Explainable Algorithm to Predict Passenger 1 Flow at CVG Airport

Javier Viaña
College of Engineering and Applied Sciences
University of Cincinnati, Cincinnati, OH, 45219

Kavli Institute for Astrophysics and Space Research
Massachusetts Institute of Technology, Cambridge, MA, 02139
Email: vianajr@mit.edu

Kelly Cohen
College of Engineering and Applied Sciences
University of Cincinnati, Cincinnati, OH, 45219
Email: kelly.cohen@uc.edu

Stephen Saunders
Director Innovation & Information Technology
Cincinnati / Northern Kentucky International Airport (CVG)
Kenton County Airport Board
Cincinnati, OH, 45275
Email: ssaunders@cvgairport.com

Naashom Marx
Director of Strategic Innovation - Advanced Mobility
Cincinnati / Northern Kentucky International Airport (CVG)
Kenton County Airport Board
Cincinnati, OH, 45275
Email: nmarx@cvgairport.com

Brian Cobb
Chief Innovation Officer
Cincinnati / Northern Kentucky International Airport (CVG)
Kenton County Airport Board
Cincinnati, OH, 45275
Email: bcobb@cvgairport.com

Submitted [Submission Date]

ABSTRACT

We present an algorithm for the prediction of the incoming passengers at the airport's security checkpoint with a resolution of 15 minutes for the next 2 weeks from the day of the prediction. This is characterized not only for its performance but also from its explainability in the outcomes. The algorithm has been integrated successfully at CVG airport for daily passenger predictions, which ultimately equipped the airport managers with information to perform a data driven deployment of the necessary airport resources. These include from tailored TSA officer schedules and staff allocation to surveillance and supervision tasks. Some of the indirect consequences of this technology are the congestion avoidance, the improvement of overall security at the airport facilities, the reduction of the waiting times, and the enhancement of the passenger experience.

Keywords: Airport Security, Forecast, Explainable AI, Fuzzy Logic, Machine Learning, Passenger Flow

PASSENGER FLOW PREDICTION AT AIRPORTS

Analyzing millions of data points airport operators and government agencies seek to predict passenger arrival times allowing for schedule optimization utilizing real-time data. This promotes an improved airport experience for staff and passengers, minimizing queues and increasing the overall safety of the airport avoiding congestions.

Perhaps one of the added challenges of the passenger forecast problem is the fact that the arrivals of passengers to the airport are actually stochastic discrete events. Several researchers have carried out discrete event simulations to model passenger flows in the airport terminals (1-6).

The passenger flow prediction has been long studied in the past, even in other public access transportations like subways or ground vehicles. Monmousseau et al. 2020 (7) investigated the passenger flow prediction at Paris Charles De Gaulle airport security checkpoints using Long Short-Term Memory neural networks. As it could be expected, NN's have been applied in this field as well, also considering hybrid DL algorithms with stacked autoencoders (SAE-DNN) (8). Xie et al. 2014 (9) studied a seasonal decomposition and least squares support vector regression (LSSVR) for Short-term forecasting of air passengers. Liu et al 2017 (10) leveraged real data of the Sanya Airport from 2008 to 2016 for the same task using a Holt-Winter Seasonal Model combined with a linear regression.

The excellent performance of DNNs, however, relies on opaque abstractions of the often hundreds of hidden layers and millions of parameters that obscure their decision-making process. This leads to the development of black box models with insufficient clarity on how they work. In that regard, the passenger flow prediction problem requires modified machine learning techniques that learn explainable features while maintaining performance, as well as more interpretable, structured causal models. This becomes more evident with the appearance of recent global AI legislations such as the European General Data Protection Regulation (11) that advocates for interpretability over accuracy and is triggering multi-million-dollar lawsuits against countless organizations (12-13).

To address this issue, DARPA launched its explainable artificial intelligence (XAI) program in May 2017 (originally proposed in August 2016), (14). DARPA defines explainable AI as AI systems that can explain their rationale to a human user, characterize their strengths and weaknesses, and convey an understanding of how they will behave in the future (15).

A complex use case that requires explainable architectures for supervision, which is also characterized by its randomness in the data, is the arriving-passengers flow prediction at big public airports. We will be tackling this problem with the explainable, noise-resilient algorithm developed in this research. Indeed, having an accurate prediction of the passengers at the airport is highly valuable, with implications for the entire flight ecosystem of the hub.

FRAMEWORK OF THE PROBLEM

The main objective is to improve passenger experience and decision-making at the airport. To achieve this, we are focused on the prediction of the exact time at which the passengers enter the airport, their flight and profile information. This forecast will allow us to perform a smart allocation of the resources in the complex, from the opening of more lines at the security check in desks (as well as custom working schedules for the officers) to tailored advertisements.

We focused on the passenger flow prediction two weeks in advance with a resolution of 15 minutes. Success has been defined according to specific figures of merit by CVG, which will be covered in the methodology.

The terrible pandemic that we have experienced since the beginning of 2020 has had a great impact on the commercial aviation sector. The multiple movement restrictions and the fear of contagion from the COVID-19 virus or its variants have significantly reduced air traffic. **Figure 1** represents the annual passengers per capita of each state both in 2019 and 2020.

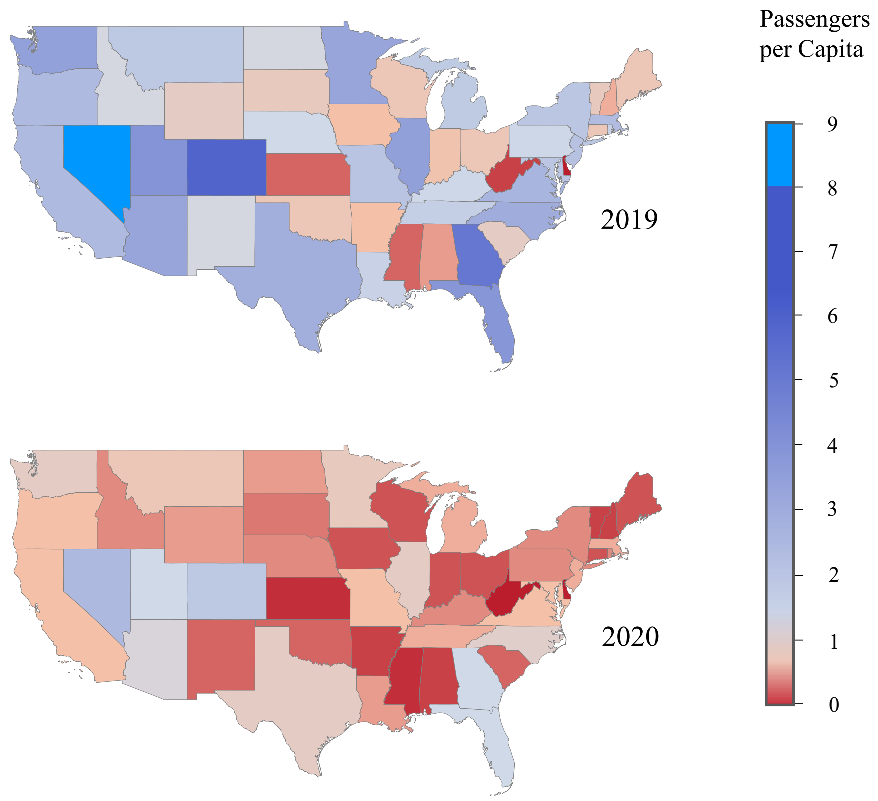


Figure 1. Annual passengers per capita of each state of the Contiguous U.S. in 2019 – 2020.

The data for the plot of **Figure 1** has been obtained from the Bureau of Transportation Statistics (16-17). It can be seen how the majority of the states have less than 1 passenger per capita in 2020, while in 2019 the situation was the opposite.
U.S. airlines carried 557 million less passengers in 2020 than in 2019 (60% reduction from 2019 to 2020). Indeed, passenger traffic in 2020 was the lowest on U.S. airlines since the mid-1980s (17). Because of the fluctuations in passenger flow values, along with low numbers registered during the pandemic, we focused our predictions using pre-pandemic or post-pandemic data. However, the algorithm presented in this section will take into account the dynamic nature of the passenger flow behavior, primarily in two different ways:

- Weighting accordingly the most recent data of the airport, the information from the past few weeks.
- Incorporating the seasonality throughout the year.

In fact, the seasonality of this time-dependent problem is certainly a component that must be considered in the elaboration of a predictor for the passenger arrival. An example of this is covered by Li et al 2017 (18), who proposed a seasonal autoregressive integrated moving average method called SARIMA tailored particularly to Kunming Changshui International Airport.
In **Figure 2**, we can see the seasonality and the yearly increase of the passenger numbers in the U.S. for the pre-pandemic decade, this data was also obtained from the US Bureau of Transportation Statistics. As expected, the summer months June, July and August are the ones with highest records, while January and February are the seasons with least passengers.

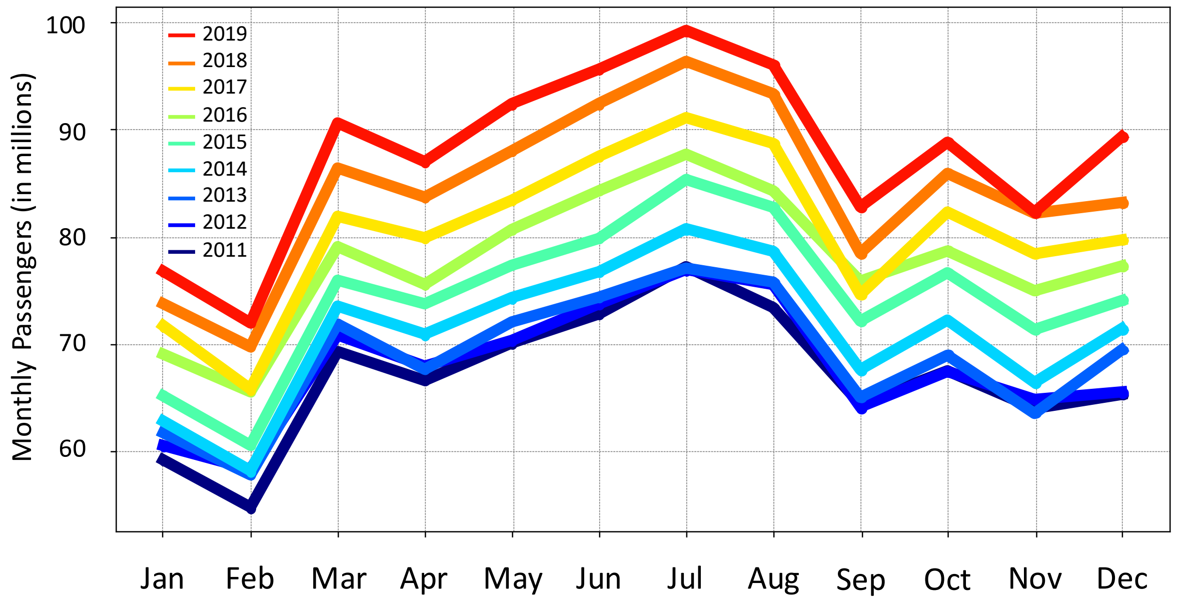


Figure 2. Monthly evolution of the total number of passengers between 2011 and 2019 (pre-pandemic years) in the United States. The seasonal nature of the time series can also be inferred from this plot.

