refund of their reservation. Except for a few no-shows, adding the number of refunds to the barrier movements does result in similar numbers when compared to the sales reservation system. We can thus assume that all vehicles with a reservation and that register themselves as such at entry have a corresponding registered barrier movement in Parkbase. Since we sum the number of regular and reservation entries in our forecast model, the vehicles with a reservation that register themselves incorrectly as a regular entry do not have a negative effect.

Movement type: regular

No comparison statistic is available for regular entries (Parkbase is the sole data source), however by using the using the analysis of the bus and reservation entries we can reasonably assume that the data quality of the regular entry logs is similar and thus adequate. We thus conclude that all data in Parkbase is complete and accurate.

5.3 Conclusion

This chapter focused on identifying the current situation regarding (real-time) data within the context of the outbound passenger flow at Eindhoven Airport. Nine data sources have been identified that are related to this flow. Of the nine identified sources, three are deemed irrelevant for the specific goals of this research project. Another three sources are unavailable for the course of the project due to our real-time data requirement. For these sources, the challenges and extensive timeliness that are inherent to the development of real-time data streams cannot be overcome. We are finally able to use three real-time data sources: AirDCS is the departure control systems that contains passenger information, SkyGuide is the flight information system and Parkbase contains all parking lot information. We find that it is inherently difficult to set-up real-time data streams for systems that are not designed for this and we were thus required to put great effort in setting up these streams for the three data sources. For example for AirDCS, we had to get the French manufacturer of the software involved since establishing a real-time data stream required substantial changes to the system.

Subsequently, data sets from the three data sources are analyzed to determine the quality of the data. We find that all three data sources score high on timeliness, completeness and accuracy of the data. The results from this chapter are used in the next chapters during model development.

Chapter 6

Data preparation

In this chapter, the *data preparation* phase of the CRISP-DM framework is addressed. The data understanding from the previous chapter and the business understanding from Chapter 4 will be used to select the data features that will be used as input and output in the forecasting model. The model aims to forecast the short-term passenger flow at EA's security filter, and is thus concerned with time-series. A time-series data set will therefore be generated in Section 6.2. Furthermore, the type of models that are be developed in the next chapter (Chapter 7) are already considered during this data preparation phase. As found in Chapter 2, we will mainly develop and train recurrent neural networks which influences some data preparation decisions.

6.1 Data selection

The first step in the data preparation phase is the data selection step. The data selection step in the CRISP-DM framework is concerned with deciding on the data features that will be used in the forecasting model. In this section a rationale for the selection of the data features is provided. The result of the data selection step is a set of data features that are relevant to the forecasting model. A total of 14 predictor data features is found, divided over four categories. These features are used to predict a single outcome feature. The data features are summarized in table 6.1.

6.1.1 Time-interval

The model's goal is to forecast the short-term passenger flow at Eindhoven Airport's security filter. In other words, it aims to predict the amount of passengers that will arrive at the security filter within the near future. In discussions with the stakeholders we identified that looking three hours ahead will suffice to enable a pro-active flow management style. The three hours are split into 12 time intervals of an equal predetermined length of 15 minutes, resulting in all data features being aggregated to 15 minutes. The chosen interval length is aligned with the way that Eindhoven Airport schedules its security lane resources: they decide on the number of lanes and operators for every 15 minutes during a day.

6.1.2 Outcome variable

The most important data feature in the forecasting model is the outcome (dependant) variable. Since the model aims to forecast the aggregate number of passengers that will arrive at the security filter within time-intervals of 15 minutes, this is the outcome variable. As discussed in Section 4.2.5, passengers are required to scan their boarding-card at the pre-security filter when arriving at the security filter. These pre-security filter scan events are captured within AirDCS in real-time. We consider these events to be the moment that a passenger arrives at the security filter and can therefore be used as the dependant variable. This variable is constructed by aggregating the scans to 15 minutes and is named *PreSecurityFilterScanIn*. A week-long example (Monday, October 1 to Sunday, October 7 2018) is depicted in Figure 6.1. This figure shows the 15 minute aggregate number of passengers that have scanned their boarding pass at the pre-security filter for every 15 minutes during a whole week. The periods with no passengers

Type	Name	Explanation
Calendar	*SecurityFilterOpen*	Security filter closed or open (value 1 or 2 respectively)
	HourOfDay	Hour of day of the interval (value between 0 and 23)
	DayOfWeek	Day of week of the interval (value between 1 and 7)
Passenger	***PreSecurityFilterScanIn****	Number of passenger pre-security filter scans in interval
	BaggageDropOff	Number of baggage drop-offs in interval
	PassengerCheckIn	Number of passenger check-ins in interval
Transport	*P1EntriesCar*	Number of car entries at car park P1 in interval
	P1EntriesBus	Number of bus entries at car park P1 in interval
	POtherEntriesCar	Number of car entries at other car parks in interval
Flight	*ExpectedPaxHour1*	Expected number of missing passengers on flights in hour 0-1 from interval
	ExpectedPaxHour2	Expected number of missing passengers on flights in hour 1-2 from interval
	ExpectedPaxHour3	Expected number of missing passengers on flights in hour 2-3 from interval
	ExpectedFlightsHour4	Expected number of flights in hour 3-4 from interval
	ExpectedFlightsHour5	Expected number of flights in hour 4-5 from interval

* Predictor variable *PreSecurityFilterScanIn* also serves as the outcome variable

Table 6.1: Data features

are the nightly hours during which the airport and thus the security filter is closed. As can be clearly seen, the daily pattern is fairly noisy with several peaks during the day. These peaks clearly indicate the need for accurate short-term forecasting, since an unexpected peak during a low capacity situation can lead to bottlenecks.

6.1.3 Predictor variables

As found in Chapter 5, three different related data sources can be considered when determining data features: AirDCS, Parkbase and SkyGuide. A lot of data resides inside these systems, so an elaborate selection has to be made as to which data to use. This requires the involvement of domain experts and serves as an opportunity to incorporate their knowledge into the data (Guyon and De, 2003). The domain experts at Eindhoven Airport are the Airport Operations Managers. Through several interviews a set of data features has been constructed. These data features are expected to be associated with the short-term expected number of passengers that arrive at the security filter.

Pre-security filter scan in

Naturally, the historical values of the outcome variable (*PreSecurityFilterScanIn*) are also used as a data feature in the forecasting model.

Calendar features

The security filter is closed between late evening and early morning (approximately from 21.00 to 05.00). To represent this, a discrete data feature is introduced called *SecurityFilterOpen*. This data feature either has value 1 (security filter is closed) or value 2 (security filter is open). The model can use this data feature to learn when the outcome variable should be zero.

According to the AOM's, the passenger flow dynamics differ during the day. A difference can for example be found in the time window that passengers incorporate between arrival at the airport and departure of their flight, which tends to be shorter for early flights. To capture

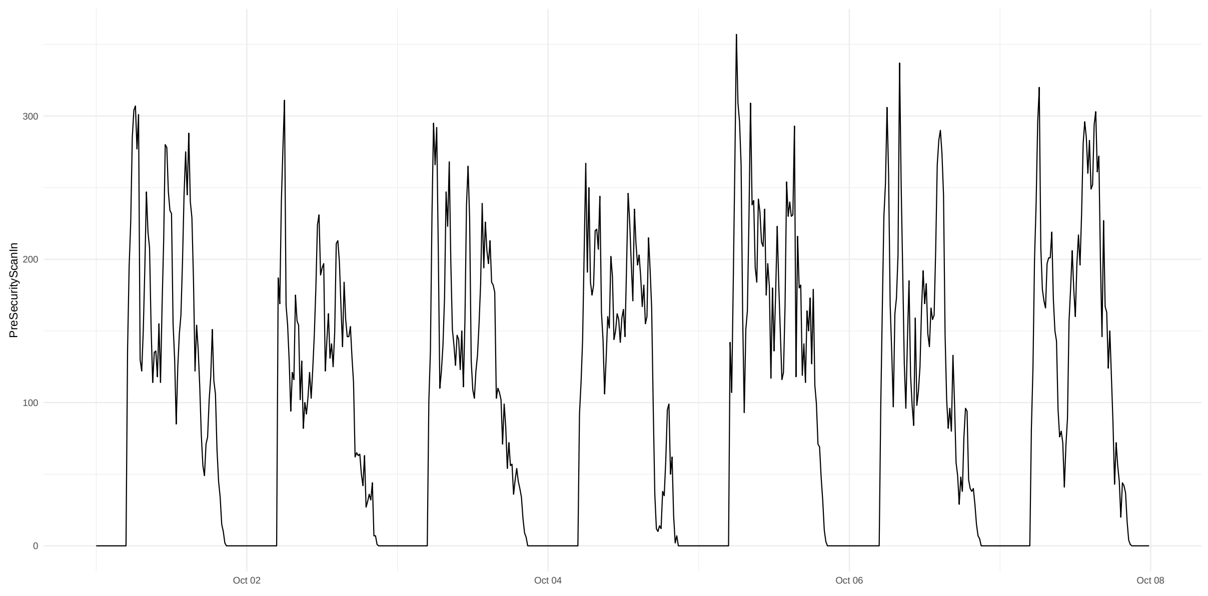


Figure 6.1: One week of 15 minute aggregated pre-security filter scan events

the flow dynamics over the day, the hour of day of the interval is introduced as data feature (*HourOfDay*), with a value between 0 and 23.

Similar to HourOfDay feature, according to the stakeholders, the dynamics between days of the week differ. These dynamics can be learned by the network through the *DayOfWeek* feature. This data feature has a value between 1 and 7, where 1 indicates that the interval is on a Monday and 7 indicates that the interval is on a Sunday.

Passenger features

Prior to arriving at the security filter, some passengers drop off their baggage at one of the drop-off locations. According to the AOM's, it is believed to be an influential predictor for the PreSecurityFilterScanIn outcome feature, since passengers that drop-of baggage will per definition pass through the security filter somewhere in the near future. The aggregate number of passengers that drop-off a piece of luggage during the 15 minute time intervals is therefore introduced as the *BaggageDropOff* data feature. This feature is computed through real-time event logs originating from AirDCS.

Some passengers did not check-in online for their flight and require to be manually checked-in at a check-in desk. Similar to the BaggageDropOff event, these events are logged in real-time in AirDCS and are also believed to be an influential predictor for the model's outcome variable. The aggregate check-in events are introduced as the *PassengerCheckIn* feature.

Transport features

Prior to entering the airport terminal, passengers arrive at the airport through different modes of transport. According to the latest customer survey (MarktEffect, 2018), most passengers are dropped-off by car (35%), followed by arriving by bus (30%) and parking at one of the airport's car parks (22%). The remaining passengers park their car at a car park that is not owned by the airport (10%) or arrive through some other mode of transportation. In order to capture these arrival features, three different data features are computed using the real-time data that is stored within Parkbase. These data features capture the three biggest groups of arrival transport categorizes (drop-off, bus and EA car park), totaling almost 90% of the passengers.

Most passengers arrive at the airport by being dropped off by friends/family. The airport has a designated drop-off (kiss & ride) location that allows cars to enter free of charge for a total of 15 minutes. These cars enter the drop-off area through the barriers associated with the short-term car park *P1*. Using the real-time movement logs of these barriers, the data feature *P1EntriesCar* can be computed. This feature captures the number of cars that have arrived at the drop-off area within the 15 minute time intervals.

Buses that arrive at the airport terminal all drive through a designated bus barrier at short-term car park *P1*. The associated barrier movements are also captured in real-time within Parkbase. Using these movement logs, the data feature *P1EntriesBus* is computed, indicating the number of buses that have arrived at the airport within the 15 minute time intervals.

Next to the short-term *P1* car park, the airport owns five long-term car parks that are located in close proximity to the airport terminal. Entry barrier movements of these car park are aggregated and represented through the *POtherEntriesCar*. Walking distance from the car parks to the terminal is believed to be comparable to each other, justifying the aggregation of all car parks' barrier movements in one data feature in order to minimize model complexity.

Flight features

A big advantage that an airport has over the public transport context studied in Chapter 2, lies in the information that the airport has concerning departing/arriving flights and their passengers. This information for example concerns the planned and expected departure times of flights and the number of expected passengers on a flight. This information can serve as an important predictor for the short-term passenger flow. To capture the information, two types of data features are introduced: *ExpectedPax* and *ExpectedFlights*.

Airlines send a Passenger-Name List (PNL) to Eindhoven Airport at least three hours before the scheduled departure time of a flight, which is then uploaded into AirDCS. A PNL contains all the relevant information for the passengers on a flight. More importantly, this is the first time that EA receives exact information on the number of passengers that will arrive at the airport for a certain flight. Before receiving the PNL, EA only has information on the maximum number of passengers on a flight (number of seats on the aircraft) which hardly represents the actual number. The number of passengers on each flight is constructed through the aggregation of all passenger creation events (see Section 5.2.1) for a flight based on the events' flight identifier. This information regarding the number of passengers on a flight is combined with the pre-security filter scan events to form the *ExpectedPax* data type feature. The pre-security filter scan events also contain a flight identifier, thus enabling the matching of the number of passengers and the number of pre-security filter scan events per flight. Three data features of this type are computed: *ExpectedPaxHour1*, *ExpectedPaxHour2* and *ExpectedPaxHour3*. The first feature represents the number of passengers that did not yet pass the pre-security filter and are scheduled on a flight with a departure time between 0 and 1 hours from the current interval time. The second and third feature represent the same number but for departure time intervals 1-2 and 2-3 hours from the current interval respectively. To illustrate, if the current time is 12.00, ExpectedPaxHour2 represents the number of passengers that are on a flight that is scheduled to depart between 13.00 and 14.00 and did not yet pass the pre-security filter.

Since Eindhoven Airport receives the majority of the PNLs on the three hour deadline time, information concerning the number of passengers that are on flights that are scheduled later than three hours from the current time interval is unavailable. Since the aim of the model is to forecast 12 time intervals ahead (3 hours), an additional data feature needs to be incorporated that the model can use to better learn to forecast values of the last time intervals. To elaborate, the passengers that will arrive in these last time intervals will likely be on flights that are scheduled to depart later than 3 hours in the future. Therefore, the data features *ExpectedFlightsHour4* and *ExpectedFlightsHour5* are introduced. The first feature represents the number of flights that are expected to depart between 3 and 4 hours from the current interval time. The second feature is similar but for the interval of 4 and 5 hours from the current interval time. This information is obtained through SkyGuide. Although the model does not know how many passengers are on those flights, it can use these two features that represent the expected number of flights to learn a general sense of crowdedness. Although the information regarding the maximum capacity of a flight is available, we now argue why we deliberately choose to use the number of flights as a feature opposed to the aggregation of this maximum capacity. EA is only visited by similar sized aircraft that have an average capacity of 190 passengers. When choosing the maximum capacity as a feature we would indirectly make the assumption that the number of passengers of a flight is related to this capacity without thorough testing. Furthermore, we believe that, because the capacity of the aircraft that visit EA is very similar, the number of aircraft opposed to the maximum capacity represents a more accurate data feature. It must be noted that for other airports (e.g. Amsterdam Airport Schiphol) this assumption probably does not hold due to the large variety in aircraft capacities.

6.2 Data generation

From the three data sources (AirDCS, Parkbase and SkyGuide), we find that the earliest date at which data is available and complete for all data sources is 3 April 2018. Furthermore, for training and validation of this model, data is available and complete up till and including 17 December 2018. The 14 data features are computed for all 15 minute time intervals between these two dates, resulting in a total of 24.864 rows (259 days). An example subset (9 October 2018) can be found in Table 6.2. Furthermore, general properties of the complete data set are provided in Table 6.3. We find that there is no missing data (*No. NA* in table) and there are no unexpected outliers. The standard deviation of most data features is high compared to the averages, which we consider to be a result of the high fluctuations in passenger numbers across the day.

The complete data set has to be split into a training set *TR* and test set *TE*. The test set is merely used after model development to test the model's predictive performance on a data set that it has never seen before. For training of the RNNs, the training set has to be further split into a training subset *TRS* and validation subset *TRV*. The model is trained on the training subsets, and the parameters that minimize error on the validation sets are chosen. Nested cross-validation will be used to prevent over-fitting and provide more robust model performance statistics. To do so, three different splits are constructed (*S1*, *S2* and *S3*). The first split consists of *TR1* with the first 40% of the complete data set and *TE1* that has the subsequent 20% of the data. *TR1* is further split into *TRS1* and *TRV1* with an 80/20 split ratio. The same logic is used for *S2*: *TR2* contains the first 60% of the complete data set and *TE2* that has the subsequent 20% of the data. Finally, *S3* consists of *TR3* that contains the first 80% of the complete data set and *TE3* that has the 20% of the rest of the data. An illustration of the splits is provided in Figure 6.2. The error on each of the three splits is averaged in order to compute a robust estimate of model error.

6.3 Conclusion

In this chapter, using the analysis of data understanding and business understanding of the previous chapters, fourteen input data features are identified that are expected to be associated with the short-term expected number of passengers that arrive at the security filter. Furthermore, in discussions with the stakeholders we identified that looking three hours ahead will suffice to enable a pro-active flow management style. The three hours are split into 12 time intervals of an equal predetermined length of 15 minutes, resulting in all data features being aggregated to 15 minutes. The fourteen data features are either calendar related (e.g. hour of day), passenger related (e.g. number of baggage items dropped off), transport related (e.g. number of car entries at car parks) or flight related (e.g. number of departing flights).

Furthermore, since we aim to forecast the number of passengers that will arrive at the security filter, a related outcome variable needs to be identified. Analysis finds that passengers are required to scan their boarding card at the pre-security filter prior to arriving at the security filter. These scan events are captured within AirDCS in real-time and are considered to be the moment that a passenger arrives at the security filter and are therefore used as the outcome variable. To sum up, the forecast models that will be constructed in the upcoming chapter will use fourteen input features to forecast the number of pre-security filter scan events in the 12 upcoming 15 minute time steps.

Start	End	**PSSI**	SFO	HOD	DOW	BDO	PCI	P1EC	P1EB	POEC	EPH1	EPH2	EPH3	EFH4	EFH5
06:15	06:30	**289**	2	6	2	97	56	39	1	17	41	255	269	4	4
06:30	06:45	**155**	2	6	2	35	23	29	3	21	68	159	479	2	3
06:45	07:00	**148**	2	6	2	31	26	21	2	15	28	35	783	2	2
07:00	07:15	**84**	2	7	2	25	10	23	3	19	20	230	693	3	2
07:15	07:30	**107**	2	7	2	31	5	31	4	24	15	176	655	4	2
07:30	07:45	**106**	2	7	2	33	2	20	6	13	5	298	332	3	3
07:45	08:00	**145**	2	7	2	38	3	27	3	15	2	439	314	2	5
08:00	08:15	**159**	2	8	2	43	6	37	5	12	23	408	462	2	5

PSSI: ***PreSecurityFilterScanIn***, SFO: *SecurityFilterOpen*, HOD: *HourOfDay*, DOW: *DayOfWeek*, BDO: *BaggageDropOff*, PCI: *PassengerCheckIn*, P1EC: *P1EntriesCar*, P1EB: *P1EntriesBus*, POEC: *POtherEntriesCar*, EPH: *ExpectedPaxHour*, EFH: *ExpectedFlightsHour*

Table 6.2: Example subset of data-features

	PSSI	SFO	HOD	DOW	BDO	PCI	P1EC	P1EB	POEC	EPH1	EPH2	EPH3	EFH4	EFH5
No. observations	**24.684**	24.684	24.684	24.684	24.684	24.684	24.684	24.684	24.684	24.684	24.684	24.684	24.684	24.684
No. NA	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0
Min. value	**0**	1	0	1	0	0	0	0	0	0	0	0	0	0
Max. value	**431**	2	23	7	215	147	102	15	56	871	1117	1702	10	10
Average	**92.3**	1.7	11.5	4.0	32.9	10.8	34.2	2.2	11.8	25.1	183.7	333.8	2.0	2.0
Standard deviation	**90.9**	0.5	6.9	2.0	37.1	15.6	24.1	2.0	10.8	49.8	203.6	350.6	2.1	2.1

PSSI: ***PreSecurityFilterScanIn***, SFO: *SecurityFilterOpen*, HOD: *HourOfDay*, DOW: *DayOfWeek*, BDO: *BaggageDropOff*, PCI: *PassengerCheckIn*, P1EC: *P1EntriesCar*, P1EB: *P1EntriesBus*, POEC: *POtherEntriesCar*, EPH: *ExpectedPaxHour*, EFH: *ExpectedFlightsHour*

Table 6.3: Data set properties

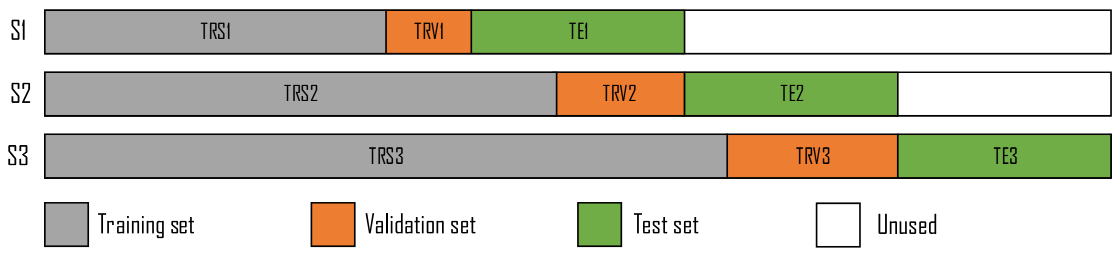


Figure 6.2: Data split

Chapter 7

Forecast modelling

This chapter describes the development of a forecasting model that aims to forecast the 12 step ahead aggregate number of passengers that will arrive at the security filter at EA within time-intervals of 15 minutes. Four different types of models are tested and compared: a direct historical average model, a simple naive model, a statistical regression model, a simple machine learning method and four variants of a recurrent neural network model. Based on our analysis of Chapter 2, the latter models are expected to have the best performance. The other models are used as a comparison and baseline of the newly developed models so as to establish an understanding of their relative performance. Firstly, the evaluation metrics that are used to compare the models are introduced. Secondly, a detailed explanation of the models is provided and subsequently the models are compared numerically. The chapter ends with a description of the lessons that were learned during model development and the scientific contribution of this paper.

7.1 Evaluation metrics

The performance of the models is primarily measured through their root mean squared error (RMSE), which is the standard deviation of the prediction errors. This is a well known method and commonly used when comparing regression prediction models (Twomey and Smith, 1995). Considering the RNNs, the RMSE is also used as the Loss function that the gradient descent optimization algorithm tries to minimize during training (Section 7.3.2). Next to the RMSE, the mean absolute error (MAE) is also computed to get a better understanding of the differences. The corresponding formulas are:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2} \qquad\qquad MAE = \frac{1}{n}\sum_{t=1}^{n} |e_t|$$

In which e_t is the difference between the actual value and the forecasted value at time t and n is the number of observations.

7.2 Baseline models

7.2.1 Historical average (HA)

In this approach, the predictions of the future values is equal to the mean of the past data during the corresponding time-step on similar weekdays. To illustrate, the prediction for Monday 10.00-10.15 is equal to the average of all historical values on previous Mondays during 10.00-10.15.

7.2.2 Last observation carried forward (LOCF)

The naive method in which the last observation is used as the forecast for the upcoming periods is called last observation carried forward. LOCF and HA are very basic methods that are not expected to result in high quality predictions but serve as a first baseline.

7.2.3 Vector auto-regression (VAR)

Vector auto-regression model is an extension of uni-variate auto-regression (AR) models to multivariate time series data, and was first introduced in Sims (1980). It has since been widely used in time-series forecasting. A VAR model is a multi-equation system where all the variables are treated as endogenous (dependent). The main idea is that the value of a variable at a time point depends linearly on the value of different variables at previous instants of time.

7.2.4 Random forest (RF)

Random Forest is a widely used supervised learning algorithm that can be used for both classification and regression tasks. It was first introduced in Breiman (2001) as a means to overcome overfitting and reducing variance. Random Forests do this by building multiple decision trees and merging them together to get a more accurate and stable prediction.

7.3 RNN models

In this section, four different RNNs are introduced. Firstly, the layout of the RNNs is provided. Secondly, the different hyper-parameters (HPs) that need to be tuned in the models to optimize their performance are introduced. Subsequently, the choice for Bayesian optimization to identify the correct value of the HPs is explained. Finally, the training algorithm that is used to train the models is provided.

7.3.1 RNN layouts

Four different model layouts are tested: two basic RNNs and two encoder-decoder RNNs. The models are multi-input multi-output (MIMO) models that preserve the stochastic dependencies between predicted values and that have been successfully applied to several real-world multi-step ahead time series forecasting tasks (e.g. Bontempi et al., 2012; Yanjie et al., 2017). These type of models have been proven to outperform direct and recursive methods. The direct method involves developing a separate model for each forecast time step and the recursive method involves using an one-step model multiple times where the prediction for the prior time step is used as an input for making a prediction on the following time step (Ben Taieb et al., 2012).

Basic GRU (B-GRU) and Basic LSTM (B-LSTM)

The first RNN model that is employed is a simple RNN with 1 input layer, 1-3 hidden layer(s) consisting of GRU units and 1 output layer. The second RNN model is identical, but replaces the GRU units with LSTM units. A total of 14 input features have been defined in Chapter 6, resulting in 14 units in the input layer. Furthermore, we have determined that a forecast of 12 steps ahead is desired, resulting in 12 units in the output layer. The number of units in the hidden layer(s) is one of the HPs and will have a value between 14 and 672 (see Section 7.3.2.

However, by using a simple RNN structure we demand the hidden layers to be capable of both memorizing important events of the past and using these events to predict future values. These tasks can be split by using an encoder-decoder (or sequence2sequence) model for our time series prediction. Rather than having identical multi-tasking layer(s), the encoder-decoder model uses two specialized layers, in which the first memorizes important events of the past (encoder) and the second converts the important events into a prediction of the future (decoder). The encoder-decoder model has been shown to outperform models with identical layers that executes both tasks (e.g. Tonin, 2018; Sutskever et al., 2014). therefore, the encoder-decoder model will also be tested for our time-series forecasting problem.

Encoder-decoder GRU (ED-GRU) and Encoder-decoder LSTM (ED-LSTM)

The third RNN model that is employed is an encoder-decoder model with one input layer, one hidden encoder layer, one hidden decoder layer and one output layer. The number of hidden layers is thus fixed to two in this model type. Both hidden layers consist of GRU units. Similar to the basic model, a fourth RNN model is introduced that is identical to the third model but the GRU units are replaced with LSTM units. The encoder layer processes the input sequence and returns its own internal state to a fixed-length vector (see Section 2.2.3), representing the current state of the time-series. The decoder layer is trained to predict the next time steps by using this state vector as input. The number of units in both hidden layers is a tunable HP that is found through the Bayesian optimization algorithm (see Section 7.3.2).

7.3.2 Hyper-parameters

HPs are the variables that determine the network structure (e.g. the number of units in a hidden layer) and the variables that determine how the network is trained (e.g. the learning rate). HPs are set before a training run (before optimizing the weights and bias). Since it makes a huge impact on the learned model, choosing appropriate HPs plays a crucial role on the quality of a neural network's outcomes.

In this section, seven HPs are identified that we deem important for our forecasting models. For each HP, a lower- and upper-bound or discrete value(s) are defined that are used by the Bayesian optimization algorithm (see next Section 7.3.3) as a search scope. Some HP values are predetermined and thus not optimized through the Bayesian algorithm. These values are based on prior research and/or expert knowledge and can be found in Table 7.1. The reasoning behind the values can be found below.

Gradient descent optimization algorithm

As indicated in Section 2.2.1, the optimization technique that uses the outcome of the loss function after an iteration (passing of one batch of data) to tweak and improve the networks parameters (weights) is called *gradient descent* (GD). The goal of gradient descent is to minimize the chosen loss function. Several gradient descent optimization algorithms are available that assist in determining by how much and when the network parameters have to be tweaked to minimize the loss function. These optimization algorithms tie together the loss function and model parameters by updating the model in response to the output of the loss function.

Some of the most popular optimization algorithms are Momentum, Adagrad, Adadelta, RMSProp and Adam (Ruder, 2017). The last three, RMSprop, Adadelta, and Adam are very similar algorithms that all perform well in similar circumstances (Ruder, 2017). However, Kingma and Lei Ba (2017) show that its bias-correction helps Adam slightly outperform RMSprop and Adadelta towards the end of optimization as gradients become sparser. Adam is therefore seen

	Lower bound	Upper bound	Discrete values
GD opt. algorithm	-	-	Adam
Learning rate	0.0001	0.01	-
Look-ahead	-	-	12
Look-back	1	672	-
No. layers	1	3	-
No. units	14	672	-
Batch size	-	-	32, 64, 128, 256, 512

Table 7.1: Hyper-parameter search bounds for Bayesian optimization algorithm

as the current overall best gradient descent optimization algorithm that outperforms other adaptive techniques (Ruder, 2017; Stanford University, 2018; Walia, 2017)). From this analysis we conclude that Adam (Adaptive Moment Estimation) is the most appropriate and thus will be used throughout this research project as sole gradient descent optimization algorithm.

Learning rate

Learning rate is a HP that controls how much the weights of the network are adjusted with respect to the loss gradient after each iteration. Basically, an updated weight is equal to the sum of the existing weight and the learning rate multiplied by the gradient. Low values of the learning rate results in small steps and thus takes a long time to converge. Also, low learning rates can result in the model getting stuck in a non optimal local minima. On the other hand, high values of the learning rate result in large leaps that quickly finds close to optimal parameter values, but potentially result in the model overshooting the local minima. The Adam optimizer introduced in Kingma and Lei Ba (2017) overcomes these problems by computing adaptive learning rates for each parameter after each iteration. It uses the first moment (the mean) and the average of the second moments (the variance) of the gradients to update the learning rates. Furthermore, it uses an initial learning rate α, the exponential decay rate for the first moment estimates β_1 , the exponential decay rate for the second-moment estimates β_2 and a very small number to prevent any division by zero in the implementation ε. The proposed default values by the authors for these four parameters are: $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. Although these values are a good starting point, some researchers suggest that tuning the initial learning rate α could serve to improve the model's performance (Jason Brownlee, 2017). Therefore, a lower-bound of 0.0001 and an upper-bound of 0.01 will be used. The values of the other three parameters will be used as suggested by the authors.