Figure 3 and **Figure 4** show additional statistics for the last two pre-pandemic years concerning both domestic and international flights. We can see that the number of passengers in the U.S. is of the order of 10^9. Not surprisingly, civil aviation is making more than a trillion dollar in the U.S. contributing significantly to the GDP (19). **Figure 4** provides a breakdown of the top 10 carriers in the U.S. using the total number of passengers considering enplaned scheduled systemwide flights. Southwest, Delta, American and United airlines represent the vast majority of the passengers. In the light of these figures, the selected problem is of utmost relevance. An accurate passenger prediction can result in a multi-million-dollar revenue not only for airports and management authorities but also for the airlines and all the stakeholders in the commercial aviation.

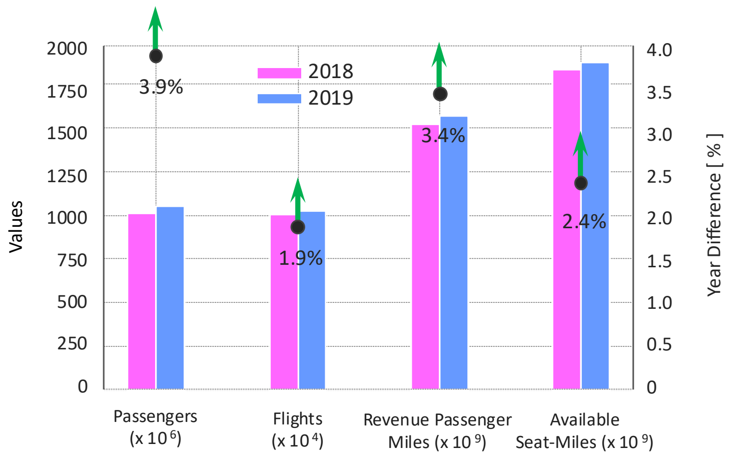


Figure 3. Evolution between 2018 and 2019 (last two pre-pandemic years) of passengers, flights, revenue, and available seat miles (ASM, a measure of an airplane's carrying capacity available to generate revenue) for scheduled systemwide flights (domestic and international).

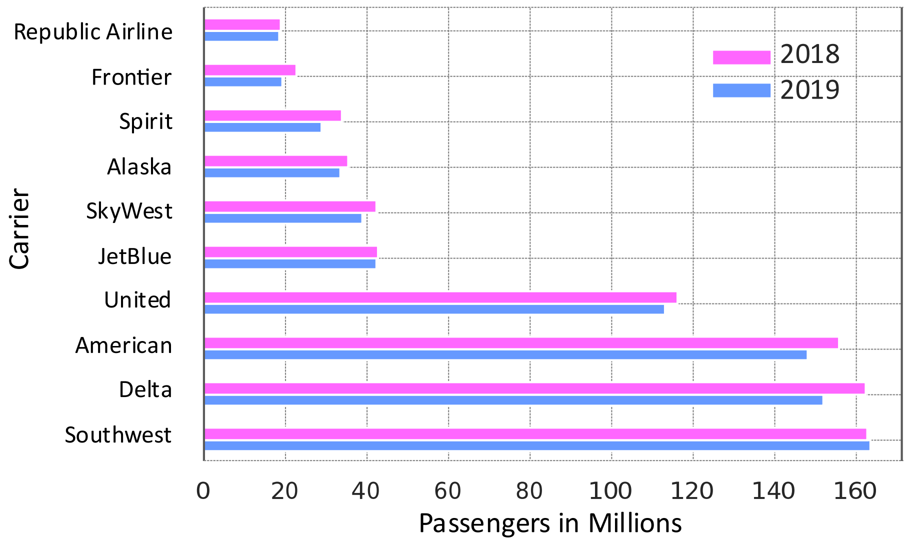


Figure 4. Passenger numbers in millions for the top 10 airlines considering enplaned scheduled systemwide flights (domestic and international flights) in 2018 and 2019 (last two pre-pandemic years).

CVG HARDWARE

CVG Airport Innovation has created a hardware tool to read the non-identifiable information of the passengers' boarding pass at the security checkpoint. The scanners were developed in a joint partnership with DESKO, a company that has been providing airport barcode reading and document scanning and self-service solutions for more than 30 years. Particularly, the scanners are part of the DESKO PENTA OEM product line (**Figure 5**).

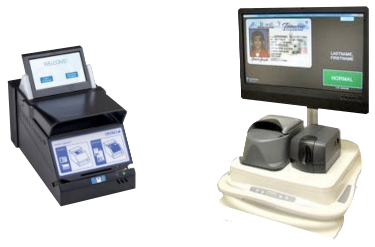


Figure 5. Scanners developed by DESKO for airport barcode reading of non-personal identifiable data. Courtesy of CVG Airport Innovation Office.

CVG Airport has been conducting research and analysis through live deployments of this technology since June 2019 with increasing success. This solution has been used to support other TSA initiatives in 3 other airports (BOS, DCA, TPA) and has proven invaluable in maintaining the fidelity of actionable data in high volumes.

The reality in an airport environment is that:

- passenger behavior is dynamic, and
- micro adjustments at scale can lead to consistent delivery of security mission.

Thus, thanks to this technology, coupled with the algorithm that we present in this paper, CVG will be able to:

- respond to the current environment and to assist with situational awareness and planning using real-time data, and
- have a complete airport consciousness for proactive security responses.

These scanners are owned by the airport, but they are located next to the TSA identity screening counter, at the security checkpoint (**Figure 6**).

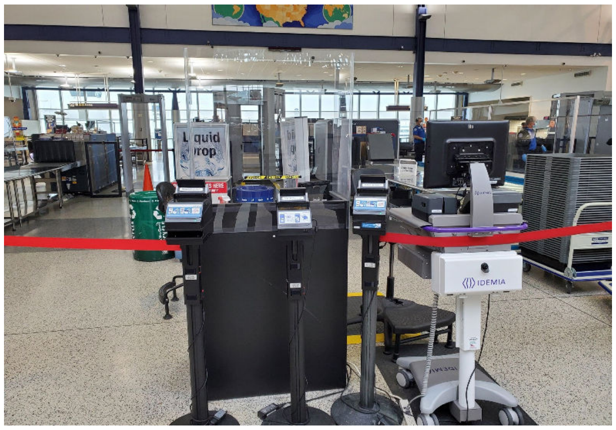


Figure 6. Security checkpoint at the entrance of Cincinnati/Northern Kentucky International Airport. Scanning technology already installed at the airport that is recording the non-identifiable passenger information for prediction of passenger flow. Courtesy of CVG Airport Innovation Office.

Thanks to the added benefit that passenger predictions can generate, TSA has recognized, accepted, and helped in this initiative. Indeed, an accurate flow prediction will ultimately provide managing TSA representatives with an invaluable data-driven decision making.

THE DATA

Inputs

The CVG Airport will provide the data required for this project. There are four main sources of input data for the current stage of the project:

- The departing flight schedules at the airport (data publicly available).
- The scans of the boarding passes at the security checkpoint (data not publicly available).
- Data to extract the seasonality: Total number of passengers of every day throughout the year at the airport (data publicly available).
- Average waiting time at the line of the security checkpoint (data not publicly available).

This data shall be sufficient for prediction of the passenger flow at the security checkpoint. In order to improve the prediction, we will incorporate at a future stage other sources of information, such that the weather forecast, the current traffic (presence of congestions), presence of community events and other external information to the airport. This will be carried out at a later phase of the project.

The departing flight schedules

This data should be tabulated with the following features: Hub time (local time at CVG), destination (with the 3 IATA letter code), market airline, flight number, equipment (vehicle), and total number of seats. The number of bookings for that particular flight is often not available until close to the fight departure time, sometimes even after the departure. Thus, we will not consider it as an input for our predictions, because some airports might not have this information and we want to make a tool that is generalizable for all the U.S. public airports. **Table 1** shows a small sample of how the flight schedule of a given day might look like, particularly the data displayed is for the morning of April 29th 2021 at CVG.

Table 1. Example of a morning flight schedule, schedule shown for April 29th, 2021, at CVG Airport (sample of 10 flights). Courtesy of CVG Airport Innovation Office.

Hub Time	Destination	Market Airline	Flight	Equip.	Seats
0600	DTW : Detroit, MI, US	DL	5027	CR7	69
0600	ATL : Atlanta, GA, US	DL	1626	739	180
0600	DFW : Dallas/Fort Worth, TX, US	AA	4137	E75	76
0600	LGA : New York-La Guardia, NY, US	DL	4852	CR9	70
0610	BWI : Baltimore, MD, US	WN	909	73W	143
0613	PHL : Philadelphia, PA, US	AA	5417	CR9	76
0613	MCO : Orlando, FL, US	DL	1763	738	160
0630	ORD : Chicago-O'Hare, IL, US	AA	4148	E75	76
0630	PIE : St. Petersburg, FL, US	G4	1332	320	186
0631	LAX : Los Angeles, CA, US	DL	1029	321	191

The raw scans of the boarding passes

We will extract only the non-identifiable information of the passengers using the barcode of the IATA Resolution and Guide of Implementation for Bar Coded Boarding Pass (BCBP). The following example is a string of characters that can be inferred from the barcode of the boarding pass:

M1XXXXXXXXXXXXXXXXXXXXEXXXXXXXCVGDENUA 5405 112Y022B0061
15D>5180 K1112BUA 2A01675211312550 UA N*30600 09

Here the X's represent the information deleted for privacy purposes and to protect the identity of the individual. From the remaining set of characters, we can therefore extract the following features: Airline code, baggage tag, nonconsecutive baggage tags, issuance date of the boarding pass, boarding pass issuer, boarding pass issuing carrier, compartment code, departing airport, departure date, destination airport, document type, fast track, flight number, frequent flyer number, international document verification, operating carrier, passenger description, passenger status, and the time stamp of the scan.

For the prediction of the passenger flow we will only use the time stamp at which the scanning occurred, the flight number, the departure date, the destination airport, and the operating airline (to fully identify the flight, as there might be several flights with the same flight number that day at the origin airport, which is unlikely but possible).

At a later stage in the project, we will incorporate the remaining information of the passengers to further characterize the profile of passengers that are arriving at the airport. For now, we will only focus on the actual number of passengers.