Look-ahead

The look-ahead HP indicates the number of time steps that the model has to be able to forecast into the future. As requested by the stakeholders, this value is set to 12 time steps, representing 3 hours into the future (12 * 15 minute time intervals).

Look-back

GRU and LSTM models are able to cope with varying input lengths (e.g. different sentence lengths to predict the next word in a sentence). However, since this research focuses on a continuous time-series data set, a fixed input length will be used. This input length determines how many time-steps the model receives as input (look-back) in order to predict the 12 upcoming time steps (look-ahead). This is an important HP since a too short look-back window can result in the loss of long term dependencies whilst a too long look-back window can result in computational difficulties and over-generalization. A lower-bound of 1 time-step and an upper-bound of 672 time-steps (7 days) is used.

Number of hidden layers

The number of hidden layers can have an impact on the ability of the model to learn the underlying relations between variables. Hornik (2017) indicate that problems that require more than two hidden layers were rare prior to deep learning. Two or fewer layers will often suffice with simple data sets. However, with complex data sets involving time-series or computer vision, additional layers can sometimes be helpful. Therefore, the number of hidden layers is introduced as an HP to determine if more than one layer is required. We will test for 1, 2 and 3 layers.

Number of units in hidden layer

The number of units in the hidden layers has a big impact in the ability of the model to learn without under or over-fitting (Heaton, 2008). Under-fitting occurs when there are too few units in the hidden layers to adequately detect the signals in a complicated data set. Using too many units in the hidden layers can also result in several problems. Firstly, too many units in the hidden layers may result in over-fitting. Secondly, a large number of units in the hidden layers can increase the time it takes to train the network, resulting in computational difficulties. A lower-bound of 14 units (equal to number of input units) and an upper-bound of 672 units is used (equal to upper-bound look-back).

Batch size

The final HP is the batch-size. The batch-size defines the number of samples that will be propagated through the network at every iteration, after which the weights are updated using the gradient descent optimization algorithm. Setting the batch-size equal to the number of samples means that all samples are propagated through the network in one go, which is computationally very expensive. On the other hand, setting the batch-size to 1 will not yield any usable results due to the low level of generalization. In their paper, Shirish Keskar et al. (2017) conclude that a batch size between 32 and 512 should yield the best results. A larger batch size results in over-fitting of the models whilst a smaller batch size are not able to escape non-optimal local minima. Furthermore, they indicate that the batch-size should be a value that is a power of two. Following their logic, the following range of possible values is used: 32, 64, 128, 256, 512.

7.3.3 Bayesian optimization

The simplest algorithm that can be used for HP optimization is a grid-search. In a grid-search, a set of parameter values is identified, after which the model is trained for all possible parameter combinations and the best training run is selected. This method is a good choice only when the model can train quickly and the number of HPs is limited, which is not the case for typical neural networks (Yurii Shevchuk, 2016). With the large amount of HPs that are used in our models, this is an inadequate and non-optimal option. Another algorithm is a random-search, which is similar to a grid-search, but instead of trying all possible combinations it randomly selects a subset of the parameters for each training run. In their paper, Bergstra and Bengio (2012) find that random-search is able to find models that are as good or better compared to grid-search within a small fraction of the computation time.

The downside of random search, however, is that it does not use information from prior training runs to select the next setting. This is particularly undesirable when the cost of training runs is high and an educated decision on what training run to execute next is thus required. This is where Bayesian optimization comes in. Bayesian optimization for HPs as introduced in the widely used paper by Snoek et al. (2012), employs a Gaussian Process to calculate the Expected Improvement (EI) for all different HP combinations. The EI represents the probability that a training run with those HP values will improve the Loss function of the current best observation. Bayesian optimization selects the set of HPs with the highest EI probability and trains the network with those values. The outcomes are then added to the Gaussian Process, improving the EI calculation accuracy. Numerous studies have shown that Bayesian optimization for HP tuning currently results in the most accurate model whilst limiting the computation time (e.g. Will Koehrsen, 2018; Vu Pham, 2016). Therefore, Bayesian optimization, using the assumptions and methods introduced by Snoek et al. (2012), will also be used to find the correct HPs for our models.

7.3.4 Training algorithm

This section explains how the RNN models are trained and how the correct HP values are found. For each of the four model lay-outs, the pseudo-code algorithm as depicted in Table 7.2 is followed. This algorithm is repeated three times, for each of the training sets as defined in 6.2. Before starting the algorithm, the data set is standardized to speed up training and reduce the chances of getting stuck in local optima. All fourteen features in the data set are standardized to 0 mean and standard deviation of 1. The training algorithm is now explained.

The Bayesian optimization algorithm starts with a random set of HP values that falls within the specified bounds. A training run with these values is executed, after which the expected improvement for the different possible HP combinations is calculated. The combination with the highest EI value is chosen and a training run with those values is executed. Based on Google ML Engine (2019), this process repeats itself 100 times. They state that, to get the most out of HP optimization, the number of training runs should be over ten times the number of HPs that are tuned. With a maximum of 7 HPs, we assume 100 training runs to be sufficient. Within each training run, the maximum number of epochs (a full pass of the whole training data subset) is set to 1000. This value is chosen since the algorithm is not expected to reach this number of epochs and if it does, further improvements are expected to be very marginal. The algorithm is not expected to reach 1000 epochs due to *early stopping*. Early stopping is introduced to prevent over-fitting and reduce computation time: if the validation accuracy (RMSE) stops increasing for five epochs in a row, the training run is ended. To make sure that the best model is used, we also revert the weights of the final model to weights of the epoch that led to the lowest validation RMSE. Finally, the HP values of the training run that leads to the lowest RMSE on the validation set are identified as the best values.

The training algorithm is translated into an on-line optimization problem so that it can be executed on the Google Cloud Machine Learning Engine. The engine is a managed service where you pay for the hardware resources that are used. This allows us to 'rent' multiple GPUs to execute the training algorithm for the 4 RNN models and their three individual training sets at the same time (a total of 12 runs). A single worker instance with a single NVIDIA Tesla K80 GPU is rented for each training run.

```
1.  Get set of random HP values
2.  For run = 1 to 100
3.      Set epochs = 0
4.      While epochs <= 1000
5.          Split training data subset in batches of size batch_size
6.          For each batch in batches
7.              Pass batch through network
8.              Update weights using gd_opt_algorithm
9.          End for each
10.         Calculate RMSE on validation data set
11.         If validation RMSE did not improve for 5 epochs then
12.             Reset weights to model with lowest RMSE
13.             Exit Do
14.         End if
15.         Set epochs =+ 1
16.     Do
17.     Compute EI for different HP combinations and select highest
18. Next run
19. Select set of HP values with lowest RMSE
```

Table 7.2: Pseudo-code for training algorithm RNNs

7.4 Results

We have identified eight different models that are tested to see which model provides the best results: historical average (HA), last observation carried forward (LOCF), vector auto-regression (VAR), random forest (RF), basic GRU (B-GRU), basic LSTM (B-LSTM), encoder-decoder GRU (ED-GRU) and encoder-decoder LSTM (ED-LSTM). As indicated in 6.2, nested cross-validation is used to provide more robust model performance statistics. The data set is split into three separate sets, after which the models are trained by using each split's individual training and validation subset. The trained models are then tested on the split's test subset. Subsequently, the errors of the three tests are averaged in order to compute the estimate of model error. The RMSE and MAE that are a result of this procedure for all eight models can be found in Table 7.3. This table specifies the errors for each individual time step ahead (1 step ahead to 12 steps ahead). The individual performance measures of the three sets and the corresponding HPs that were obtained through Bayesian optimization can be found in appendix B.

We can observe that, especially for the first 10 time-steps, the four RNN models outperform the baselines that were set in terms of both RMSE and MAE. Furthermore, we observe that the performance of an encoder-decoder model with LSTM units outperforms a basic LSTM RNN. However, this does not hold for the GRU models, in which the basic GRU model outperforms its encoder-decoder counterpart. This may be explained by the fact that a GRU has only one state, whereas LSTM has two states (see 2.2.1). The value of these states represent the information that is passed from the encoder to the decoder layer. The GRU would therefore benefit less from having a separate decoder layer that is able to learn what it is supposed to generate by using these states.

We find that the basic GRU model outperforms the other RNN models whilst also having the shortest training time. To illustrate, the training algorithm for the complete training set S3 (see 6.2) took just over 20 hours to complete for the basic GRU model and just shy of 26 hours for the basic LSTM model. Similar to Chung et al. (2014), we consider the close measures of accuracy between LSTM and GRU to suggest that the best choice of the type of unit may depend heavily on the data set and corresponding task. However, due to the reduced complexity of GRU units, the training time is shorter compared to LSTM and thus often preferable. The Bayesian optimization algorithm for the HPs resulted in a model with a single layer for all three of the data set splits, indicating that this is suffice to learn the relationships between the input and output features in the experiments. Concluding, the best short-term passenger flow forecasting for EA is a GRU based model with a single hidden layer. When training the model in the future, the HPs needs to be redetermined with Bayesian optimization using the newest available data. The model has a mean absolute error of just over 22 passengers, that we believe to be accurate enough to provide meaningful decision-support.

7.5 Lessons learned

The used training algorithm, feature set and resulting final model were not established directly from scratch. As indicated in Section 3.2, the steps in the used CRISP-DM framework are often performed iteratively, going back and forth between stages. However, in order to create a structured and coherent report, we have only included the final outcomes of the iterations in this chapter. In this section, an overview is provided of the most important findings that we have come across during the model development that finally led to the above results.

The first steps of model development were taken in October with a data set that covered April up to and including September. We initially used only 9 data features, in which we did not have the discrete data feature *SecurityFilterOpen* and had only one data feature to represent the five flight features of the current model (see 6.1.3). The single flight feature represented the

	Baseline				RNN			
	HA	LOCF	VAR	RF	B-GRU	B-LSTM	ED-GRU	ED-LSTM
RMSE								
t + 1	99.09	40.91	36.26	34.98	**26.54**	28.54	27.79	28.13
t + 2	99.07	56.19	36.23	35.32	**27.02**	29.36	27.61	28.20
t + 3	99.08	70.01	36.35	35.90	**27.75**	29.13	27.94	28.32
t + 4	99.10	81.16	36.57	36.21	**28.42**	30.97	28.43	28.75
t + 5	99.12	89.47	37.13	36.51	**28.65**	31.28	28.98	29.37
t + 6	99.14	95.49	37.81	37.06	**29.29**	31.51	29.37	29.71
t + 7	99.16	99.75	38.59	37.37	**29.27**	31.21	29.75	29.89
t + 8	99.16	102.97	39.04	38.10	**29.77**	33.17	30.27	30.33
t + 9	99.17	106.13	39.23	38.41	**29.92**	32.81	30.52	30.81
t + 10	99.18	109.55	39.53	38.71	**30.41**	32.31	30.88	31.51
t + 11	99.2	113.26	40.75	39.52	**31.48**	35.23	32.63	33.39
t + 12	99.21	116.98	44.22	41.10	**37.28**	38.13	37.38	37.46
MAE								
t + 1	80.73	29.70	27.42	25.70	**19.85**	21.45	20.63	21.24
t + 2	80.71	41.84	27.46	25.90	**20.21**	21.98	20.85	21.27
t + 3	80.72	53.30	27.64	26.32	**20.50**	21.80	21.26	21.38
t + 4	80.73	62.59	27.97	26.64	**21.16**	23.02	21.64	21.70
t + 5	80.75	69.41	28.53	26.85	**21.48**	23.38	22.12	22.20
t + 6	80.76	74.25	29.10	27.24	**21.78**	23.49	22.46	22.48
t + 7	80.77	77.49	29.85	27.49	**21.97**	23.29	22.66	22.65
t + 8	80.77	80.25	30.29	28.10	**22.47**	24.35	22.86	22.96
t + 9	80.78	83.38	30.51	28.36	**22.06**	24.46	23.10	23.41
t + 10	80.80	86.80	30.76	28.53	**23.04**	24.17	23.65	23.91
t + 11	80.82	90.54	31.70	29.21	**24.62**	26.19	25.19	25.26
t + 12	80.84	94.34	34.37	30.55	**28.15**	28.62	28.5	28.38

Table 7.3: Forecast modelling results: RSME and MAE

number of passengers that did not yet pass the pre-security filter and were scheduled on a flight with a departure time between 0 and 3 hours from the interval time. By using a simple grid-search (see 7.3.3) to find the HPs, we were able to immediately generate satisfactory forecasts with basic GRU and basic LSTM models. When new data became available so that October and the first half of November could also be included in the data set, we confidently tested the developed models on the expanded data set. However, the results were not as expected and were clearly not satisfactory. Analysis on the new data set resulted in the finding that the number of departing passengers during the beginning of November was a factor 1.5 smaller than the previous months of data. Arguably, the model did not learn a general sense of crowdedness since the available months of training data all had similar passenger numbers and lack seasonality.

To mitigate this problem, the best solution would be to include more data that also covered the less crowded months (November-February according to the stakeholders). However, this data was not available due to policies on how long the logs can be stored in AirDCS (see 5.2.1). We therefore decided to expand the number of features so that the model would have more handles to about learn general crowdedness. These features are the five features as introduced in Section 6.1.3. With the new total of 13 data features, a grid-search was performed and the resulting basic GRU and LSTM models were tested. This greatly improved accuracy and the results were close to our initial model that was tested on the smaller data set. Analyzing the predictions that the models made, we however found that it took the models some time-steps before realizing that no more passengers would be arriving after closing the security filter at the end of a day. To provide the model with these insights, the final data feature *SecurityFilterOpen* was introduced. The renewed results for the time-steps during which the security filter was closed were instantly improved.

With the increased number of data features, the underlying relationships between the input features also got increasingly complex. We considered that the model could possibly be improved by first extracting the relevant (underlying) features and then using them to make the prediction, opposed to using the 14 raw input features directly to make the predictions. We found that this can be accomplished by using encoder-decoder models (see 7.3.1) and decided to incorporate them into the analysis. Furthermore, we decided to incorporate the number of layers for the basic models into the analysis as an additional HP to see if an increased number of GRU or LSTM hidden layers could achieve the same. This however created an additional problem. With the increased number of models and the increased number of HPs, the computational requirements of a basic grid-search exploded. Using a basic grid-search with only a limited number of possible values per HP, we required well over 5000 training runs of at least 15 minutes each. Furthermore, a grid-search was not expected to result in near-optimal HP values but was expected to provide a more general direction of the values.