For training purposes, we will need to have data for several days in order to make a prediction of the future incoming passenger flow. We will start working with a full month of data from a post-pandemic scenario at CVG.

Seasonality

The seasonality of the passengers over the year refers to the pattern of fluctuations on the number of passengers that is repeated over the years (with minor modifications), we can appreciate such trend in **Figure 2**.

Logically these changes should be considered in order to improve the performance of the algorithm. The way we will approach this correction is by comparing the total daily passengers of the airport over the year and then refactoring the predictions that we are

making. In other words, we will upscale or downscale the predicted profiles to account for the difference between the time of the historic data used for the training and the time for which the prediction is being calculated.
For this task, we will utilize the total daily number of incoming passengers at CVG in the last 3 pre-pandemic years.

Security Line Waiting Time
The purpose of this data is to correct the time that the incoming passengers spend at the security checkpoint line when they arrive to the airport.
There's a difference between the time at which the passengers enter the airport's doors and the time at which they scan the boarding pass at the security checkpoint. This difference is not easy to forecast because it depends on the waiting time of the line before the screening. Nevertheless, CVG airport has already installed technology that allows to estimate the average waiting time at the line of the security checkpoint based on the number of portable devices that have Bluetooth or Wi-Fi activated in the proximity. This is achieved thanks to a set of sensors distributed strategically that track the minutes elapsed between the connection of the devices to each sensor. Future work will focus on the incorporation of this information in the algorithm to increase the performance of the pipeline.

Outputs, The Histogram
The daily passenger flow, and thus the prediction output, is visualized as a histogram with bins of 15 minutes starting from 2:00am of the day for which we are making the prediction and ending 24 hours later (**Figure 7**). Although the CVG security checkpoint is only open from 3:30 a.m. to 8:50 p.m., we decided to provide the full 24-hour picture just in case there are extremal conditions where we get abnormal scans outside that range, such as flight delays that surpass 8:50 p.m. threshold.

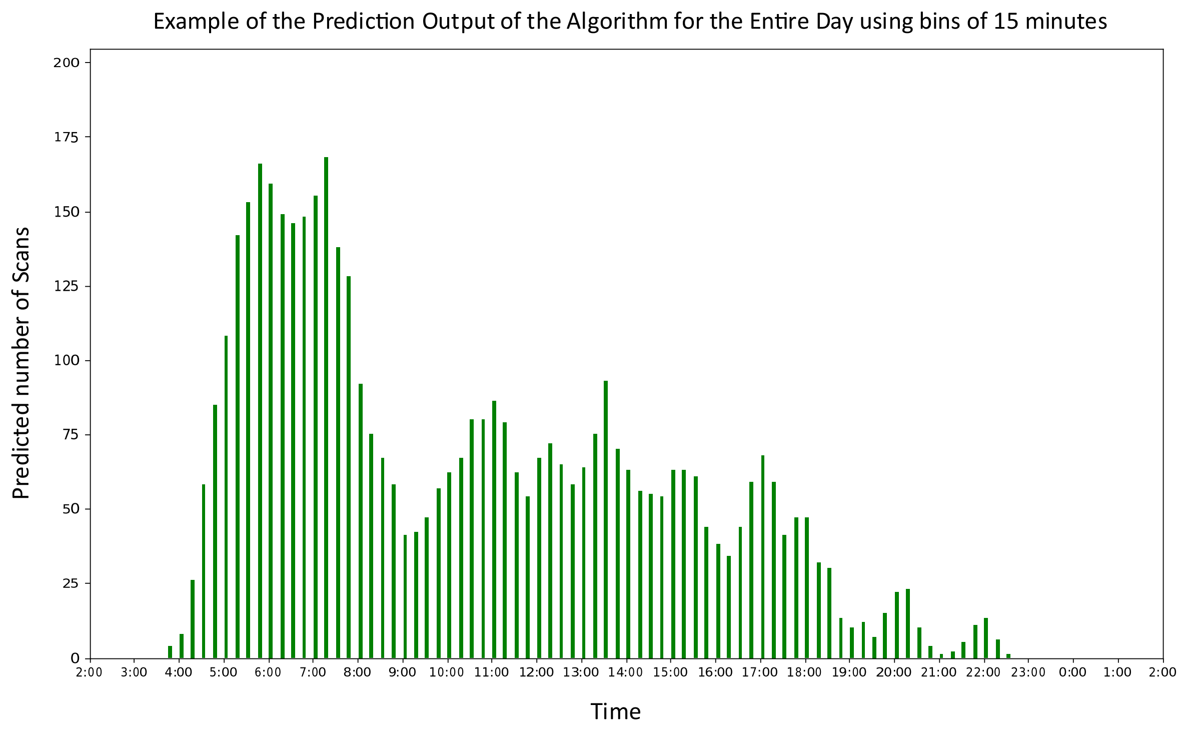


Figure 7. Predicted histogram of incoming passengers' flow at the security checkpoint of CVG airport. Output of the algorithm for a particular day considering a range of 24 hours and 15-minute bins.

The architecture of algorithm is able to make individualized predictions for each flight, and then merge the result to obtain the histogram for the entire day. Thus, the additional prediction of the flow of passengers for each particular flight of the day will also be available. This information will be extremely valuable particularly for the respective airlines, that might as well exploit it for tailored passenger experience and advertisements along the airport.

Data Visualization – Preliminary considerations

We resort to the continuous scan density functions for all the flights of a given day. We utilize a 4-hour window (240 minutes) before the scheduled departure time which we identify as Time Until Take-Off (TUTO). This will serve as a common reference to create a plot where all the scan densify functions can be shown together. We provide these plots for the same weekday throughout four different weeks (**Figure 8**).

Notice that most of the mass is between 30 and 150 minutes in the TUTO scale. However, the data we are working with is very noisy, some of the continuous curves are not as regular as they might seem. In addition to the human factor and the inherent uncertainty of the problem, the lack of scans has also a big impact in the irregularity of the curve's shape (recall that not all the passengers scan their boarding pass in these scanners, its optional).

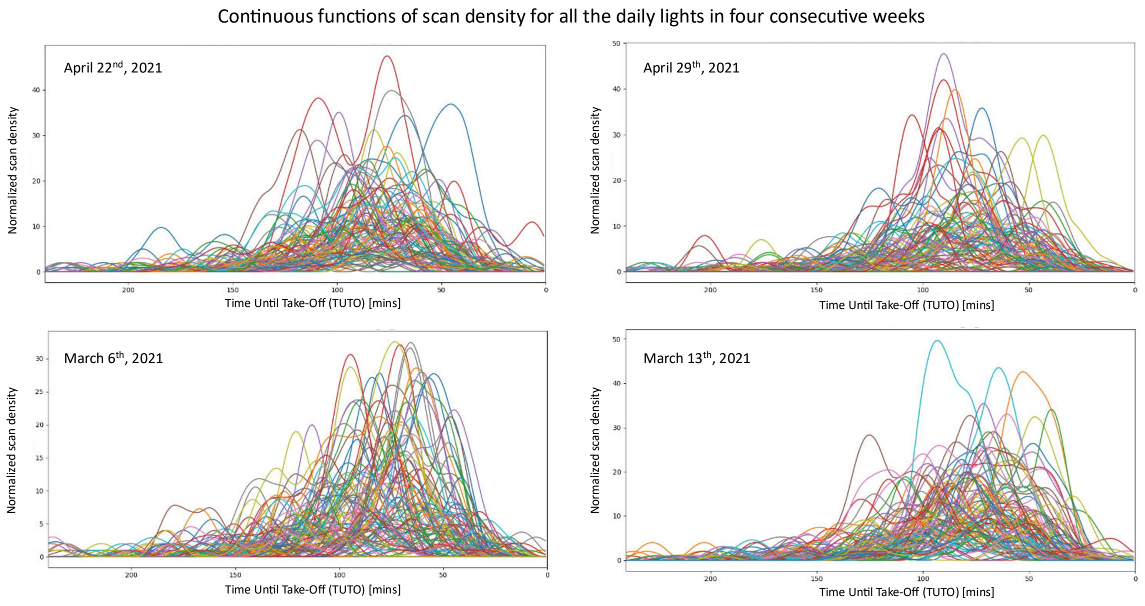


Figure 8. Scan density functions for all the flights of four weekdays shown together using the Time Until Take-Off (TUTO) reference.

Figures of Merit

The goal is to have a prediction of the values for the histogram's bins as close as possible to the real number of scans that were recorded within each 15-minute window of the day.

The prior software installed at CVG was able to calculate the average of number of scans over the last days and make predictions in windows of 30 minutes for the next day. While the present algorithm is a substantial upgrade in the prediction method, the prior baseline serves as a proof of concept.

The figures of merit to measure the performance of the predictor will be the followings:

- Sum of absolute errors of each bar of the entire daily histogram.
- Average absolute error of each bar of the entire daily histogram with respect to the number of 15-minute windows (or histogram bins).

Accuracy Type 1

The first type of accuracy, η_1, is calculated by averaging all the differences of the histogram,

$$\eta_1 = 100\left(1 - \frac{1}{B}\sum_{i=1}^{B}\frac{|y_i - \hat{y}_i|}{\max(y_i,\ \hat{y}_i)}\right) = 100\,(1 - \|y, \hat{y}\|) \tag{1}$$

where η_1 is measured in percentage, B represents the total number of bins in the histogram, $\hat{y}$ is the prediction for a particular bin, y is the ground truth, and $\|y, \hat{y}\|$ is the norm as we define it in **Equation 1**.

Accuracy Type 2

The difference with respect to the previous accuracy definition is the consideration of a penalty for those bins that have greater truth than the prediction. In other words, when the number of scans predicted is lower than the observed scans, the penalty shall be applied. In order to be fair with the overall performance, the opposite scenario will be rewarded in the same proportion,

$$\eta_2 = 100\left(1 - \frac{1}{B}\sum_{i=1}^{B}\frac{w_i\,|y_i - \hat{y}_i|}{\max(y_i,\ \hat{y}_i)}\right) \quad \begin{cases} w_i = 1 + \partial & \text{if } y_i \geq \hat{y}_i \\ w_i = 1 - \partial & \text{else} \end{cases} \tag{2}$$

where ∂ represents the penalty.

METHODOLOGY

The Big Picture

There are several algorithms that we have developed as part of this pipeline, where the common link is transparency. **Figure 9** represents the different steps followed for data processing.

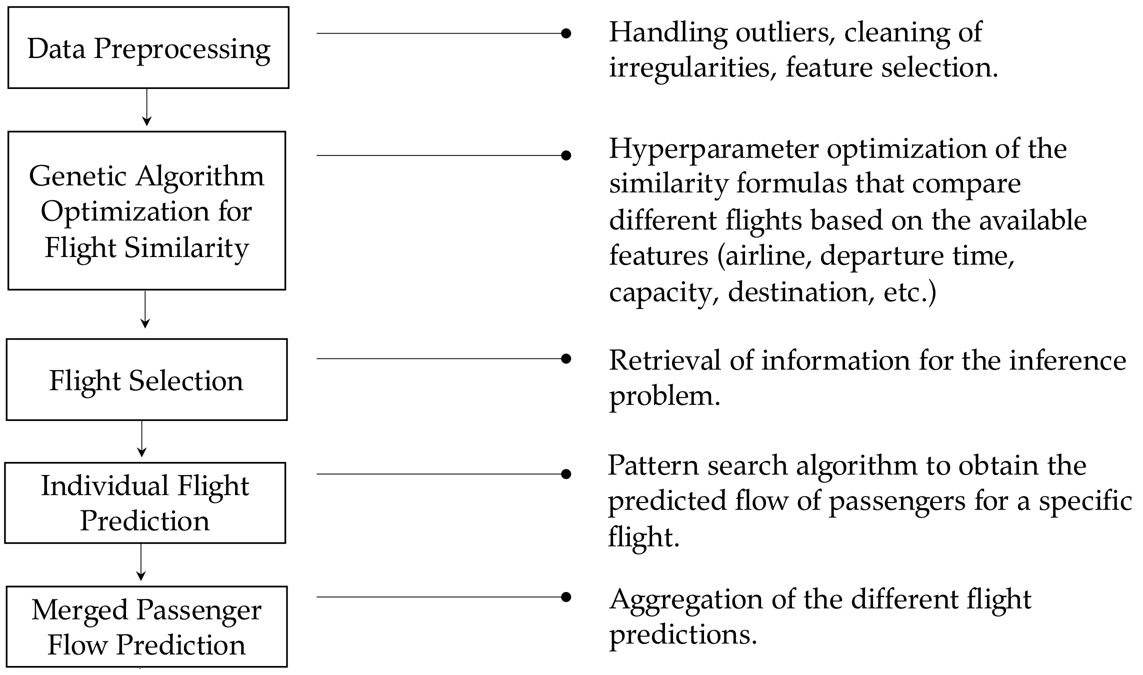

Data Preprocessing
Handling outliers, cleaning of irregularities, feature selection.
Genetic Algorithm Optimization for Flight Similarity
Hyperparameter optimization of the similarity formulas that compare different flights based on the available features (airline, departure time, capacity, destination, etc.)
Flight Selection
Retrieval of information for the inference problem.
Individual Flight Prediction
Pattern search algorithm to obtain the predicted flow of passengers for a specific flight.
Merged Passenger Flow Prediction
Aggregation of the different flight predictions.

Figure 9. High-level block diagram of the data processing pipeline.

Data-Flow

Before making the predictions, a preliminary treatment of the data is more than necessary. This involves several transformations to obtain the continuous passenger flow curves of a specific flight from the raw scans dataset of a given day. **Figure 10** and **Figure 11** represent the evolution of the data to obtain the predicted and ground truth histograms of the total incoming passengers at the security checkpoint. Similarly, the figures also show the Python functions (files with .py extension) in charge of the data transformations at each step.

Each of the operational lines of the CVG security checkpoint is equipped with a scanner that performs the reading of the boarding pass, removes the associated personal information, and finally sends to the database those instances accumulated in their local memory.

The raw scans data file for each day is built as the scanners send information of their readings. At 2:00 a.m. the file is closed and a new one starts. In fact, the raw scans data cannot be treated until the full raw file is populated entirely. After getting it, we filter its entries to obtain the clean data file. At this point we focus on the identification of outliers such as boarding passes where the origin is not CVG, flights that are not for the current date, entries with missing information or unexpected values, etc. We also keep a record file of references to track the modifications carried out. Details such as the fact that IWA and AZA are the same airport are also handled in this transformation.

The pushing of information from the scanners to the database occurs every few seconds (less than 60 seconds between updates). Thus, we can assume that the reception of data is quasi-dynamic. However, the time stamp of raw scans in the database is not completely sorted because of the small delay in the update, the multiple scanners in place, and their asynchronous behavior.

Instead of sorting all the scans in the clean file (which has thousands of entries), we first perform a breakdown of the different flights that were scheduled for that particular day and then we store their information in separate files (these will ultimately be referred as curated data files). Such approached is preferred because the total time required to sort all the flight files is significantly smaller than the sorting of the entire aggregated raw scans.

On a different level, the flight number is not a unique identifier. There could be two different flights with the same flight number that same day. Thus, when performing the flight breakdown, we look at the time, the airline, and the flight number to make sure there are no wrong associations.

What follows is the calculation of the difference between the scan time and the actual departure time of the flight, for each of the flight files. We use the scale of minutes for this computation and we refer to that difference as the relative scan time. In the curated data folders of **Figure 10** and **Figure 11**, we sort the entries using the relative scan time (in descending order), while we keep the scan identification number for reference. Note that because of the quasi-dynamic update (and the simultaneous pushing of data from several devices) there might be cases where the identification numbers are not ordered in time (the greater the id does not necessarily imply that it was scanned later in time).

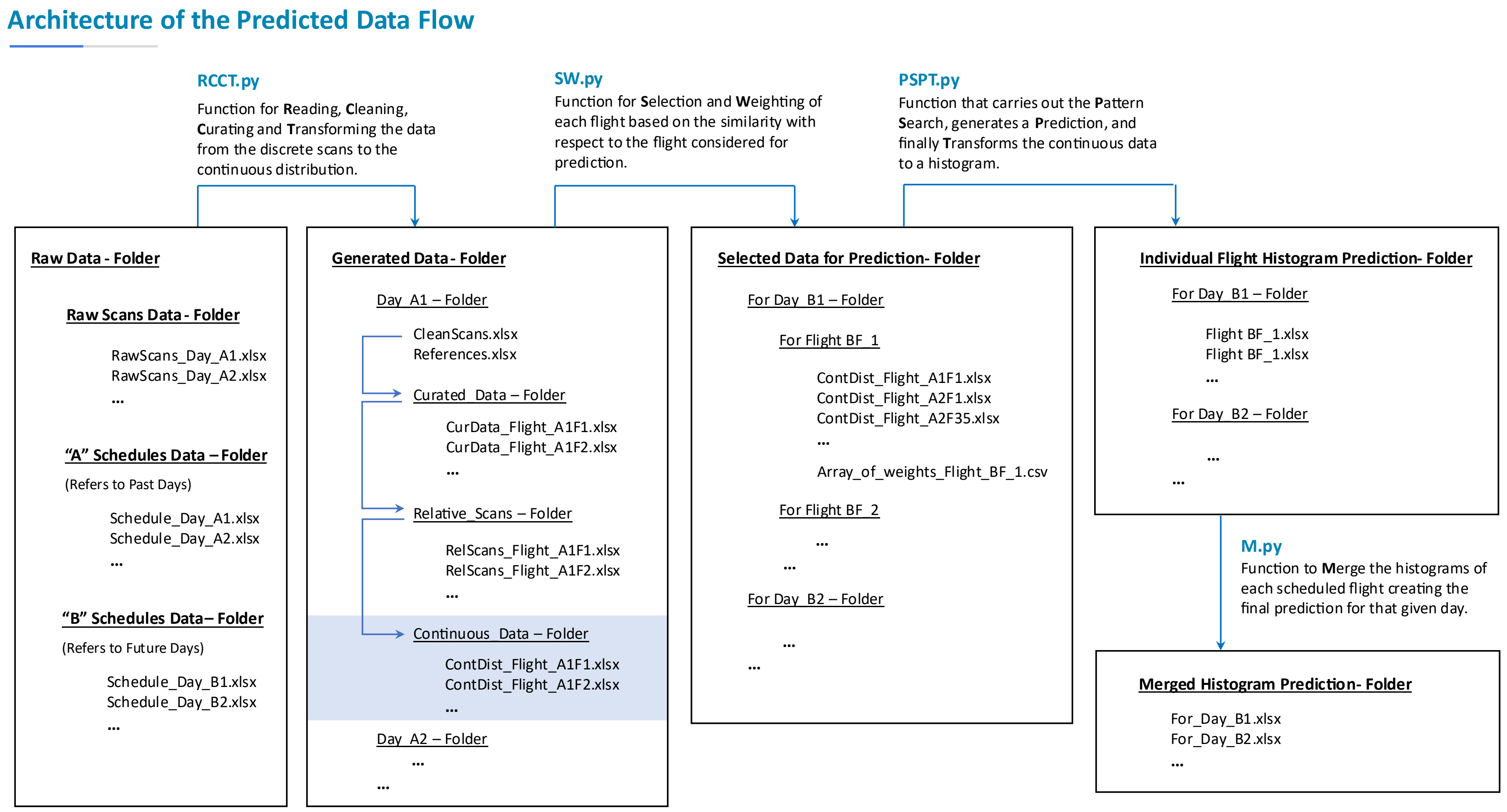

Architecture of the Predicted Data Flow
RCCT.py
Function for Reading, Cleaning, Curating and Transforming the data from the discrete scans to the continuous distribution.
SW.py
Function for Selection and Weighting of each flight based on the similarity with respect to the flight considered for prediction.
PSPT.py
Function that carries out the Pattern Search, generates a Prediction, and finally Transforms the continuous data to a histogram.
Raw Data - Folder
Raw Scans Data - Folder
RawScans_Day_A1.xlsx
RawScans_Day_A2.xlsx
"A" Schedules Data – Folder
(Refers to Past Days)
Schedule_Day_A1.xlsx
Schedule_Day_A2.xlsx
"B" Schedules Data– Folder
(Refers to Future Days)
Schedule_Day_B1.xlsx
Schedule_Day_B2.xlsx
Generated Data - Folder
Day A1 – Folder
CleanScans.xlsx
References.xlsx
Curated Data – Folder
CurData_Flight_A1F1.xlsx
CurData_Flight_A1F2.xlsx
Relative_Scans – Folder
RelScans_Flight_A1F1.xlsx
RelScans_Flight_A1F2.xlsx
Continuous Data – Folder
ContDist_Flight_A1F1.xlsx
ContDist_Flight_A1F2.xlsx
Day A2 – Folder
Selected Data for Prediction- Folder
For Day B1 – Folder
For Flight BF 1
ContDist_Flight_A1F1.xlsx
ContDist_Flight_A2F1.xlsx
ContDist_Flight_A2F35.xlsx
Array_of_weights_Flight_BF_1.csv
For Flight BF 2
For Day B2 – Folder
Individual Flight Histogram Prediction- Folder
For Day B1 – Folder
Flight BF_1.xlsx
Flight BF_1.xlsx
For Day B2 – Folder
M.py
Function to Merge the histograms of each scheduled flight creating the final prediction for that given day.
Merged Histogram Prediction- Folder
For_Day_B1.xlsx
For_Day_B2.xlsx

Figure 10. Flow of data in the prediction architecture. Letter A identifies past days (for which we have scans data) while letter B identifies future days (for which we will be making the prediction). RCCT, SW, PSPT and M are all python functions that make the transformations from one type of data to the next. The following is a dictionary of the abbreviations used: "Rel" means relative, "ContDist" means continuous distribution, and "Wgtd" means weighted.