This is where we decided to incorporate Bayesian optimization into the training algorithm to find the correct HP settings. It has proven itself across several papers (e.g. Will Koehrsen, 2018; Vu Pham, 2016), and with a fraction of the computational requirements generally finds better results than a grid-search or random-search. The Bayesian optimization algorithm is however still relatively computational expensive, so in order to improve flexibility of our models, the whole training algorithm was translated into an on-line optimization problem. This problem could be executed in parallel for all models on the Google Cloud Machine Learning Engine. The engine is a managed service where you pay for the hardware resources that are used. This allowed us to ‘rent’ multiple GPUs to execute the training algorithm as defined in Section 7.3.4 for the 4 RNN models at the same time.

Finally, the created flexibility also allowed for an increase in the robustness of our validation accuracy. The prior models that were optimized through a grid-search were all only trained and tested on the full data set. With the flexibility of the cloud engine, we were able to introduce nested cross-validation (see 6.2). By introducing three separate data set splits the computational requirements tripled, but by employing Googles distributed network of computers this did not form an issue.

7.6 Scientific contribution

In Section 2.3 we identified three scientific contributions that the development of a short-term passenger flow forecasting model in an airport context would bring. These three points are addressed separately.

The first contribution lies in the insights that are found when developing, using and analyzing a short-term passenger flow forecast model within the specific airport context. This contribution results from the analysis presented in Chapter 2 where an extensive literature review specifically aimed at short-term passenger flow forecasting within the airport context did not yield any relevant results. The previous chapters have primarily focused on the development side of this contribution, whereas the coming chapters focus on the actual usage of the forecast model. During development we have found that we are able to build a model that can forecast the short-term passenger flow at the security filter with acceptable accuracy. We discovered that the information that an airport receives throughout the day concerning the number of passengers and flights that are expected to depart in the near future is quite valuable and increases short-term passenger flow forecast accuracy. Furthermore, airport specifics like the open and closing times need to be incorporated in the forecasts to improve accuracy. We believe that, by carefully selecting the features, the same development steps can be executed for other bottlenecks at EA to create accurate forecasting models. We also believe that similar models can be used at different airports, by taking the specific airport characteristics into account.

A second contribution lies in the insights found when developing and using a multi-source multivariate GRU short-term passenger flow forecasting model with temporal links between the data sources. As indicated, we established that the majority of the methods that are proposed throughout literature merely use a single source time-series and its derivatives (e.g. day of week) as input for the forecasting model. In order to compare our model to these methods, we run the training algorithm of Section 7.3.4 on a reduced data set that includes only 4 of the 14 data features that were introduced in Chapter 6: we use the historical values of the outcome variable (*PreSecurityFilterScanIn*) and the three derived calender features (*SecurityFilterOpen*, *HourOfDay* and *DayOfWeek*). This is comparable to the input features that are used in the proposed methods throughout literature (e.g. Tsai et al., 2009; Ma et al., 2014; Toque et al., 2018; Zhu et al., 2018).

The results of the models with a reduced set of input parameters can be found in Table 7.4. We find that excluding the time-series data with temporal links from the data set such as the baggage drop-off, check-in or parking related time-series greatly reduces accuracy. This is especially the case for the first few future time-steps. We find the basic GRU to also outperform the other methods on this reduced data set. The RMSE of this model averages just over 46 and the MAE averages just shy of 35, whilst its complete data set counter part respectively averages 35 and 22 on these performance measures. To illustrate, Figure 7.1 shows the difference in forecasts for two days in December 2018 between a basic GRU model that uses the complete set of input parameters and a basic GRU model that uses the reduced set. The figure shows the four time step (one hour) ahead *PreSecurityFilterScanIn* forecasts for the two models as well as the actual values. As can be seen, the reduced model fails to forecast peaks and both under- and overestimates the actual number of passengers. It is clear that the model that employs all 14 data features greatly outperforms the reduced model. Completely analyzing the individual effect of including or excluding individual data features could result in a separate research project on its own, and due to time constraints is therefore reduced to these preliminary tests that excludes all external time-series. We suspect that the dynamics within an airport terminal are too complex to allow for a uni-variate model that only includes a single-time series and its derivatives. We conclude that a multi-source multivariate GRU model is able to capture the dependencies that exist and can be used for short-term passenger flow prediction.

The final scientific contribution lies in the comparison of different short-term forecasting models. This comparison has been evaluated in Section 7.4, resulting in the conclusion that the best short-term passenger flow forecasting is a GRU based model with a single hidden layer. However, this is specifically the case for EA and since the forecast accuracy of the different RNN models were relatively close, other researchers should evaluate if this is also the case in their specific context.

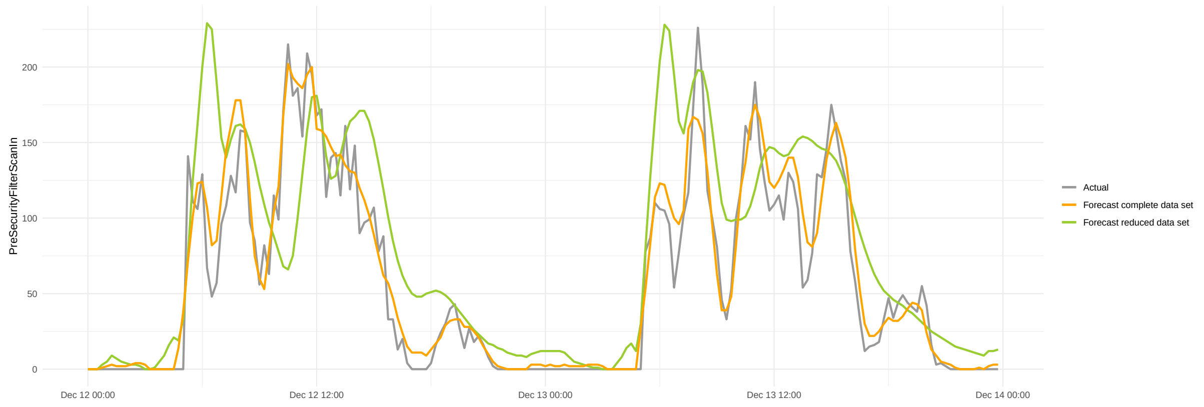


Figure 7.1: Two days of four time step (one hour) ahead *PreSecurityFilterScanIn* event actuals and forecasts. The forecasts are made using a basic GRU model that uses the complete set of data features and a basic GRU model that uses a reduced set of data features.

	Baseline			RNN			
	HA	LOCF	VAR	B-GRU	B-LSTM	ED-GRU	ED-LSTM
RMSE							
t + 1	99.09	**40.91**	52.25	44.84	45.92	44.50	48.21
t + 2	99.07	56.19	61.92	**45.68**	49.14	46.58	47.82
t + 3	99.08	70.01	68.70	**45.61**	49.61	46.97	47.50
t + 4	99.10	81.16	72.90	**46.73**	49.58	47.06	47.34
t + 5	99.12	89.47	75.10	**46.64**	50.31	47.12	47.22
t + 6	99.14	95.49	75.72	**46.00**	50.81	47.21	47.20
t + 7	99.16	99.75	75.08	**46.37**	50.48	47.32	47.29
t + 8	99.16	102.97	74.16	**47.42**	50.14	47.45	47.50
t + 9	99.17	106.13	73.69	**46.58**	49.50	47.69	47.74
t + 10	99.18	109.55	73.71	**46.94**	50.32	48.01	47.97
t + 11	99.20	113.26	74.09	**46.91**	50.23	48.40	48.22
t + 12	99.21	116.98	74.58	**46.28**	49.20	49.10	48.64
MAE							
t + 1	80.73	**29.70**	39.43	33.53	34.55	32.98	35.69
t + 2	80.71	41.84	47.85	**34.14**	36.82	34.53	35.17
t + 3	80.72	53.30	53.57	**34.08**	37.16	34.85	34.92
t + 4	80.73	62.59	56.79	**34.70**	37.24	34.94	34.84
t + 5	80.75	69.41	58.12	**34.65**	37.67	35.05	34.84
t + 6	80.76	74.25	58.61	**34.21**	38.14	35.19	34.93
t + 7	80.77	77.49	58.41	**34.70**	37.73	35.37	35.08
t + 8	80.77	80.25	58.12	**35.47**	37.56	35.55	35.29
t + 9	80.78	83.38	58.42	**34.63**	37.23	35.80	35.50
t + 10	80.80	86.80	59.17	**34.94**	37.80	36.12	35.69
t + 11	80.82	90.54	60.00	**34.82**	37.77	36.49	35.94
t + 12	80.84	94.34	60.86	**34.45**	37.00	37.13	36.43

Table 7.4: Reduced data set forecast modelling results: RSME and MAE

7.7 Conclusion

In this chapter we have first introduced the metrics (RMSE and MAE) and baselines models (HA, LOCF, VAR and RF) that are used to evaluate the performance of four different RNN models. These RNN models are trained through an introduced algorithm that also determines the correct hyper-parameters through Bayesian optimization. We find that a basic GRU model with a single hidden layer outperforms the other models and is able to forecast the short-term passenger flow with satisfactory accuracy. During this first analysis, the models use the complete set of data features as introduced in Chapter 6. A second analysis that employs a reduced set of data features is performed in order to place our model in current literature and to find the added value of including multi-source time-series in the forecasting model. We find that including these time-series with temporal links between them such as the baggage drop-off, check-in or parking related time-series greatly increase accuracy.

Chapter 8

Decision-support modelling

Eindhoven Airport desires to use the short-term forecasting model that was developed in the previous chapter in the day-to-day operation. This chapter therefore introduces the model that uses the real-time short-term forecasts to provide decision-support and enables proactive flow management at Eindhoven Airports security filter.

In order to generate the forecasts in real-time, a server is set-up that receives real-time data from AirDCS, Parkbase and Skyguide. Using the real-time data, a renewed data set is created every 15 minutes that consists of the 14 data features as defined in 6.1 and contains the number of time-steps as required by the forecasting model (see look-back: 7.3.2). This data set is passed through the forecast model to generate an updated 12 step-ahead forecast of the number of passengers that is going to arrive at the security filter. This process starts directly after every 15 minutes, and takes less than a minute to complete.

However, providing the bare forecasts to the relevant stakeholders is not expected to be particularly useful. In order to determine if a bottleneck is imminent (unacceptable queue waiting times), this forecast for example has to be combined with the capacity of the security filter (see 4.2.5) and information on the current crowdedness. Furthermore, the variation in processing and arrival times and overflow of passengers between time-steps has to be taken into account. This is out of the skill-set of the stakeholders and should be taken out of their hands. Therefore, in Section 8.1 we propose to use a queuing model that utilizes the forecast and the planned capacity to compute the expected waiting times for each of the 12 upcoming time-steps.

Subsequently, to further provide decision-support, actions are proposed that can help mitigate forecasted bottlenecks. The action proposal part of the model is introduced in Section 8.2. Finally, the design of the dashboard is provided that combines the forecast, expected queue times and proposed actions in a single view.

8.1 Waiting time calculation

Through data analysis on the historical *PreSecurityFilterScanIn* events we have found that the inter-arrival times of passengers at the security filter queue has an coefficient of variation (CV) of just shy of 1.1 and thus closely follows an exponential distribution (CV of 1). Subsequently, the CV of the service times is found through data analysis on the G4S logs (see Section 5.1.2). We do not present an exact figure for the CV here since this is identified to be sensitive business information. Nevertheless, we find that the service times follow a more general distribution with a high CV. Finally, we find that there are multiple lanes at the security filter. This leads to the conclusion that the security filter queue is a $M/G/c$ queue: the arrival rate closely follows an exponential distribution (M) whilst the service times have a general distribution (G) and there are c lanes. Furthermore, there is a single queue for all lanes, the number of waiting positions is assumed to be infinite and waiting passengers are served according to the First-Come-First-Serve (FCFS) policy.

Queuing models usually assume a steady-state environment when calculating performance measures (e.g. queue waiting time), in which the rates and number of servers (lanes) do not change. However, the arrival rate, service times and number of queues are all time dependent for the security filter that we consider. This means that general steady-state calculations can not directly be applied to calculate the expected waiting time. Considering the time dependency, we are thus dealing with a more specific *M(t)/G(t)/c(t)* model in which the arrival rate, service time and number of lanes is different for each 15 minute time interval t.

Closely related, Stolletz (2011) introduce a *M(t)/G/c(t)* model to calculate performance measures for a check-in queue at an airport. A variation of their method will be used to compute the expected waiting times in our model. The difference lies in that they assume the service times to have the same distribution for every time interval whilst this is not the case for the security filter at EA. The service times change between time intervals because they are dependent on the number and type of staff members that are operating the lanes: more staff members results in shorter service times per lane. This difference results in an adaption of their method where the processing times are assumed to be time dependent rather than stationary (*M(t)/G(t)/c(t)* opposed to *M(t)/G/c(t)*).

The basic idea behind their approximation relies on a loss queuing model. Generally, in a loss queuing model there is a finite number of waiting positions: all passengers that would arrive after the finite amount has been reached would leave the queue and not return. The loss queuing model uses probability calculations to approximate the number of passengers that will be lost. Obviously, passengers will not just leave the queue of an airport's security filter. To overcome this, the loss models of consecutive periods are connected via an artificial backlog, which equals the (artificial) loss rate of the preceding period. Passengers that are blocked/lost in one period are added to the queue of the succeeding period. Thus, to derive the artificial arrival rate of the subsequent period, the backlog rate is added to the original arrival rate. This artificial arrival rate is then used to approximate the utilization. Based on this utilization, a modified arrival rate is derived such that a stationary waiting system would reach the same expected utilization. Finally, the performance measures can be approximated using this modified arrival rate and a stationary *M/G/c*-approximation. The algorithm and corresponding formulas can be found in appendix C.

The algorithm requires four input parameters: arrival rate in period i λ_i, processing rate per lane in period i μ_i, open lanes in period i c_i and the squared coefficient of variation of the processing rate CV^2. The arrival rate λ_i of past time intervals is calculated through actual data and the arrival rate of the 12 upcoming periods are computed using the generated forecast. As mentioned in Section 4.2.5, the planned processing rate μ_i per lane and the number of open lanes c_i is a direct result of the planning of the lane configuration that is constructed by one of EA's AOMs every week. This lane-configuration planning states exactly how many lanes will be open during every 15 minute time interval in a day and also indicates the number of staff members and their required skills. The specific configuration for a 15 minute time interval corresponds to a contracted rubric code and with that a contracted processing rate. For the course of this research project, this information is uploaded to the server manually every week. Finally, the squared coefficient of variation of this processing rate CV^2 is obtained through data analysis on a historical throughput data set of the security lanes provided by G4S.

However, the stakeholders indicate that generally the contracted processing rate is not completely achieved and might need some alterations when using them in the waiting time calculations. Furthermore, the CV^2 of the processing rate was computed on a data set of which data quality is questionable (see Section 5.1.2). These findings leave room for tuning of these variables in order to make the calculated expected waiting times match the waiting times that are experienced in reality. At the start of the pilot period (see Chapter 9), the contracted processing rate and computed CV will be used but these can thus be altered based on the findings of the stakeholders.