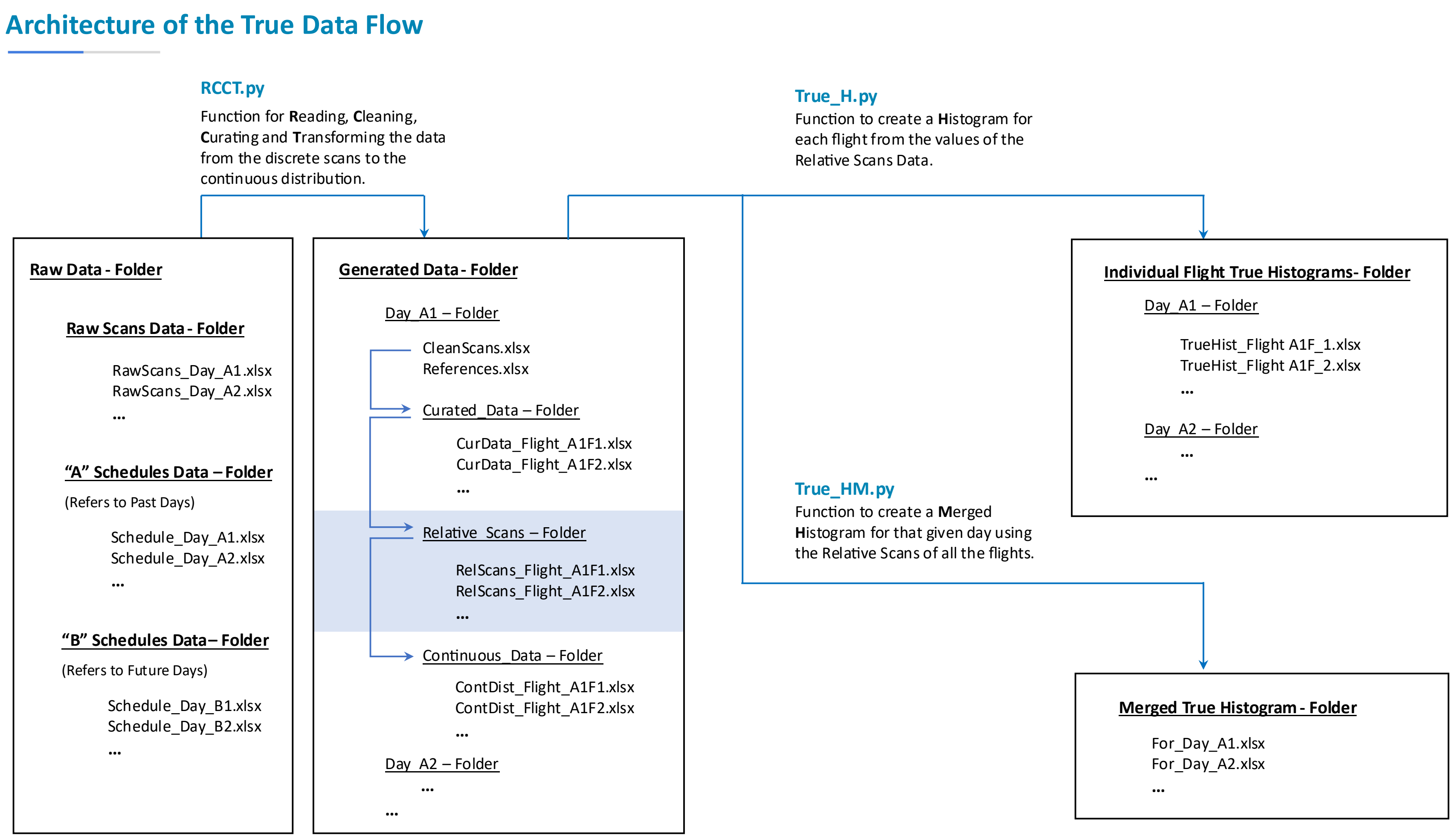

Architecture of the True Data Flow
RCCT.py
Function for Reading, Cleaning, Curating and Transforming the data from the discrete scans to the continuous distribution.
True_H.py
Function to create a Histogram for each flight from the values of the Relative Scans Data.
True_HM.py
Function to create a Merged Histogram for that given day using the Relative Scans of all the flights.
Raw Data - Folder
Raw Scans Data - Folder
RawScans_Day_A1.xlsx
RawScans_Day_A2.xlsx
...
"A" Schedules Data – Folder
(Refers to Past Days)
Schedule_Day_A1.xlsx
Schedule_Day_A2.xlsx
...
"B" Schedules Data– Folder
(Refers to Future Days)
Schedule_Day_B1.xlsx
Schedule_Day_B2.xlsx
...
Generated Data - Folder
Day_A1 – Folder
CleanScans.xlsx
References.xlsx
Curated_Data – Folder
CurData_Flight_A1F1.xlsx
CurData_Flight_A1F2.xlsx
...
Relative_Scans – Folder
RelScans_Flight_A1F1.xlsx
RelScans_Flight_A1F2.xlsx
...
Continuous_Data – Folder
ContDist_Flight_A1F1.xlsx
ContDist_Flight_A1F2.xlsx
...
Day_A2 – Folder
...
...
Individual Flight True Histograms- Folder
Day_A1 – Folder
TrueHist_Flight A1F_1.xlsx
TrueHist_Flight A1F_2.xlsx
...
Day_A2 – Folder
...
...
Merged True Histogram - Folder
For_Day_A1.xlsx
For_Day_A2.xlsx
...

Figure 11. Flow of data to generate the ground truth. Letter A identifies past days (for which we have scans data) while letter B identifies future days (for which we will be making the prediction). RCCT, True_H, and True_HM are all python functions that make the transformations from one type of data to the next. The following is a dictionary of the abbreviations used: "Rel" means relative, and "ContDist" means continuous distribution.

Another consideration in the transformation from the clean data to the curated files is the removal of duplicated instances. In other words, scans that refer to the same person for a given flight. This can occur, for example, in the event that the passenger voluntarily scans the boarding pass more than once, under the wrong belief that the device has not performed a correct reading. This corruption in the data is not necessarily consecutive in time, there can be situations when some other passengers have scanned their boarding passes in between (due to the simultaneous pushing of data or simply as a consequence of exiting and re-entering to the secured area).
The n-plicates (extension of duplicates, in case there are n repetitions) have been removed preserving only the first scan recorded in the system. To remove the n-plicates of a given flight, we look at the seat number for the passengers. If there are several instances with the same seat that is considered a duplicate.
There is one exception to the previous statement: some passengers have no information for their seat. Fortunately, those can be identified when the "passenger status" entry is standby, or the seat is 0, 00, 000, 0000 or empty (we will not consider these cases as duplicated scans). This is not a mistake; it could be that the passenger is an airline employee. Thus, they do not get assigned a seat until the gate. Airline employees are the last to board, and that is the reason why the seat 000 is assigned to them. We will not see a seat assigned for passengers with standby passenger status.

Transformation 1

We now focus on the transformation of the relative scans of each flight to the passenger flow curves that we use for prediction, i.e., converting the discrete events (**Figure 12** left) to continuous distributions (**Figure 12** right). Logically, the separation between two relative scans is not constant. If there is any scan that was obtained passed the take-off time, this scan would be considered as a noisy datapoint. Thus, we focus only on the timeframe until schedule take-off time. The continuous functions are very valuable for the present algorithm as they allow to perform comparisons between flights.

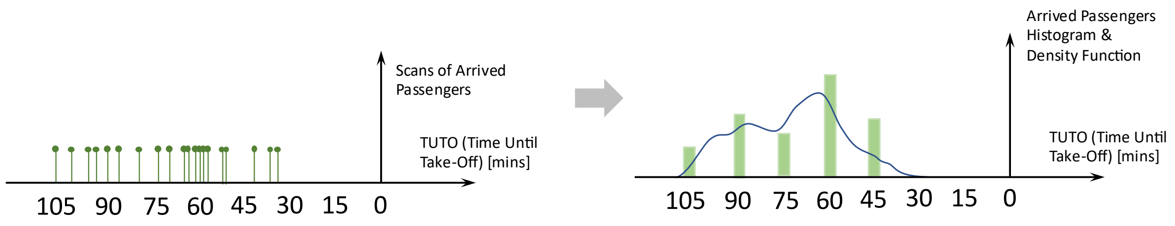

Scans of Arrived Passengers
TUTO (Time Until Take-Off) [mins]
105 90 75 60 45 30 15 0
Arrived Passengers Histogram & Density Function
TUTO (Time Until Take-Off) [mins]
105 90 75 60 45 30 15 0

Figure 12. Graphical representation of the necessity of a transformation from the relative discrete scans to the continuous curve that represents the passenger flow.

The continuous density function, the probability of having a scan at a certain time, and the rate of change in the scanners (how many passengers arrive to the security point every x minutes, i.e., the velocity of passengers) are all directly related to the passenger flow curves that we generate.

- In the first step of this transformation, we require the definition of two parameters: The first is the parameter τ, which specifies the separation between every pair of adjacent points in the time axis of the continuous output. The second is w, the length of the sliding window that will traverse the discrete events, weighting the number of datapoints seen. Note that τ must be smaller or equal to $\frac{w}{2}$.
- For each sliding window, the relative position of a scan is weighted with respect to the center of the window using a sinusoidal distribution as the membership function. This can be seen as a fuzzy membership function or a soft gating strategy.

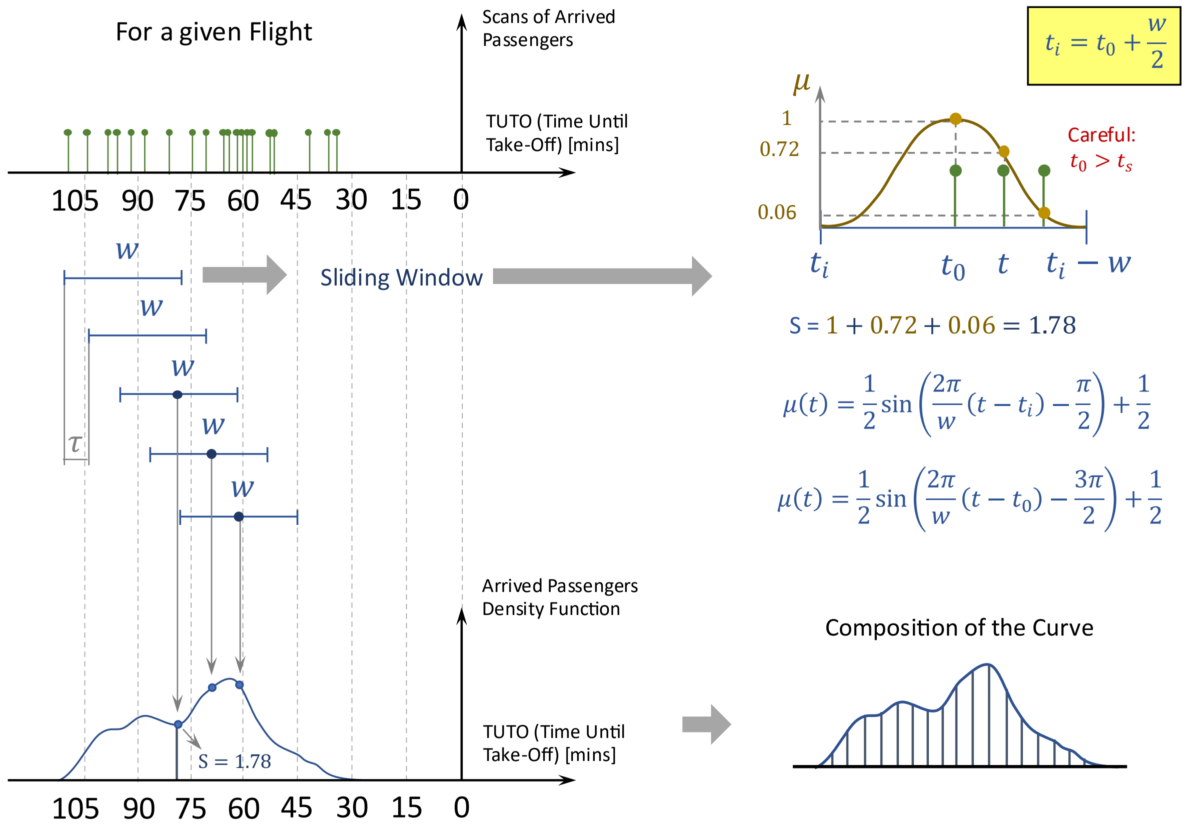

For a given Flight
Scans of Arrived Passengers
TUTO (Time Until Take-Off) [mins]
105 90 75 60 45 30 15 0
w
Sliding Window
τ
Arrived Passengers Density Function
S = 1.78
$t_i = t_0 + \frac{w}{2}$
μ
1
0.72
0.06
Careful: $t_0 > t_s$
t_i t_0 t $t_i - w$
S = 1 + 0.72 + 0.06 = 1.78
$\mu(t) = \frac{1}{2}\sin\left(\frac{2\pi}{w}(t - t_i) - \frac{\pi}{2}\right) + \frac{1}{2}$
$\mu(t) = \frac{1}{2}\sin\left(\frac{2\pi}{w}(t - t_0) - \frac{3\pi}{2}\right) + \frac{1}{2}$
Composition of the Curve

Figure 13. Graphical representation of the sliding window's usage and the composition of the continuous function. Sinusoidal membership function chosen for the fuzzification of the discrete events.

The choice of the sinusoidal membership of function shown in **Figure 13** is further elaborated by Holguin et al. 2022 (20). In order to generate such distribution, the membership function should satisfy the following conditions:

- Tangent to the horizontal axis with a null value in both extremes.
- Bounded within the window.
- With a unitary maximum value and tangent to the horizontal axis in the center.
- Symmetric and smooth.
- With two inflexion points, located at the first and third quarters of the range.
- With antisymmetric behavior in the change from the left corner till its first inflexion point and the change from the inflexion point to the center.

With these constrains, the Gaussian, the Cauchy or any other membership functions that have infinite range in the input dimension are discarded. A sinusoidal membership function, on the other hand, satisfies these requirements. Additionally, we could apply an exponent to each of the values of the sine function to make it flatter in the center or in the extremes at will. In **Figure 14** we are using an exponent that is changing gradually as it approaches the corners of the function to make a flatter center while assuring the tangency with the horizontal axis.

If these factors are increased significantly, we would see how the powered sine degenerates into a step function. Nevertheless, the last of the properties mentioned would be lost when we make the powers of the sine (despite weighting the exponent gradually). Thus, it results in a curve that has more sudden changes and more fragile to noise. To avoid such behavior, we will stick to the sinusoidal membership function.

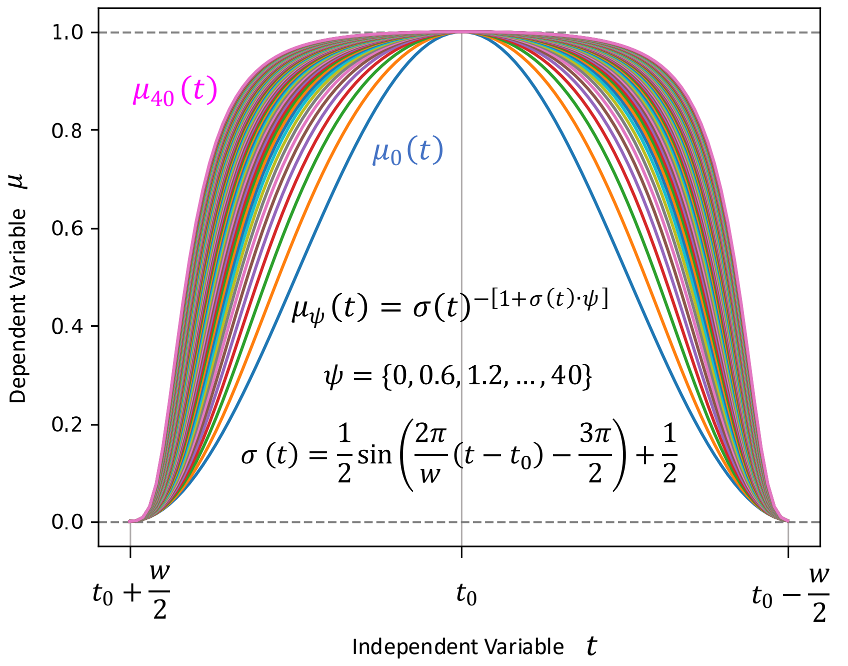


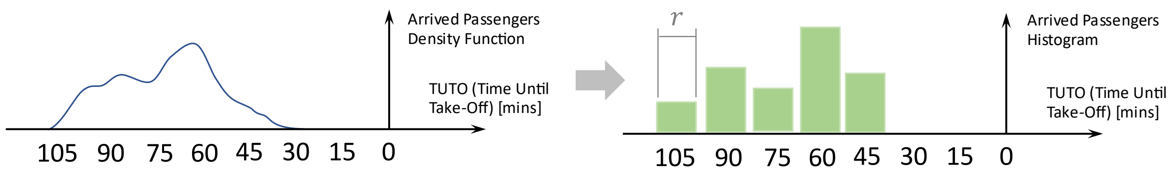
Figure 14. Shape of the different possibilities for the sinusoidal exponential membership function for the range of the sliding window. The curves have been generated ranging the parameter ψ from 0 to 40 in steps of 0.6. The tangency of the extremes and the center is preserved, as well as its symmetric nature and the values of the corners and the middle point.

Transformation 2

Before moving on to the architecture for the prediction, we discuss one last transformation method that will be used at the end of the algorithm. That is precisely the generation of a histogram given a certain smooth function. In fact, it could be seen as the opposite to the previous technique which dealt with the conversion from the discrete events to a continuous function, i.e., we are now going to undo the change.

In order to provide an interpretable and easy-to-understand diagram to the airport authorities, the final output of the algorithm will be the histogram plot with the flow of passengers over time.

The width of the histogram has been fixed at 15 minutes, but in the software we created it could be modified by the user, if needed. The variable that represents this width is identified as r, for resolution. Logically, the value of r has to be greater or equal than τ. **Figure 15** explains graphically the goal of the present task. Similarly, the span of the horizontal axis of all the flights is fixed at 3 hours, which means that if there are any passengers arriving to the airport 3 hours before the scheduled take-off time, they would not be considered. (Although, for the figures we will utilize a smaller scale for simplicity.)



Arrived Passengers Density Function
TUTO (Time Until Take-Off) [mins]
105 90 75 60 45 30 15 0
r
Arrived Passengers Histogram
TUTO (Time Until Take-Off) [mins]
105 90 75 60 45 30 15 0

Figure 15. Graphical representation of the necessity of a transformation from the continuous curve that represents the passenger flow to the histogram with a bin size of r.

To solve this problem, we consider the sum of segments within each "bin" of the histogram (defined by the resolution r). Then, this sum is divided by the total sum of segments under the area, which results into a fraction (**Figure 16**). We can obtain the height of the histogram simply multiplying this fraction by the total number of passengers estimated for the flight.

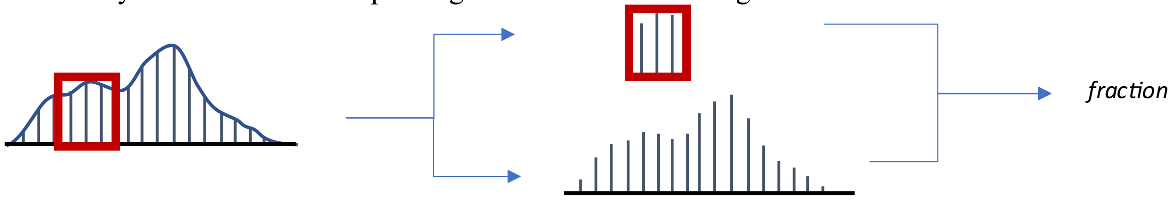


Figure 16. Graphical representation of the calculation for the fraction used to convert the continuous curve to the discrete histogram.

The second part of the transformation to the bar plot deals with the weighting of the histogram's bars based on the valleys and peaks of the curve.
We first study the variance of the segments that are enclosed within a bin. Then, the system is able to add or subtract a fraction of that variance if there is a peak or a valley within the bin, respectively. The variance is also weighted by the proximity of the peak or the valley to the center of the bin, making a zero contribution if these occur in either extreme of the bin and maximum if in the center.
The third and last part of the transformation refers to an iterative algorithm to get to integer predictions in the histogram's bars. In the end, the predictions should not have decimal points, i.e., a fraction of a person has no meaning. This subroutine divides the fractional portion of the bins so that the entire histogram adds up to the total number of people forecasted for that particular flight. Note that this problem would be very simple if that last requirement was not applicable. In other words, we could simply round each bar to the closest integer, but that would result in a total number of people different from the predicted value.
The algorithm starts by rounding the bar whose fractional value is closest to an integer value. The remaining fractional mass (negative or positive) that has been added or subtracted is shared between the two neighbor bars equally (provided that these bars are still fractional). If there is only one neighbor bar then the mass is transmitted fully to this. If there are no neighbor bars with fractional values then the mass is discarded. After applying this to one bar, the same process is applied to all the remaining non-integer bars. The remaining mass with number of people to distribute along the bars should be updated at each step. In the Algorithm 1, we provide a pseudocode for such function, whose time and space complexities are $O(n_{bins}^2)$ and $O(n_{bins})$ respectively, where n_{bins} is the number of bins. The **Algorithm 1** can be further optimized to $O(n_{bins} \cdot \log(n_{bins}))$ time complexity and $O(n_{bins})$ space complexity. We also provide the pseudocode for such reduction in **Algorithm 2**.