8.2 Action proposal

Computing the expected passenger numbers and expected waiting times provides basic decision-support for the AOMs. With that information they are able to foresee the formation of a bottleneck and possibly take proactive decisions to prevent the problems. However, as found through several interviews, the decisions that are made are currently mainly based on experience and (in similar settings) are believed to vary among the AOMs. Furthermore, the range of possible proactive decisions that an AOM can choose from to prevent a future bottleneck often depends on dynamic external factors (e.g. one cannot open another security lane if six lanes are already open). This calls for additional decision-support. Therefore, in this section, an action-proposal model is introduced that provides the AOMs with a range of possible decisions to mitigate upcoming bottlenecks.

Decision support systems (DSS) architecture contains three key components: a knowledge base, a computerized model, and an user interface (Jao, 2010). The latter is introduced in the next section in the form of a real-time dashboard. This dashboard primarily contains three elements to provide decision support: the passenger flow forecast, the expected waiting time calculations and action proposals to prevent upcoming bottlenecks. The latter will be initially grafted on a knowledge base whilst the other two mainly rely on the developed computerized model.

Using interviews with several stakeholders, a range of possible mitigating actions is constructed for each of the 12 upcoming time-steps. The range is different for every time-step since not all actions can be executed within the remaining time between that time-step and the current time. To illustrate, an AOM cannot request an additional lane only 15 minutes in advance since no staff members will be available. The proposed actions for a time-step are actions that will mitigate the problems in that specific time-step. A maximum number of three actions is shown per time-step and based on initial discussions with stakeholders, they are ordered on level of the highest desirability (1st proposed action is more desirable than 2nd proposed action).

8.2.1 Actions

The seven mitigating actions that have been identified are now introduced. In appendix D, a table can be found that indicates which action is allowed to be proposed for which future time-step.

Open crew lane

As also illustrated in the problem exemplification of Section 1.2.3, the AOM is able to request for the crew lane to be opened for passengers. The crew lane normally is only used for staff members of the airport and airline crew members. This action can be executed on the spot or requested in advance, since the crew lane is practically always staffed.

Passenger announcement: Security filter for flight X is closed / opened early

The airport has a public address system by which passengers can be informed throughout the terminal (e.g. to urge missing passengers to go to their gate). It can also be used to make an announcement stating that the security filter is currently closed for a specific flight. This action can be used when a bottleneck is forecasted to exist in the first upcoming time-periods and when over-capacity is expected after these busy time-periods.

Similarly, an announcement can be made that indicates that the security filter is opened early for a specific flight. This option can be executed when a bottleneck is expected to emerge somewhere in the near future and over-capacity is expected in the time-periods leading up to the problematic time-period.

These actions will likely result in passengers of the announced flight going to the security filter later or earlier than initially planned, thus spreading the flow of passengers over time. The model computes the expected over-capacity of a time-period by calculating the number of passengers that can be added to the arrival rate of that time-period so that the expected waiting is still below a 10 minute threshold (see 8.3 for treshold explanation). If there is enough over-capacity, the model suggests announcing a flight that fits in the over-capacity window (number of passengers is less than over-capacity amount). The model always chooses the flight with the highest number of passengers that fits in order to maximie the effect of the action.

Open X lane Y minutes early / later

As indicated, a lane configuration planning is constructed every week indicating how many lanes will be open during every 15 minutes of a day. On average, the lane configuration changes every hour (the configuration rubrics is the same for 4 consecutive time-periods). The change-overs almost always include a change in number of lanes that are open, which leaves room for a bottleneck mitigation action. In case a bottleneck is expected to form in the future, the AOM can request to open lanes early or close lanes later. To illustrate, if a bottleneck is expected to arise 4 time-periods from now since only 3 lanes will be open, but a 4th lane was open in the preceding period, the AOM can request that 4th lane to remain open for another 15 minutes.

Extra staff members internal / external

A more general action is to request more staff members to assist at the security filter. As indicated, extra staff at the lanes results in a higher throughput. This is mainly a result of quicker processing of the bags and additional assistance for passengers that are struggling with the procedure. When problems suddenly arise in the current time-period or are expected to arise in the next time-period, the on-duty AOM can request his direct colleagues (off-duty AOMs) to assist at the security filter. This action is an internal request for extra staff members and is not always desired but an option nonetheless. If the model foresees problems in the after the next time-period, the AOM can request extra staff members externally from either G4S or Viggo to assist at the security filter.

8.3 Dashboard design

A dashboard is designed that combines the short-term forecasts, waiting time calculation and action proposal in a single screen. Figure 8.1 shows a screen-shot of the dashboard and includes the prediction of a bottleneck that will form in the near-future. A total of 16 rows are shown in chronological order, each representing a time step of 15 minutes. The first four rows refer to the previous hour (historical), the fifth row is the current time step (historical and forecast) and the next eleven rows are the upcoming time steps (forecast).

The Pax column (industry's abbreviation for passengers) indicates the actual (A) or expected (E) number of passengers during that time step. For the current time step both the actual and the total expected number of passengers for that time step is shown. In this row, the actual number of passengers that have arrived until the current time is updated every minute. The planned capacity column merely provides the AOMs with some more information and is not used directly in the models. It is calculated by using the contracted throughput per minute, multiplied by 15 minutes and then multiplied by the number of open lanes. The expected waiting time column indicates the total time that a passenger that arrives in that time-step is expected to wait. Based on discussions with the AOMs, a three-fold color scheme is introduced: waiting time of 0-10 minutes is green (level 1), waiting time of 11-20 minutes is yellow (level 2) and waiting time of more than 20 minutes is red (level 3).

The proposed action columns are only filled for the 12 last rows (no need for historical action proposals). Furthermore, the model only determines which actions are appropriate if the expected waiting time is at level 2 or level 3. The last taken action field serves as an input field. It includes a drop-down list of actions that are proposed during that time-step but also allows for free-form text input. This field is used to record the actions that are taken for that time-step and is constantly displayed until that time-step has passed by more than a hour (outside of the 4 historical time-steps). This also helps AOMs in the hand-over between their shifts.

The last three columns serve as additional information to provide more context about the current situation. If required, users are able to alter the rubric code of a specific time-step by pressing the gray button on the right-top, which brings them to a different overview that shows the configured rubric code per time-step.

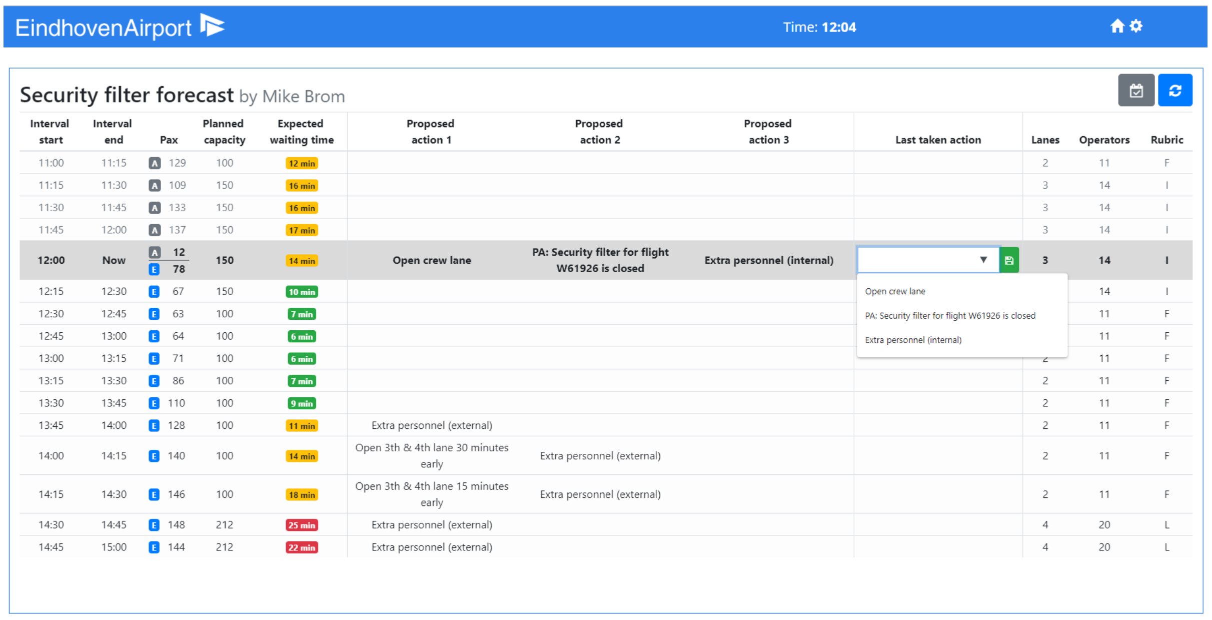
8.4 Limitations

There are a few limitations of the decision-support model that are a result of the available implementation time of the model and the scope of this research project. These limitations are now introduced to provide a detailed overview of what a more elaborate model would entail.

The introduced list of actions is seen as a limited initial starting point that needs to be updated in the future by using data analysis to see which actions have been taken and what their effect has been. The dashboard allows the AOMs to record the actions that have been taken and also allows for free-form input to gather possible mitigating actions that have not yet been included in the list. This should result in better and increasingly data-driven action proposals.

AOMs are able to record their actions that they took to mitigate a bottleneck in a specific time-period in the dashboard. However, the model does not currently take these actions into account when updating the forecasts and expected waiting times. To illustrate, when an AOM requests a lane to be opened early, the model with have no knowledge of this and will therefore continue indicating that a problem will arise in the near future. The AOMs are currently expected to take the already executed actions into account themselves to determine if further actions are required.

Lastly, the actions in the established list are not always feasible. This is especially the case when requesting additional staff members and requesting a lane to be opened early. These actions mostly require staff members to get to work earlier than expected, which is not always possible. Therefore, future analysis needs to also take into account how many times the actions were executed and identify the reasons for not executing them, in order to be able to construct a more appropriate list of actions.



Security filter forecast by Mike Brom

Interval start	Interval end	Pax	Planned capacity	Expected waiting time	Proposed action 1	Proposed action 2	Proposed action 3	Last taken action	Lanes	Operators	Rubric
11:00	11:15	A 129	100	12 min					2	11	F
11:15	11:30	A 109	150	16 min					3	14	I
11:30	11:45	A 133	150	16 min					3	14	I
11:45	12:00	A 137	150	17 min					3	14	I
12:00	**Now**	**A 12 / E 78**	**150**	14 min	**Open crew lane**	**PA: Security filter for flight W61926 is closed**	**Extra personnel (internal)**		**3**	**14**	**I**
12:15	12:30	E 67	150	10 min					[illegible]	14	I
12:30	12:45	E 63	100	7 min					[illegible]	11	F
12:45	13:00	E 64	100	6 min					[illegible]	11	F
13:00	13:15	E 71	100	6 min					2	11	F
13:15	13:30	E 86	100	7 min					2	11	F
13:30	13:45	E 110	100	9 min					2	11	F
13:45	14:00	E 128	100	11 min	Extra personnel (external)				2	11	F
14:00	14:15	E 140	100	14 min	Open 3th & 4th lane 30 minutes early	Extra personnel (external)			2	11	F
14:15	14:30	E 146	100	18 min	Open 3th & 4th lane 15 minutes early	Extra personnel (external)			2	11	F
14:30	14:45	E 148	212	25 min	Extra personnel (external)				4	20	L
14:45	15:00	E 144	212	22 min	Extra personnel (external)				4	20	L

Open crew lane
PA: Security filter for flight W61926 is closed
Extra personnel (internal)

Figure 8.1: Screenshot decision-support dashboard

8.5 Conclusion

In this chapter we have constructed the final decision-support model. The model is three-fold: it provides the decision-makers with a three hour ahead forecast of passengers that will arrive at the security filter, an expected waiting time for the 12 time intervals of 15 minutes within these three hours and finally the model suggests actions that the AOMs can take to mitigate upcoming exploding waiting times.

The expected waiting times are computed by offsetting the passenger flow forecast to the planned security filter capacity. An *M(t)/G(t)/c(t)* queue approximation model is used in which the arrival rate, service time and number of lanes is different for each 15 minute time step. The basic idea behind the approximation relies on a loss queuing model in which the loss models of consecutive periods are connected via an artificial backlog. This backlog equals the loss rate of the preceding period. Passengers that are blocked/lost in one period are thus added to the queue of the succeeding period. Several parameters in this approximation allow for tuning in order to make the calculated expected waiting times match the actual waiting times.

Furthermore, the range of possible actions to mitigate expected undesirable waiting times is constructed for each of the 12 upcoming time steps. The range is different for every time step since not all actions can be executed within the remaining time between that time step and the current time. The range of actions that are actually proposed is dynamic and depends on the current conditions (e.g. one cannot open another security lane if six lanes are already open). Although the model is currently mainly based on expert knowledge and contains some limitations, we believe it to be a strong start to enable a proactive management flow.

Chapter 9

Evaluation

In the previous two chapters we have developed the short-term forecast model and the additional decision-support model. Since the models are fed by real-time data with sufficient timeliness, for sake of evaluation we are able to deploy them in the day-to-day operation of Eindhoven Airport. This deployment is done during a pilot phase in which a subset of AOMs is given instructions as to how to use the dashboard and what is expected of them. This pilot phase had an initially planned length of six weeks, starting from the beginning of December 2018. We considered six weeks to be enough for the AOMs to get acquainted with the system, allow for some parameter setting of the waiting time calculations (see 8.1) and finally be able to quantitatively evaluate the decision-support model (e.g. number of taken actions). However, due to the project's dependencies on several external parties, we experienced multiple last-minute issues with data availability and data-stream stability that were outside of our control. Therefore, also due to time-constraints of this research project, we had to cut the pilot phase short to just 2.5 weeks (7th to 23th of January 2019). The decision-support model needs tuning and the AOMs need to get acquainted and build trust before they will actively use the model to perform pro-active actions. The limited pilot-phase length does not allow for this and we are therefore not able to perform quantitative evaluation on the decision-support model. Instead we will use qualitative evaluation, based on interviews with the AOMs, to evaluate the (expected) usefulness and perceived accurateness of the model. The chosen forecast model will first be evaluated quantitatively by using the newly available data of the 2.5 week pilot phase.

9.1 Quantitative evaluation

During the pilot phase, a new forecast is made every 15 minutes by using the latest available data. These 12-step ahead forecasts are automatically stored in a database to allow for model evaluation. The model that is used is a newly trained model, using the same training algorithm as introduced in Section 7.3.4. In this training run we use the whole data set that was established in Chapter 6 (from 3th of April 2018 to 17th of December 2018) and split it into a training and validation set (80%-20% ratio). No test set is needed, as the model is tested during the pilot-phase. The results in term of RMSE and MAE of the deployment of this model during the pilot-phase can be found in Table 9.1.