Algorithm 1: Part 3 of Transformation 2 with $O(n_{bins}^2)$ time complexity and $O(n_{bins})$ space complexity.

Input: Current non-integer histogram $\boldsymbol{h_0}$, total number of passengers predicted for the flight p_0.

Output: Updated histogram with integer values $\boldsymbol{h_f}$, corrected total number of passengers $p_f = p_0 \pm 1$

```
1  $\boldsymbol{h_f} \leftarrow \boldsymbol{h_0}$;
2  Remaining indexes to convert: $\boldsymbol{k} \leftarrow$ all;
3  while $\boldsymbol{k}$ is not empty do
4      Find index $i$ of bin $h$ from $\boldsymbol{k}$ set: $\min_i \left|\text{round}[\boldsymbol{h_f}(i)] - \boldsymbol{h_f}(i)\right|$;
5      pop $i$ from $\boldsymbol{k}$ set;
6      Find neighbor bins $\boldsymbol{b}$ (max 2) of $\boldsymbol{h_f}(i)$: $\boldsymbol{b} \leftarrow$ neigs[$\boldsymbol{h_f}(i)$];
7      $\boldsymbol{b_k} \leftarrow j$ for index $j$ in $\boldsymbol{b}$ if $j$ in $\boldsymbol{k}$ set;
8      $d \leftarrow$ len($\boldsymbol{b_k}$);
9      if $d$ is not 0 do
10         $m \leftarrow \frac{\{\text{round}[\boldsymbol{h_f}(i)] - \boldsymbol{h_f}(i)\}}{d}$;
11         for each index $j$ in $\boldsymbol{b_k}$ do
12             $\boldsymbol{h_f}(j) \leftarrow \boldsymbol{h_f}(j) + m$;
13         end
14     end
15     $\boldsymbol{h_f}(i) \leftarrow$ round[$\boldsymbol{h_f}(i)$];
16 end
17 $p_f \leftarrow \sum_{l=1}^{n_{bins}} \boldsymbol{h_f}(l)$;
18 return $\boldsymbol{h_f}$, $p_f$
```

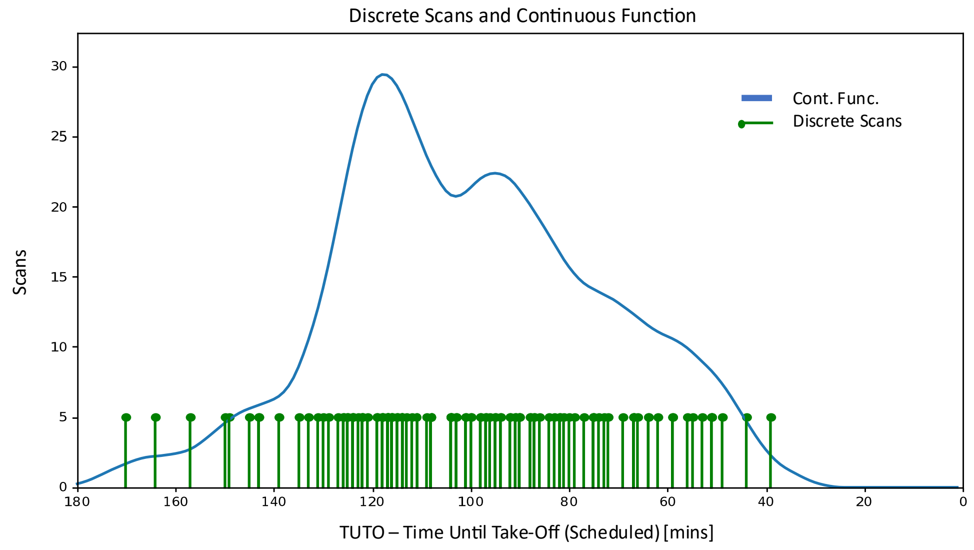
Algorithm 2: Part 3 of Transformation 2 with $O(n_{bins} \cdot \log(n_{bins}))$ time complexity and $O(n_{bins})$ space complexity.

```
Input: Current non-integer histogram $\boldsymbol{h_0}$, total number of passengers predicted for the flight $p_0$.
Output: Updated histogram with integer values $\boldsymbol{h_f}$, corrected total number of passengers $p_f = p_0 \pm 1$

1  $\boldsymbol{h_f} \leftarrow \boldsymbol{h_0}$;
2  Remaining indexes to convert: $\boldsymbol{k} \leftarrow$ all;
3  Get vector of differences: $\boldsymbol{r} \leftarrow |\mathrm{round}[h] - h|$ for $h$ in $\boldsymbol{h_f}$;
4  $\boldsymbol{s} \leftarrow \mathrm{sort}(\boldsymbol{r})$ ;
5  $\boldsymbol{q} \leftarrow$ index of $s$ in $\boldsymbol{r}$ for $s$ in $\boldsymbol{s}$;
6  Get minimum value and index: $s, i \leftarrow s_0, t_0$;
7  pop $i$ from $\boldsymbol{k}$ set;
8  while $\boldsymbol{k}$ is not empty do
9      Find neighbor bins $\boldsymbol{b}$ (max 2) of $\boldsymbol{h_f}(i)$: $\boldsymbol{b} \leftarrow \mathrm{neigs}[\boldsymbol{h_f}(i)]$;
10     $\boldsymbol{b_k} \leftarrow j$ for index $j$ in $\boldsymbol{b}$ if $j$ in $\boldsymbol{k}$ set;
11     $d \leftarrow \mathrm{len}(\boldsymbol{b_k})$;
12     Use an auxiliary index: $a \leftarrow i$;
13     Use a flag: $f \leftarrow$ False;
14     if $d$ is not 0 do
15         $m \leftarrow \frac{s}{d}$;
16         for each index $j$ in $\boldsymbol{b_k}$ do
17             $\boldsymbol{h_f}(j) \leftarrow \boldsymbol{h_f}(j) + m$;
18         end
19         $\boldsymbol{\delta} \leftarrow |\mathrm{round}[\boldsymbol{h_f}(j)] - \boldsymbol{h_f}(j)|$ for each index $j$ in $\boldsymbol{b_k}$;
20         $\zeta \leftarrow \min_{\delta,i} \delta$ for $\delta$ in $\boldsymbol{\delta}$;
21         if $\zeta < s$ do
22             $s \leftarrow \zeta$;
23             $i \leftarrow$ index of $\zeta$ in $\boldsymbol{\delta}$;
24             pop $i$ from $\boldsymbol{k}$ set;
25             $f \leftarrow$ True;
26         end
27     end
28     $\boldsymbol{h_f}(a) \leftarrow \mathrm{round}[\boldsymbol{h_f}(a)]$;
29     if not $f$ do
30         $i \leftarrow$next $q$ in $\boldsymbol{q}$ that is not in $\boldsymbol{k}$ set;
31         $s \leftarrow |\mathrm{round}[\boldsymbol{h_f}(i)] - \boldsymbol{h_f}(i)|$;
32         pop $i$ from $\boldsymbol{k}$ set;
33     end
34 end
35 $p_f \leftarrow \sum_{l=1}^{n_{bins}} \boldsymbol{h_f}(l)$;
36 return $\boldsymbol{h_f}$, $p_f$
```

1

On a further note, the result obtained when we use the 1st and 2nd transformations consecutively, may differ slightly from the true histogram that is generated from the discrete events. This occurs because we used a sliding window in the 1st transformation whereas the calculation of a histogram directly from the scans uses crisp boundaries. Nonetheless, the latter may result into sudden unexpected peaks or valleys in the predicted histogram. The fact that we used the sliding window instead of the crisp boundaries makes the prediction more resilient to noise.
For the scans data shown in **Figure 17**, should the width of the sliding window be chosen wisely, there is nearly no difference between true histogram (calculated using crisp boundaries) and predicted histogram (considering the 1st transformation and all the three parts of the 2nd transformation).