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8	t+9	t+10	t+11	t+12
RMSE	25.53	25.60	27.22	27.92	28.72	29.41	30.37	30.85	30.73	31.66	33.52	36.64
MAE	19.43	19.54	20.89	21.56	22.19	22.85	23.38	23.81	23.72	24.52	26.07	28.89

Table 9.1: Forecast accuracy during pilot phase: RSME and MAE

The RMSE and MAE that were obtained during the pilot phase closely match the results that were found during model development. This indicates that the model is generalizable to new (unseen) data sets.

9.2 Qualitative evaluation

In this section we will use qualitative measures to evaluate the dashboard expected usefulness and perceived accurateness in providing decision-support. A total of 5 semi-structured interviews were performed with individual AOMs, of which citations will be used in this evaluation. Three out of five of these AOMs were given instructions before the start of the pilot on how to use the dashboard and what to pay attention to. These AOMs were selected based on their personal interests and their relative experience: one of them is the most experienced AOM with more than 30 years experience whilst another is new to the field with only one year of experience. Interestingly enough, the two other AOMs saw their colleagues working with the dashboard and directly started exploring and using it during their own shifts. The evaluation focuses on perceived current accurateness and future expected usefulness. The three facets of the dashboard are evaluated individually in the sections below.

9.2.1 Passenger forecast

All interviewees indicate that the perceived accurateness of the passenger forecast is high, stating that the “model seems to be close to the real values, or at least close enough for the purposes of the model”. They indicate that the model is better during busy periods compared to the calmer periods, which is most valuable. The differences that exist “do not have significant consequences on the outcomes of the rest of the model”. They indicate that the model seems to sometimes overestimate the forecast. They speculate that this is probably due to the high number of no-shows during this particular time of the year (January): “10-20% of the passengers of some flights do not show up, even though they are expected to arrive in your system”. This is something that needs to be taken into account in further model development. The perceived accurateness of the passenger forecast is thus deemed high enough by the interviewees for current purposes. They also indicate that “the forecast will be most relevant in a few months, when the busy holiday peak starts”. Finally, the AOM who is responsible for the security configuration planning indicates that he would like to be able “to use the models to evaluate his own forecasts”. This, and other statements make us conclude that the expected usefulness of the forecasts themselves is high.

9.2.2 Waiting-time calculation

We found differences in the perceived accurateness of the waiting-time calculation. The interviewees that were scheduled on the relatively calm evening shifts indicate that the “waiting time seems to accurately represent the real values”. During busier periods however, the other interviewees indicate that the waiting times seem to be slightly underestimated by the model. They understand the direct link between the planned throughput and the forecast in the waiting time calculation, and indicate that the fixed contracted throughput levels for each rubric are not always correct. One of the interviewees indicates that “the throughput is always 10% lower than contracted”, whilst the others indicate that the actual throughput during a time-interval depends on several factors: “for example passenger types, weather conditions and specific very slow passengers all influence the current throughput”. They indicate that they would like to be able to adjust the used throughput level during waiting time calculations for some of the upcoming time periods “based on their experience and current knowledge of the flow”. Also, when they decide to request additional lanes or staff members, this should be taken into account in the model. Finally, considering the color scheme (red/yellow/red), there seems to be a difference in perception on what is seen as an unacceptable queue length: additional discussion among the AOMs is required to determine this. We conclude that the current waiting-time calculation serves as a good starting-point but needs to be further tuned by using the experience of the stakeholders. This will increase perceived accurateness and with that boost the usefulness.

9.2.3 Action proposal

The third facet of the dashboard are the action proposals. Although no actions have been taken yet that were suggested by the model, the interviewees indicate that "they would definitely use the proposed actions after establishing confidence in the model's accuracy". During the interviews a difference has been found among the interviewees about what the action proposals intend to indicate. Two interviewees indicate that (preferably based on extensive data) the model "should only propose actions of which it is sure that they improve the passenger flow". The others prefer the way in which the model operates now, where it suggests multiple actions and leaves it to the experience of the user to determine if and what action should be executed. The latter see the actions more as general tips (AOM decides) while the others see them as mandatory actions (system decides). We believe that the current form of the model is able to become smarter in the future by incorporating passed historical data on taken actions, but this is not available right now. This is an important distinction that needs to be communicated to the users. We conclude that the actions will certainly be used after the stakeholders have gained trust in the accuracy of the forecasts and waiting-time calculations. However, we also see that additional steps are required to make the proposed actions smarter by including more data and incorporating related business dependencies.

9.2.4 Conclusion

Concluding, we see that there is abundant support for the model, which is also clearly shown by the two AOMs who have independently started to use the dashboard during their shifts. The interviewees were all generally positive and the feedback mainly concerned modest improvements to the current model. The suggested improvements are all within realistic bounds of a subsequent project. Also, the interviewees indicate "although being a first draft, the model is already very useful and will only get better in the future". A a final conclusion the stakeholders indicate that "the system has a lot of potential, is very user-friendly and comes very close to our wishes".

Chapter 10

Conclusion

In this research we have constructed a forecasting model that is able to predict the short-term number of passengers that will arrive at the security filter of Eindhoven Airport. The short-term forecast covers 12 time-steps of 15 minutes each and thus provides a 3 hour look into the future. Subsequently, the forecasts are used to compute the expected waiting time that passengers will experience at the security filter. Finally, a model is developed that is able to suggests mitigating actions in case the waiting times are expected to rise to undesirable lengths. The passenger flow forecast, expected waiting times and action proposals are captured in a single dashboard. This final solution is the result of an elaborate study that was aimed at enabling proactive flow management in order to reduce the severity and number of bottlenecks experienced in Eindhoven Airport's outbound passenger flow. In this section, the main findings and answers to the research questions as introduced in 1.4 are provided. The five sub-questions are addressed individually which ultimately leads to the answering of our main research question:

How can proactive flow management be enabled by using real-time short-term passenger flow forecasting and providing additional decision-support in order to reduce the severity and number of bottlenecks experienced in an airport terminal environment?

The answer to this main research question lies directly in the answers to the sub-questions. After providing the summarized answers to the sub-questions we will provide recommendations for Eindhoven Airport and finally suggest possible directions for future research.

10.1 Research questions

10.1.1 Bottlenecks and flow management

The first sub-question concerned identifying the current situation regarding bottlenecks and flow management in the outbound passenger flow at EA. We found that the capacity crunch at Eindhoven Airport leads to an increase in operational problems in the form of those bottlenecks. These problems are amplified by inadequate flow management, that includes both insufficient resource planning and a reactive flow management style in the day-to-day operation. To get a better understanding of the problems, we have identified the six individual bottlenecks in the outbound passenger flow and determined how the corresponding flow management amplifies the problems. The security filter is found to be the most critical bottleneck, causing both the highest number and most severe flow problems. We believe that in the current situation, without dealing with the causes, these operational problems will likely get worse in the near future.

We propose to tackle the amplifying effect of a reactive flow management style by enabling a proactive flow management style. We believe that a proactive flow management style is able to decrease the operational problems by preventing upcoming issues instead of resolving those issues when they already exist. A proactive flow management style can be enabled by providing the stakeholders with real-time short-term forecasts of the passenger flow (3 hours ahead) and additional decision-support.

10.1.2 Data

The second sub-question focused on identifying the current situation regarding (real-time) data within the context of the outbound passenger flow at Eindhoven Airport. Nine data sources were identified that are related to this flow. Of the nine identified sources, three were deemed irrelevant for the specific goals of this research project. Another three sources were unavailable for the course of the project due to our real-time data requirement. For these sources, the challenges and extensive timeliness that are inherent to the development of real-time data streams could not be overcome. We were finally able to use three real-time data sources: AirDCS is the departure control systems (DCS) that contains passenger information, SkyGuide is the flight information system (FIS) and Parkbase contains all parking lot information. During the research we found that it is inherently difficult to set-up real-time data streams for systems that are not designed for this. Furthermore, the extensive dependency on external parties made it challenging and ultimately impossible to finish the research project within the initial agreed period.

10.1.3 Forecast model

The third sub-question aimed at designing a model that is able to forecast the short-term passenger flow at Eindhoven Airport's security filter in real-time. We have used an elaborate literature study in Chapter 2 to identify the different methods that are used for short-term passenger flow forecasting. As there is no literature to be found regarding short-term forecasting within the airport context, we focused on the field of public transport. We found that neural networks are by far the most commonly used method in this context, indicating that neural networks provide high applicability in different settings. Of the different types of neural networks, we identified that recurrent neural networks are most applicable to our context due to its ability to allow information to persist over time and recognize patterns in sequences of data. More specifically, we focus on a gated recurrent unit neural networks, since these have proven to outperform or be on par with their biggest challenger: long short-term memory recurrent neural networks.

In Chapter 6 and Chapter 7 we use the results from the literature study to build a short-term passenger flow forecasting model for Eindhoven Airport's security filter. We have found that we are able to build a model that can forecast the short-term passenger flow with acceptable accuracy. The best forming model is a basic GRU model with a single layer. Furthermore, we have provided scientific insights in the development of a short-term passenger flow forecasting model for the specific airport model and have established that incorporating multiple time-series from different data sources into the forecasting model greatly increases accuracy. This is an important scientific contribution since existing passenger flow forecasting models mainly use a single time-series and that series derivatives. We believe that, by carefully selecting the features, the development steps that were used for the security filter can also be taken for other bottlenecks at Eindhoven Airport to create accurate forecasting models. We also believe that similar models can be used at different airports, by taking the specific airport characteristics into account.

10.1.4 Decision-support model

The fourth sub-question concerned the development of a model that uses the short-term forecasts and real-time data to provide decision-support and enable proactive flow management at Eindhoven Airport's security filter. To ultimately enable this proactive flow management style, Chapter 8 introduces this decision-support model. The decision-support model is three-fold: it provides the decision-makers with a three hour ahead forecast of passengers that will arrive at the security filter, an expected waiting time for the 12 time intervals of 15 minutes within these three hours and finally the model suggests actions that the AOMs can take to mitigate upcoming exploding waiting times. Although the model is currently mainly based on expert knowledge and contains some limitations, we believe it to be a strong start to enable a proactive management flow. With the proper adjustments, we expect the model to be applicable to other bottlenecks at Eindhoven Airport and ultimately also be useful to other airports.

10.1.5 Impact

The last sub-question focused on determining the expected impact of implementing a real-time data driven proactive flow management style. Through a quantitative and qualitative evaluation we have analyzed the (expected) usefulness and (perceived) accuracy of the decision-support model in Chapter 9. The short pilot-phase length limited us, so that we were unable to measure the actual quantitative impact of enabling a proactive flow management on for example the number of queue length exceedances: the decision-support model needs tuning and the AOMs need to get acquainted and build trust before they will actively use the model to perform pro-active actions. However, we have established that the forecast model performs in similar fashion during the pilot-phase as it did during model development. Furthermore, the qualitative evaluation with the AOMs resulted in the conclusion that the decision-support has a lot of potential and will definitely be used after the AOMs establish additional confidence in the model's accurateness. Although the decision-support model is not optimally tuned, during the pilot phase we have already seen occurrences of where the model was able to predict upcoming bottlenecks and proposed actions that were reasonably executable. Furthermore, the AOMs were adamant about using the model even after the pilot phase and it is therefore still available. Concluding, we believe that the positive impact of enabling a proactive flow management style at the security filter is high. We also believe that the model can be used for the other bottlenecks to further increase the positive impact. Ultimately, the decision-support model can be used to enable proactive flow management for the whole outbound passenger flow at Eindhoven Airport in order to reduce the severity and number of bottlenecks experienced.

10.2 Recommendations

A practical (business) goal for Eindhoven Airport was introduced in Section 1.3.1. This goal stated that the search should provide insights on how to reduce the severity and number of bottlenecks experienced in the outbound passenger flow, that are a result of the capacity crunch and that are amplified by inadequate flow management. The insights that were found indicate that enabling a proactive flow management style is one of best courses of action to reduce the number and severity of bottlenecks in the outbound passenger flow. A number of recommendations for Eindhoven Airport are found that can aid this goal.

10.2.1 Deploy and improve decision-support model

Passengers that go through the security filter at Eindhoven Airport currently too often experience unacceptable queue lengths. The decision-support model that was constructed throughout this research project aims to predict these situations in order to resolve them before they occur. We therefore recommend Eindhoven Airport to directly start using the model in the day-to-day operation. During initial deployment, the action proposals will probably not directly be used by the decision makers. However, by already using the dashboard, they can get a feeling of when and why the computed waiting times are correct/incorrect and when the action proposals make sense. This evaluation can then be used to improve the model by increasing the accuracy of especially the waiting time calculations and action proposals. This will lead to an increase in the confidence in the model's accuracy and that in turn will lead to greater use of the model (and its action proposals), thus enabling a proactive flow management.

Also, EA needs to put effort in improving the model to allow for more user control and adaptability. Currently, the model does not take executed actions into account when updating the forecasts, the expected waiting times and the action proposals. Also, as found in the pilot-phase, the users requested to have more control over for example the current throughput of the lanes. The model requires these improvements in user control and adaptability to increase the usefulness.

Finally, after the model has been deployed for an extensive amount of time, additional improvements can be made by analyzing its usage. The dashboard allows the AOMs to record the actions that they take, which can be used to analyze what actions are taken during which situations and what their impact is, to make the action proposal more intelligent. This should result in better and increasingly data-driven action proposals. If desirable, a transition can be made from a model that merely suggests actions and in which the AOM decides to a model that decides for the AOM.

10.2.2 Improve forecast model

The final forecast model that was constructed relied on a data set of 8.5 months and thus did not include a full year of data. When more data becomes available, the forecast model needs to be re-evaluated to see if it is still the best choice. Other features can possibly be added to for example include the different seasonality trends. Furthermore, we recommend the airport to rerun the training algorithm of Section 7.3.4 every month so as to recalculate the optimal HP's and let the model determine the optimal weights and biases.

10.2.3 Increase real-time data availability

Enabling a proactive flow management style depends heavily on the availability of real-time data. As we experienced during this research project, the extensive dependency on external parties to access this type of data created several hurdles. To increase data availability and with that data usage, we recommend that EA to continue its efforts in setting up an integration platform that controls the flow of data between all software systems. This will result in a decrease in its dependency on third parties and an increase in data control. Furthermore, we recommend EA to invest in data-sharing relationships with external third-party companies, such as the public transport bus company Hermes. Hermes should be able to supply EA with real-time data concerning the number of passengers that embark on a bus at Eindhoven Central Station that is destined for EA. Increasing data-availability and control can lead to an increased number of opportunities to enable proactive flow management.

10.2.4 Apply model to other bottlenecks

The developed model was specifically designed for Eindhoven Airport's security filter. However, as already mentioned, we believe that the model is highly applicable to other bottlenecks. Similar forecast models, waiting time calculations and action proposals can be used. We recommend EA to start the process of enabling proactive flow management by using a similar research process at other bottlenecks after the current model has been generally accepted and is actively used.

10.2.5 Interconnect decision-models

A next recommended step is interconnecting the decision-models to each other to enable an all-embracing proactive flow management style. As illustrated in the problem exemplification of Section 1.2.3, the different bottlenecks and general passenger flow are all highly connected. By not treating the bottlenecks as independent, the forecasts, action proposals and expected effects of taken actions can all be linked together. As an example, recording the action of adding extra staff members at the check-in counters should result in higher forecasts for the near-future at the security filter due to a suddenly higher expected influx of passengers. Furthermore, the model for example needs to make sure that actions that are proposed do not form bottlenecks at other locations in the passenger flow if executed. Such an integrated system will ultimately have the capability to create a near perfect passenger flow at Eindhoven Airport.

10.3 Limitations

The research is subject to several limitations. First of all, as indicated, the used data set contains about 8.5 months of data. Since it concerns time-series data with seasonal trends, a data set of at least two years is desirable. Due to company policies, the data was not available for that time-frame and the research was therefore limited to the 8.5 month data set. This limits the generalizing capabilities of the models. Since the required data is actively stored now, the models need to be reevaluated when more data is available.

Another limitation lies in the validation of the decision-support model. The planned pilot-phase of 6 weeks had to be cut short to 2.5 weeks due to multiple last-minute issues with data availability and data-stream stability. This limited the evaluation of the model to a qualitative evaluation in which the expected usefulness and current perceived accurateness were assessed through interviews with the stakeholders. For better model evaluation, quantitative evaluation is required that is able to determine the actual usage of the model and the impact on the severity and number of bottlenecks that are experienced.

A third limitation of the research is the assumed generalizability of the decision-support model. The practical goal of the research was to provide insights on how to reduce the severity and number of bottlenecks experienced in the whole outbound passenger flow. However, we solely focused on the security filter and made the assumption that the model is also applicable to the rest of the outbound passenger flow. Additional research and model building is required to support this claim. This limitation is a result of the time constraints that are imposed on the research project.

Finally, the decision-support model relies on the expected waiting-time calculations considerably. These calculations are based on theoretical models that are adapted from another research paper. When using the dashboard, the stakeholders are expected to generally first look at the expected waiting time and use the passenger forecasts as a secondary information source. However, there is currently no data-driven method to verifying these waiting times. The airport's current sole option to measure the waiting times is by tracking an individual passenger using security cameras and a stopwatch to see how long it takes. This limits the validation options of this element of the decision-support model. As indicated, adaptions of the waiting time calculations are required to make sure they match the real-world situation, which is thus currently merely based on experience of the stakeholders.

10.4 Future research

The scientific goal of the research entailed providing insights in the usage of real-time short-term passenger flow forecasting in an airport terminal environment and providing more general insights in additional decision-support that collectively enable a proactive flow management style. In the process of accomplishing this goal, several research directions were found that can be explored in the future.

During development of the forecasting model, multiple input features were used to predict a single output time-series using GRUs and LSTMs. Future research could expand these RNNs by incorporating multiple output time-series. These models can then be used to predict the short-term passenger flow at different bottlenecks at the same time whilst also incorporating the dependencies between the output time-series.

Moreover, the research has primarily focused on GRUs and LSTMs. Although these structures have proven to be accurate, other neural network and machine learning architectures can be researched to see if there are alternatives that are even better at incorporating the different temporal links between the data sources and their time series.

Furthermore, to compute the expected waiting times during the upcoming time intervals, the model as introduced in Stolletz (2011) was adapted to include the independence of the service times across time intervals. This lead to the adaption of their *M(t)/G/c(t)* model to a *M(t)/G(t)/c(t)* model. Due to the absence of data on the actual waiting times, this adapted model has not been validated. Future research is required to verify the applicability of this model.

Finally, the proposal of actions in the current model are mainly based on expert knowledge. This part of the research is therefore rather explorative. Future research could improve these proposals by using more elaborate methods and thoroughly evaluating the effects of the actions. Also, the action proposal research can be taken a step further by incorporating dependencies between multiple bottlenecks and their proposed and taken actions.

Bibliography

Ackoff, R. L. (1981). The Art and Science of Mess Management. *Interfaces*, 11(1):20–26. 27

Adacher, L., Flamini, M., Guaita, M., and Romano, E. (2017). A model to optimize the airport terminal departure operations. *Transportation Research Procedia*, 27:53–60. 1, 37

Alodhaibi, S., Burdett, R. L., and Yarlagadda, P. K. (2016). Framework for Airport Outbound Passenger Flow Modelling. *Procedia Engineering*, 174:1100–1109. 1, 37

Anvari, S., Tuna, S., Canci, M., and Turkay, M. (2016). Automated Box-Jenkins forecasting tool with an application for passenger demand in urban rail systems. *Journal of Advanced Transportation*, 50(1):25–49. 17

Ben Taieb, S., Bontempi, G., Atiya, A. F., and Sorjamaa, A. (2012). A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Systems with Applications*, 39(8):7067–7083. 54

Bergstra, J. and Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization Yoshua Bengio. *Journal of Machine Learning Research*, 13:281–305. 57

Bontempi, G., Ben Taieb, S., and Le Borgne, Y.-A. (2012). LNBIP 138 - Machine Learning Strategies for Time Series Forecasting. In *Business Intelligence*, pages 62–77. Springer, Berlin, Heidelberg. 54

Breiman, L. (2001). Random Forests. Technical report, University ofCalifornia, Berkeley. 54

Cheng, H., Tan, P. N., Gao, J., and Scripps, J. (2006). Multistep-ahead time series prediction. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 3918 LNAI, pages 765–774. Springer, Berlin, Heidelberg. 17

Chiang, P. N. and Taaffe, K. (2014). Analysis of passenger flow in airport terminal. *Proceedings - 2014 10th International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IIH-MSP 2014*, pages 102–105. 1, 37

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics. 21

Chung, J., Gulcehre, C., and Cho, K. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *NIPS 2014 Deep Learning and Representation Learning Workshop.* 14, 18, 59

Ding, C., Wang, D., Ma, X., and Li, H. (2016). Predicting short-term subway ridership and prioritizing its influential factors using gradient boosting decision trees. *Sustainability (Switzerland)*, 8(11):1–16. 16, 24

Fandango, A. and Wiegand, R. P. (2018). Towards investigation of iterative strategy for data mining of short-term traffic flow with Recurrent Neural Networks. *Proceedings of the 2nd International Conference on Information System and Data Mining.* 23

Frank, R. J., Davey, N., and Hunt, S. P. (2001). Time Series Prediction and Neural Networks. *Journal of Intelligent and Robotic Systems*, 31. 19

Fu, R., Zhang, Z., and Li, L. (2016). Using LSTM and GRU Neural Network Methods for Traffic Flow Prediction. *31st Youth Academic Annual Conference of Chinese Association of Automation*, pages 5–9. 23

Gatersleben, M. R. (1999). Analysis and Simulation of Passenger Flows in an Airport Terminal. *Winter Simulation Conference*, pages 1227–1321. 1, 37

Gers, F. A., Urgen Schmidhuber, J. J., and Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, 12:2451–2471. 21

Gong, M., Fei, X., Wang, Z., and Qiu, Y. (2014). Sequential Framework for Short-Term Passenger Flow Prediction at Bus Stop. *Transportation Research Record: Journal of the Transportation Research Board*, 2417:58–66. 15

Google ML Engine (2019). Using Hyperparameter Tuning — Cloud ML Engine for TensorFlow — Google Cloud. 58

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2017). TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 1 LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems (Volume: 28 , Issue: 10 , Oct. 2017)*. 20

Guyon, I. and De, A. M. (2003). An Introduction to Variable and Feature Selection André Elisseeff. *Journal of Machine Learning Research*, 3:1157–1182. 48

Heaton, J. (2008). *Introduction to neural networks with Java*. Heaton Research. 57

Hee, K. J. and Zeph, Y. C. (1998). An airport passenger terminal simulator: A planning and design tool. *Simulation Practice and Theory*, 6(4):387–396. 1, 37

Hochreiter, J. (1991). *DIPLOMARBEIT IM FACH INFORMATIK Untersuchungen zu dynamischen neuronalen Netzen*. PhD thesis, Institut fur Informatik, Technische Universitat Munchen. 20

Hochreiter, S. and Urgen Schmidhuber, J. J. (1997). Long Short-Term Memory. *MEMORY Neural Computation*, 9(8):1735–1780. 20

Hornik, K. (2017). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2). 56

Ian Goodfellow, Yoshua Bengio, A. C. (2017). The Deep Learning Book. *MIT Press*, 521(7553):785. 22

IBM (2011). IBM SPSS Modeler CRISP-DM Guide. *IBM Systems Journal*. 9, 28

Jao, C. S. (2010). *Decision Support Systems*. Intech. 67

Jason Brownlee (2017). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. 56

Ji, J. and Hou, J. (2018). Forecast on bus trip demand based on ARIMA models and gated recurrent unit neural networks. *2017 International Conference on Computer Systems, Electronics and Control, ICCSEC 2017*, pages 105–108. 14, 23

Jiang, X., Zhang, L., and Chen, X. (2014). Short-term forecasting of high-speed rail demand: A hybrid approach combining ensemble empirical mode decomposition and gray support vector machine with real-world applications in China. *TRANSPORTATION RESEARCH PART C*, 44:110–127. 16

Jiao, P., Li, R., Sun, T., Hou, Z., and Ibrahim, A. (2016). Three Revised Kalman Filtering Models for Short-Term Rail Transit Passenger Flow Prediction. *Mathematical Problems in Engineering*, 2016. 15

Jin, J., Wang, Y. H., and Li, M. (2013). Prediction of the Metro Section Passenger Flow Based on Time-Space Characteristic. *Applied Mechanics and Materials*, 397-400:1038–1044. 13

Jozefowicz, R. and Zaremba, W. (2015). An Empirical Exploration of Recurrent Network Architectures. In *ICML'15 Proceedings of the 32nd International Conference on International*, pages 2342–2350. Google Inc, New York University, Facebook. 18

Kingma, D. P. and Lei Ba, J. (2017). ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. In *ICLR 2015*. 55, 56

Leng, B., Zeng, J., Xiong, Z., Lv, W., and Wan, Y. (2013). Probability tree based passenger flow prediction and its application to the Beijing subway system. *Front. Comput. Sci*, 7(2). 16

Li, Y., Wang, X., Sun, S., Ma, X., and Lu, G. (2017). Forecasting short-term subway passenger flow under special events scenarios using multiscale radial basis function networks. *Transportation Research Part C: Emerging Technologies*, 77:306–328. 13

Ma, Y., Li, Q., and Yan, T. (2015). An intelligent bus dispatch system based on fuzzy genetic algorithms. *11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015)*, pages 1–5. 16

Ma, Z., Xing, J., Mesbah, M., and Ferreira, L. (2014). Predicting short-term bus passenger demand using a pattern hybrid approach. *Transportation Research Part C: Emerging Technologies*, 39:148–163. 14, 15, 62

MarktEffect (2018). Rapport Reizigersmonitor Eindhoven Airport - November 2018. 31, 33, 49

Ni, M., He, Q., and Gao, J. (2017). Forecasting the Subway Passenger Flow under Event Occurrences with Social Media. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1623–1632. 11

Nielsen, M. A. (2015). Neural Networks and Deep Learning. 19

Nmmik, A. and Antov, D. (2017). Modelling Regional Airport Terminal Capacity. *Procedia Engineering*, 178:427–434. 37

Oxford University Press (2018). Definition of public transport in English by Oxford Dictionaries. 11

Roos, J., Gavin, G., and Bonnevay, S. (2017). A dynamic Bayesian network approach to forecast short-term urban rail passenger flows with incomplete data. *Transportation Research Procedia*, 26:53–61. 17

Ruder, S. (2017). An overview of gradient descent optimization algorithms. 55, 56

Schalken, J. and Nefkens, B. (2018). Continu Bezoekersonderzoek 2e halfjaar 2017 - Eindhoven Airport. *Continu Bezoekersonderzoek 2e halfjraar 2017 - Eindhoven Airport*. 3

Shen, G., Tan, Q., Zhang, H., Zeng, P., and Xu, J. (2018). Deep Learning with Gated Recurrent Unit Networks for Financial Sequence Predictions. *8th International Congress of Information and Communication Technology (ICICT-2018).* 23

Sheu, J.-B. (2005). A fuzzy clustering approach to real-time demand-responsive bus dispatching control. *Fuzzy Sets and Systems*, 150(3):437–455. 12

Shirish Keskar, N., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tak, P., and Tang, P. (2017). ON LARGE-BATCH TRAINING FOR DEEP LEARNING: GENERALIZATION GAP AND SHARP MINIMA. In *ICLR 2017*. Northwestern University. 57

Sims, C. A. (1980). Macroeconomics and Reality. *Econometrica*, 48(1):1–48. 54

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. In *NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems*. University of Toronto. 57

Song, Y. and Lu, J. (2018). RNN-based traffic flow prediction for dynamic reversible lane control decision. In *Data Science and Knowledge Engineering for Sensing Decision Support*, pages 323–330. WORLD SCIENTIFIC. 23

Srivastava, N., Hinton, G., Krizhevsky, A., and Salakhutdinov, R. (2014). Dropout :A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958. 19

Stanford University (2018). CS231n Convolutional Neural Networks for Visual Recognition. 56

Stolletz, R. (2011). Analysis of passenger queues at airport terminals. *Research in Transportation Business and Management*, 1(1):144–149. 66, 80

Sugiyama, Y., Matsubara, H., Myojo, S., Tamura, K., and OZAKI, N. (2010). An Approach for Real-time Estimation of Railway Passenger Flow. *Quarterly Report of RTRI*, 51(2):82–88. 12

Sun, Y., Leng, B., and Guan, W. (2015). A novel wavelet-SVM short-time passenger flow prediction in Beijing subway system. *Neurocomputing*, 166:109–121. 16

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In *NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, pages 3104–3112. Google. 54

Teng, J. and Chen, S. (2015). Modified Bus Passenger Flow Forecasting Model Based on Integrating ARIMA with Neural Network. *CICTP 2015.* 13

Tonin, L. (2018). Keras implementation of a sequence to sequence model for time series prediction using an encoder-decoder architecture. 54

Toque, F., Come, E., Oukhellou, L., Trepanier, M., Toqué, F., Côme, E., and Trépanier, M. (2018). Short-Term Multi-Step Ahead Forecasting of Railway Passenger Flows During Special Events With Machine Learning Methods. In *CASPT 2018, Conference on Advanced Systems in Public Transport and TransitData 2018, Jul 2018, Brisbane, Australia*, pages 15–2018. 14, 23, 24, 62

Toque, F., Khouadjia, M., Come, E., Trepanier, M., and Oukhellou, L. (2017). Short & long term forecasting of multimodal transport passenger flows with machine learning methods. *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 560–566. 14

Tsai, T. H., Lee, C. K., and Wei, C. H. (2009). Neural network based temporal feature models for short-term railway passenger demand forecasting. *Expert Systems with Applications*, 36(2 PART 2):3728–3736. 12, 13, 62

Twomey, J. M. and Smith, A. E. (1995). Performance Measures, Consistency, and Power for Artificial Neural Network Models. *Mathl. Comput. Modelling*, 21(2):243–258. 53

van Aken, J., Berends, H., and van der Bij, H. (2012). *Problem Solving In Organizations.* EBSCO Publishing. 9, 27, 30

van Lint, H. J. W. C. and van Hinsbergen, C. C. P. I. J. (2012). *Short-Term Traffic and Travel Time Prediction Models.* Transportation Research Board. 11

Vorhies, W. (2016). CRISP-DM a Standard Methodology to Ensure a Good Outcome - Data Science Central. 28

Vu Pham (2016). Bayesian Optimization for Hyperparameter Tuning - Arimo. 57, 61

Walia, A. S. (2017). Types of Optimization Algorithms used in Neural Networks and Ways to Optimize Gradient Descent. 56

Wei, Y. and Chen, M. C. (2012). Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. *Transportation Research Part C: Emerging Technologies*, 21(1):148–162. 13, 16

Will Koehrsen (2018). A Conceptual Explanation of Bayesian Hyperparameter Optimization for Machine Learning. 57, 61

Wirth, R. and Hipp, J. (2017). CRISP-DM: Towards a Standard Process Model for Data Mining. In *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining.* DaimlerChrysler Research & Technology. 28

Xue, R., Sun, D. J., and Chen, S. (2015). Short-term bus passenger demand prediction based on time series model and interactive multiple model approach. *Discrete Dynamics in Nature and Society*, 2015(i). 15

Yanbing, J., Aihua, W., and Haiying, C. (2007). Simulation and optimization for the airport passenger flow. *2007 International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2007*, pages 6599–6602. 1, 37

Yanjie, J., Liang-peng, G., Xiaoshi, C., and Weihong, G. (2017). Strategies for multi-step-ahead available parking spaces forecasting based on wavelet transform. *J. Cent. South Univ*, 24:1503–1512. 54

Yurii Shevchuk (2016). Hyperparameter optimization for Neural Networks NeuPy. 57

Zhang, D. and Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: a GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7):578–585. 23

Zhu, H., Yang, X., and Wang, Y. (2018). Prediction of Daily Entrance and Exit Passenger Flow of Rail Transit Stations by Deep Learning Method. *Journal of Advanced Transportation, Volume 2018*, 2018. 14, 62

Appendix A

Data sources

Name	**Type**	**Data**	**Notes**
AirDCS	Departure Control System	Passenger information (e.g. name, flight details, baggage allowance)	AirDCS is used for >97% of the flights (Transavia, Ryanair, WizzAir and Corendon)
iPort	Departure Control System	Passenger information (e.g. name, flight details, baggage allowance)	iPort is only used for TUI Netherlands/ Belgium
VIBES	Baggage Handling Management System	Baggage information (e.g. destination, weight, route of baggage) and timestamps (e.g. check-in, x-ray)	
SkyGuide	Flight Information System	Flight information (e.g. expected time of arrival/departure, assigned gate/stand)	
Parkbase	Parking Access Control System	Parking lot information (e.g. reservations, barrier motions incl. timestamps)	
MSCompact	X-ray scanner management system	x-ray information (e.g. timestamps, scan result)	Concerns x-ray scanners at security lanes and x-rays at baggage carousel
G4S Data	Security Lane Log	Security lane throughput information (e.g. throughput per detector per 15 minutes, cause registration of SLA violations)	Manually inputted by employees every 15 minutes. Throughput values are read directly from metal detector
PFM Intelligence	People counting	People count information at the toilet groups (e.g. timestamps, numbers)	

Table A.1: Data sources within the scope of the outbound passenger flow at Eindhoven Airport

Appendix B

Forecast modelling results

	HA			LOCF		
	S1	S2	S3	S1	S2	S3
RMSE						
t + 1	123.97	93.88	79.43	46.13	42.47	34.12
t + 2	123.92	93.86	79.44	63.37	59.70	45.50
t + 3	123.94	93.86	79.44	78.48	75.28	56.28
t + 4	124.01	93.86	79.44	90.96	87.75	64.78
t + 5	124.10	93.83	79.44	101.27	96.70	70.43
t + 6	124.18	93.81	79.44	109.54	103.05	73.88
t + 7	124.23	93.80	79.44	115.56	107.86	75.84
t + 8	124.23	93.79	79.45	119.83	111.50	77.56
t + 9	124.25	93.80	79.46	123.45	115.06	79.88
t + 10	124.27	93.81	79.47	127.25	118.84	82.57
t + 11	124.29	93.83	79.48	131.33	123.22	85.23
t + 12	124.30	93.84	79.49	135.72	127.96	87.25
MAE						
t + 1	105.04	76.92	60.22	32.81	31.38	24.91
t + 2	105.00	76.90	60.24	46.25	44.73	34.55
t + 3	105.02	76.90	60.24	59.01	57.49	43.41
t + 4	105.06	76.90	60.24	69.54	67.56	50.68
t + 5	105.14	76.86	60.23	77.86	74.48	55.88
t + 6	105.21	76.84	60.23	84.26	79.28	59.21
t + 7	105.27	76.81	60.22	88.55	82.39	61.54
t + 8	105.26	76.80	60.24	92.38	84.94	63.45
t + 9	105.28	76.80	60.24	95.75	88.69	65.71
t + 10	105.30	76.82	60.26	99.71	92.57	68.12
t + 11	105.33	76.84	60.28	104.15	97.26	70.22
t + 12	105.35	76.86	60.29	108.87	102.23	71.92

Table B.1: Forecast modelling results for individual training sets (*S1*, *S2* and *S3*). RMSE and MAE for models HA and LOCF.

	VAR			RF		
	S1	S2	S3	S1	S2	S3
RMSE						
t + 1	38.64	37.10	33.05	40.86	36.39	27.68
t + 2	39.09	36.34	33.25	41.50	36.12	28.35
t + 3	39.44	36.48	33.13	42.05	36.26	29.40
t + 4	39.88	36.68	33.16	42.43	36.52	29.69
t + 5	40.41	37.13	33.85	42.72	36.92	29.87
t + 6	41.11	37.55	34.78	42.80	37.12	31.26
t + 7	41.78	38.11	35.88	42.96	37.17	31.99
t + 8	42.09	38.42	36.62	43.15	37.28	33.88
t + 9	42.26	38.56	36.86	43.17	37.42	34.62
t + 10	42.64	38.96	37.00	43.37	37.52	35.25
t + 11	43.93	40.16	38.17	43.71	37.59	37.25
t + 12	47.64	43.80	41.23	44.54	37.97	40.80
MAE						
t + 1	28.28	28.23	25.75	29.49	26.53	21.07
t + 2	28.65	27.68	26.05	29.91	26.16	21.61
t + 3	29.08	27.82	26.02	30.37	26.24	22.34
t + 4	29.66	28.13	26.11	30.57	26.61	22.73
t + 5	30.17	28.69	26.74	30.74	26.89	22.93
t + 6	30.69	29.10	27.52	30.78	27.01	23.92
t + 7	31.36	29.65	28.54	30.91	27.06	24.49
t + 8	31.76	29.99	29.11	31.07	27.20	26.02
t + 9	32.00	30.16	29.38	31.01	27.25	26.83
t + 10	32.33	30.42	29.55	31.18	27.33	27.09
t + 11	33.23	31.42	30.44	31.52	27.41	28.72
t + 12	35.89	34.23	32.99	32.19	27.75	31.70

Table B.2: Forecast modelling results for individual training sets (*S1*, *S2* and *S3*). RMSE and MAE for models VAR and RF.

	B-GRU			B-LSTM		
	S1	S2	S3	S1	S2	S3
GD opt. alg.	Adam	Adam	Adam	Adam	Adam	Adam
Learning rate	0.0036	0.0007	0.0001	0.0001	0.0006	0.0073
Look-ahead	12	12	12	12	12	12
Look-back	85	110	528	24	54	257
Numb. layers	1	1	1	1	1	1
Numb. units	72	194	271	94	39	155
Batch size	32	32	32	32	32	32
RMSE						
t + 1	31.25	25.31	26.83	31.40	25.92	27.06
t + 2	30.79	25.40	26.65	31.26	26.20	27.14
t + 3	31.22	25.81	26.77	31.19	26.37	27.38
t + 4	31.75	26.25	27.30	31.53	26.50	28.20
t + 5	31.93	26.85	28.16	31.84	26.95	29.31
t + 6	32.01	27.11	29.01	32.08	27.10	29.97
t + 7	32.16	27.26	29.83	32.24	26.96	30.48
t + 8	32.81	27.26	30.73	32.83	27.12	31.05
t + 9	32.61	27.48	31.45	33.22	27.61	31.59
t + 10	32.48	27.74	32.42	33.77	27.87	32.89
t + 11	34.27	28.02	35.58	34.73	28.40	37.06
t + 12	40.83	29.66	41.66	36.99	30.79	44.61
MAE						
t + 1	22.93	19.28	20.15	23.47	19.64	20.63
t + 2	22.67	19.27	20.20	23.37	19.78	20.65
t + 3	22.97	19.55	20.30	23.40	19.93	20.83
t + 4	23.45	19.80	20.71	23.68	20.01	21.41
t + 5	23.90	20.20	21.37	23.99	20.35	22.26
t + 6	23.99	20.41	22.11	24.16	20.50	22.79
t + 7	24.10	20.45	22.78	24.30	20.45	23.20
t + 8	24.61	20.50	23.50	24.65	20.52	23.72
t + 9	24.58	20.69	24.25	24.92	20.95	24.36
t + 10	24.39	20.92	25.21	25.25	21.12	25.37
t + 11	25.73	21.26	27.78	25.84	21.59	28.35
t + 12	30.77	22.72	32.20	27.70	23.57	33.87

Table B.3: Forecast modelling results for individual training sets (*S1*, *S2* and *S3*). The hyperparameters are found through Bayesian optimization. RMSE and MAE for models B-GRU and B-LSTM.

	ED-GRU			ED-LSTM		
	S1	S2	S3	S1	S2	S3
GD opt. alg.	Adam	Adam	Adam	Adam	Adam	Adam
Learning rate	0.0036	0.0007	0.0001	0.0001	0.0006	0.0073
Look-ahead	12	12	12	12	12	12
Look-back	85	110	528	24	54	257
Numb. layers	1	1	1	1	1	1
Numb. units	72	194	271	94	39	155
Batch size	32	32	32	32	32	32
RMSE						
t + 1	28.68	24.79	26.13	30.33	25.70	29.59
t + 2	28.75	25.36	26.94	30.79	25.95	31.36
t + 3	29.69	26.07	27.50	30.79	26.22	30.37
t + 4	30.26	26.48	28.52	31.93	26.68	34.30
t + 5	30.37	26.44	29.13	33.11	27.23	33.52
t + 6	31.53	26.94	29.39	32.67	27.97	33.89
t + 7	31.18	27.28	29.34	32.07	27.17	34.37
t + 8	30.84	27.13	31.35	33.10	27.12	39.29
t + 9	31.25	27.60	30.89	32.95	27.10	38.38
t + 10	32.27	27.89	31.06	33.33	27.28	36.33
t + 11	32.66	28.20	33.59	36.17	28.29	41.24
t + 12	37.08	30.09	44.67	37.32	29.64	47.42
MAE						
t + 1	21.77	18.64	19.80	22.54	19.35	22.47
t + 2	21.87	19.08	20.31	22.72	19.55	23.66
t + 3	22.42	19.64	20.87	22.69	19.78	22.91
t + 4	22.75	19.92	21.48	23.50	20.24	25.32
t + 5	22.94	19.92	22.13	24.56	20.53	25.04
t + 6	23.80	20.22	22.19	24.15	21.05	25.25
t + 7	23.46	20.41	22.09	23.77	20.53	25.56
t + 8	23.38	20.36	23.63	24.29	20.42	28.34
t + 9	23.60	20.73	23.46	24.73	20.48	28.18
t + 10	24.24	20.83	23.69	24.90	20.54	27.08
t + 11	24.77	21.26	25.57	26.83	21.38	30.35
t + 12	28.06	22.85	33.04	28.27	22.63	34.95

Table B.4: Forecast modelling results for individual training sets (*S1*, *S2* and *S3*). The hyperparameters are found through Bayesian optimization. RMSE and MAE for models ED-GRU and ED-LSTM.

Appendix C

Waiting time calculation

Input variables:

$$\lambda_i = \text{Arrival rate in period i (Pax per 15 minutes)}$$

$$\mu_i = \text{Processing rate per lane in period i (Pax per 15 minutes)}$$

$$c_i = \text{Open lanes in period i}$$

$$CV^2 = \text{Squared coefficient of variation of processing times (default = 35)}$$

Result variables:

$$b_i = \text{Artificial blocking rate in period i}$$

$$\tilde{\lambda}_i = \text{Artificial arrival rate in period i}$$

$$\rho_i = \text{Utilization in period i}$$

Set:

$$\tilde{\lambda}_1 = \lambda_1$$

Calculate:

$$\tilde{\lambda}_{i+1} = \lambda_{i+1} + b_i$$

$$b_i = \tilde{\lambda}_i \cdot \frac{\left(\tilde{\lambda}_i/\mu_i\right)^{c_i}}{c_i! \sum_{k=0}^{c_i} \frac{\left(\tilde{\lambda}_i/\mu_i\right)^k}{k!}}$$

$$\rho_i = \frac{\tilde{\lambda}_i - b_i}{c_i\mu_i} = \frac{\lambda_i + b_{i-1} - b_i}{c_i\mu_i}$$

$$\Pi_{W_i} = \frac{(c_i\rho_i)^{c_i}}{c_i!}\left((1-\rho_i)\sum_{k=0}^{c_i-1}\frac{(c_i\rho_i)^k}{k!} + \frac{(c_i\rho_i)^{c_i}}{c_i!}\right)^{-1}$$

$$E\left[W_i^{M/M/c}\right] = \Pi_{W_i} \cdot \frac{1}{1-\rho_i} \cdot \frac{1}{c_i\mu_i}$$

Output variable (expected waiting time in period i):

$$E\left[W_i^{M/G/c}\right] = \frac{CV^2+1}{2} E\left[W_i^{M/M/c}\right]$$

Appendix D

Action proposal

Timestep	Action1	Action2	Action3	Action4
t	Open crew lane	PA: Security filter for flight X is closed	Extra personnel (internal)	
t + 1	Open crew lane	PA: Security filter for flight X is closed	Extra personnel (internal)	
t + 2	PA: Security filter for flight X is opened early	PA: Security filter for flight X is closed	Open crew lane	
t + 3	PA: Security filter for flight X is opened early	Open crew lane		
t + 4	PA: Security filter for flight X is opened early	Close X lane Y minutes later	Extra personnel (external)	
t + 5	PA: Security filter for flight X is opened early	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)
t + 6	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	
t + 7	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	
t + 8	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	
t + 9	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	
t + 10	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	
t + 11	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	
t + 12	Close X lane Y minutes later	Open X lane Y minutes early	Extra personnel (external)	

Table D.1: Range of possible action proposals per time-step ahead