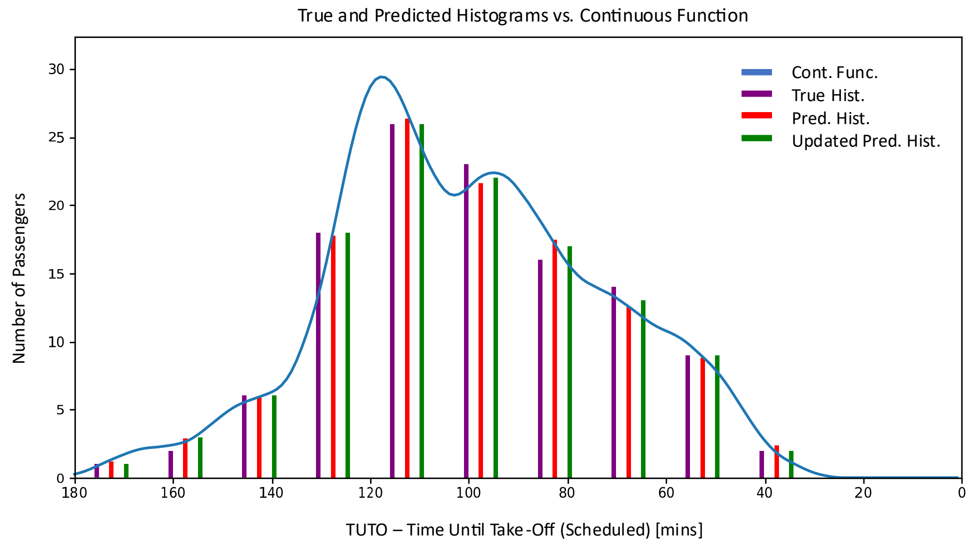
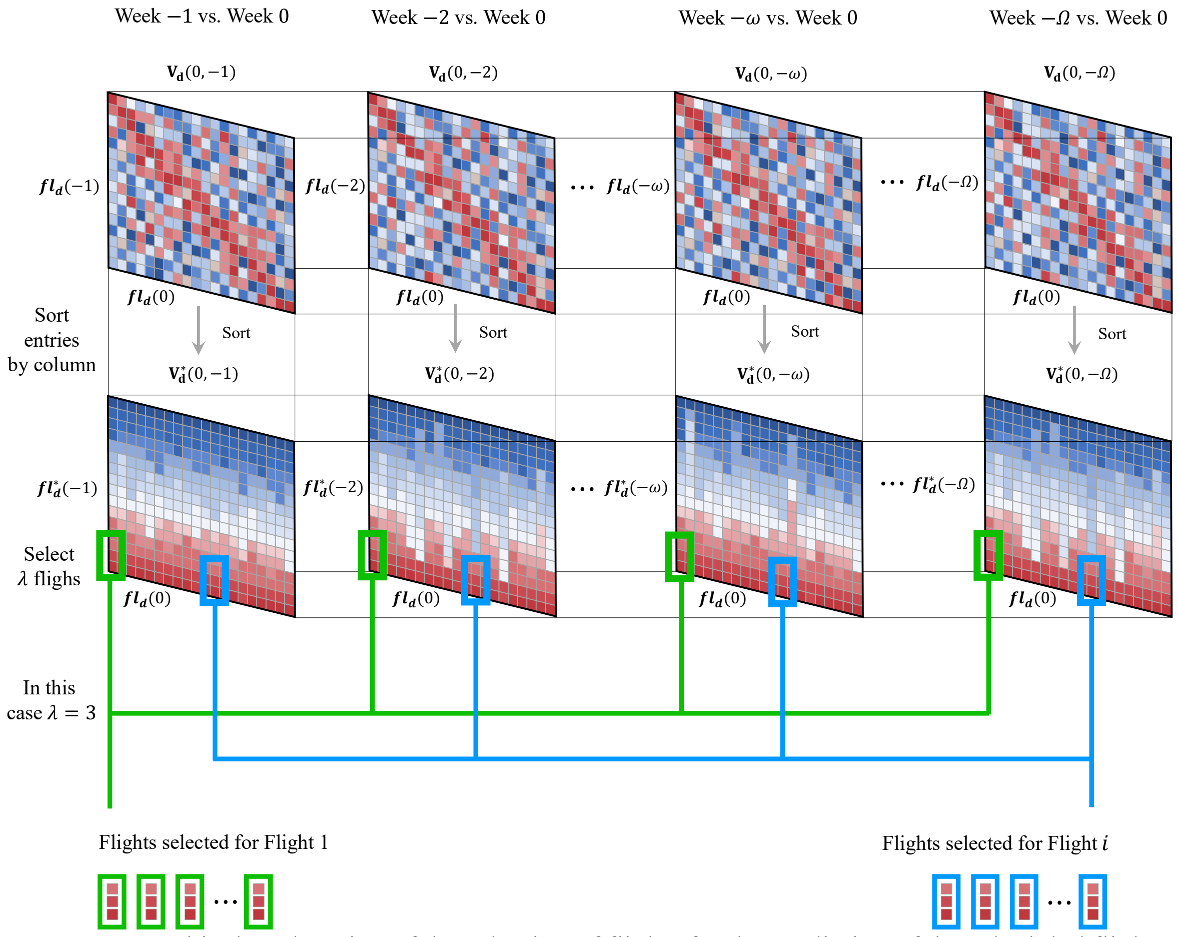


Figure 17. Study of the difference between the predicted histogram and the updated histogram (after applying the 2nd and 3rd parts of the Transformation 2) with respect to the true histogram that was calculated without using any transformation, i.e., from the discrete scans.

Flight Similarity

The following section explains the algorithm that we use for the optimal selection of the past flights that are considered in the prediction of a future flight. Each day of the week will be treated independently. That is, the prediction made for the scheduled flights of a Monday, for example, will be based on the information of the Mondays of the past weeks. We use the index d to refer to a generic day of the week. The algorithm uses the information from the last Ω weeks to select the flights. Logically, we will select those with the greatest similarity with respect to the flight for which the passenger flow prediction is being made. We denote the reference of the flights of the predicted day as $\boldsymbol{fl_d}(0)$ where 0 is the index that identifies the week. Analogously, we denote the reference of the flights of the same day w weeks ago as $\boldsymbol{fl_d}(-\omega)$ (where ω is bounded between -1 and Ω). Then, we build a similarity matrix that compares the flights of $\boldsymbol{fl_d}(0)$ and $\boldsymbol{fl_d}(-\omega)$, namely $\mathbf{V_d}(0,-\omega)$, and we sort the entries of this matrix for each of the scheduled flights, thus obtaining the sorted matrix $\mathbf{V_d^*}(0,-\omega)$. The matrix $\mathbf{V_d^*}(0,-\omega)$ records the past flights that have the closest resemblance to the future flights of $\boldsymbol{fl_d}(0)$. Finally, we extract the first λ flights of $\mathbf{V_d^*}(0,-\omega)$ for each scheduled flight in $\boldsymbol{fl_d}(0)$. Repeating this calculation for the last Ω weeks, we obtain the set of flights that will be used to predict the future passenger flow for each scheduled flight in $\boldsymbol{fl_d}(0)$. A graphical explanation of the prior process is shown in **Figure 18**.



Week -1 vs. Week 0
Week -2 vs. Week 0
Week $-\omega$ vs. Week 0
Week $-\Omega$ vs. Week 0
$\mathbf{V_d}(0,-1)$
$\mathbf{V_d}(0,-2)$
$\mathbf{V_d}(0,-\omega)$
$\mathbf{V_d}(0,-\Omega)$
$fl_d(-1)$
$fl_d(-2)$
$\cdots$ $fl_d(-\omega)$
$\cdots$ $fl_d(-\Omega)$
$fl_d(0)$
Sort entries by column
Sort
$\mathbf{V_d^*}(0,-1)$
$\mathbf{V_d^*}(0,-2)$
$\mathbf{V_d^*}(0,-\omega)$
$\mathbf{V_d^*}(0,-\Omega)$
$fl_d^*(-1)$
$fl_d^*(-2)$
$\cdots$ $fl_d^*(-\omega)$
$\cdots$ $fl_d^*(-\Omega)$
Select λ flighs
In this case $\lambda = 3$
Flights selected for Flight 1
Flights selected for Flight i

Figure 18. Graphical explanation of the selection of flights for the prediction of the scheduled flights, $\boldsymbol{fl_d}(0)$, for a day d of the week, based on the information from the past flights $\boldsymbol{fl_d}(-\omega)$ for $\omega \in [-1,\Omega]$.

For the calculation of the similarity between two flights, the following approach is chosen: The information we have for week 0, for which the prediction is being carried out, is comprised in the input variables of each flight. Among others, scheduled departure time, destination, operating carrier, flight number, number of seats and type of vehicle. Information of bookings is not usually available a week prior to the scheduled departure time, therefore we do not consider it at this point. The formula that we use for the computation of the similarity, based on the most relevant available input features of each flight, is

$$s_{a,b}^{in} = s_{a,b}^{in} = \frac{v_{a,b}^{time} + v_{a,b}^{dest} + v_{a,b}^{carr} + v_{a,b}^{flnum} + v_{a,b}^{seats}}{5} \tag{3}$$

where $s_{a,b}^{in}$ is the similarity score and v denotes each of the values for the 5 variables that we consider in this calculation: departure time ($time$), destination ($dest$), operating carrier ($carr$), flight number ($flnum$), and number of seats ($seats$). Each of these values is defined as,

$$v_{time} = \frac{w_1}{w_2 + |time_a - time_b|} \tag{4}$$

$$v_{dest} = \begin{cases} w_3 & if \quad dest_a = dest_b \\ 0 \end{cases} \tag{5}$$

$$v_{carr} = \begin{cases} w_4 & if \quad carr_a = carr_b \\ 0 \end{cases} \tag{6}$$

$$v_{flnum} = \begin{cases} w_5 & if \quad flnum_a = flnum_b \\ 0 \end{cases} \tag{7}$$

$$v_{seats} = \frac{w_6}{w_7 + |seats_a - seats_b|} \tag{8}$$

where $w_1, w_2, w_3, w_4, w_5, w_6$, and w_7 are weights that need to be chosen.

Next we propose a novel method for the selection of the weights that can also be used in other forecast problems that are similar to the one explained in this section. First of all, we distinguish two types of similarities:

- Similarity based on the input data; the features described above.
- Similarity based on the output passenger flow continuous curves.

The second type of similarity is calculated as

$$s_{a,b}^{out} = s_{a,b}^{out} = \frac{1}{1 + \sqrt{\frac{1}{M}\sum_{i=1}^{M}[f_a(t_i) - f_b(t_i)]^2}} \tag{9}$$

where M is the total number of points that we want to evaluate in the f_a and f_b curves that represent the passenger flow. In other words, we compare the values for the flow of both flights a and b at each t_i, as shown in **Figure 19**.

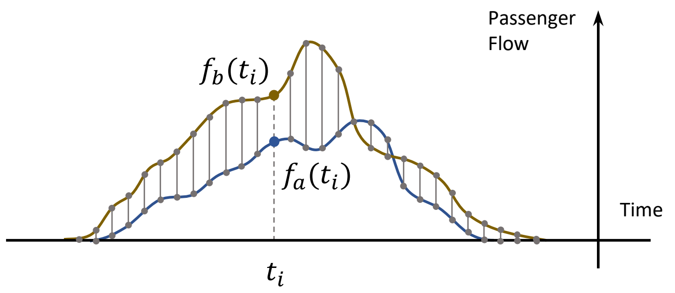


Figure 19. Computation of the similarity for two different passenger flow profiles a and b. Display of the generic values $f_a(t_i)$ and $f_b(t_i)$ at a given t_i.

Ultimately, we want to choose those flights whose passenger profile curves resemble more to the flight for which we are making the prediction. Therefore, ideally, we would choose the weights in such a way that both similarities are equal. However, given the large number of flights and comparisons that need to be carried out, the real objective will be to minimize the difference between similarities for the entire set of comparisons.

This is a very simple optimization problem, where there are only 7 parameters. We solved it stochastically using a Genetic Algorithm (GA).

For the optimization of the parameters, we focus only on weeks -1 to Ω. That is, we exclude the week for which we are making the prediction, the week with index 0, since for this we lack information on the output passenger curves.

First, we compute all the resulting input similarity matrices for day d, $\mathbf{V_d}(-1,-\omega)$, $\omega \in [-2,\Omega]$. In the same way we calculate the output similarity matrices (those that result from the comparison of the passenger flows, denoted by $\mathbf{F_d}(-1,-\omega)$, $\omega \in [-2,\Omega]$. Another important aspect to keep in mind is that the similarity based on the inputs features depends on the selection of weights, and therefore is variable throughout the optimization process, while the similarity based on outputs is fixed. (The convention adopted for the symbols is **V** for variable and **F** for fixed.)

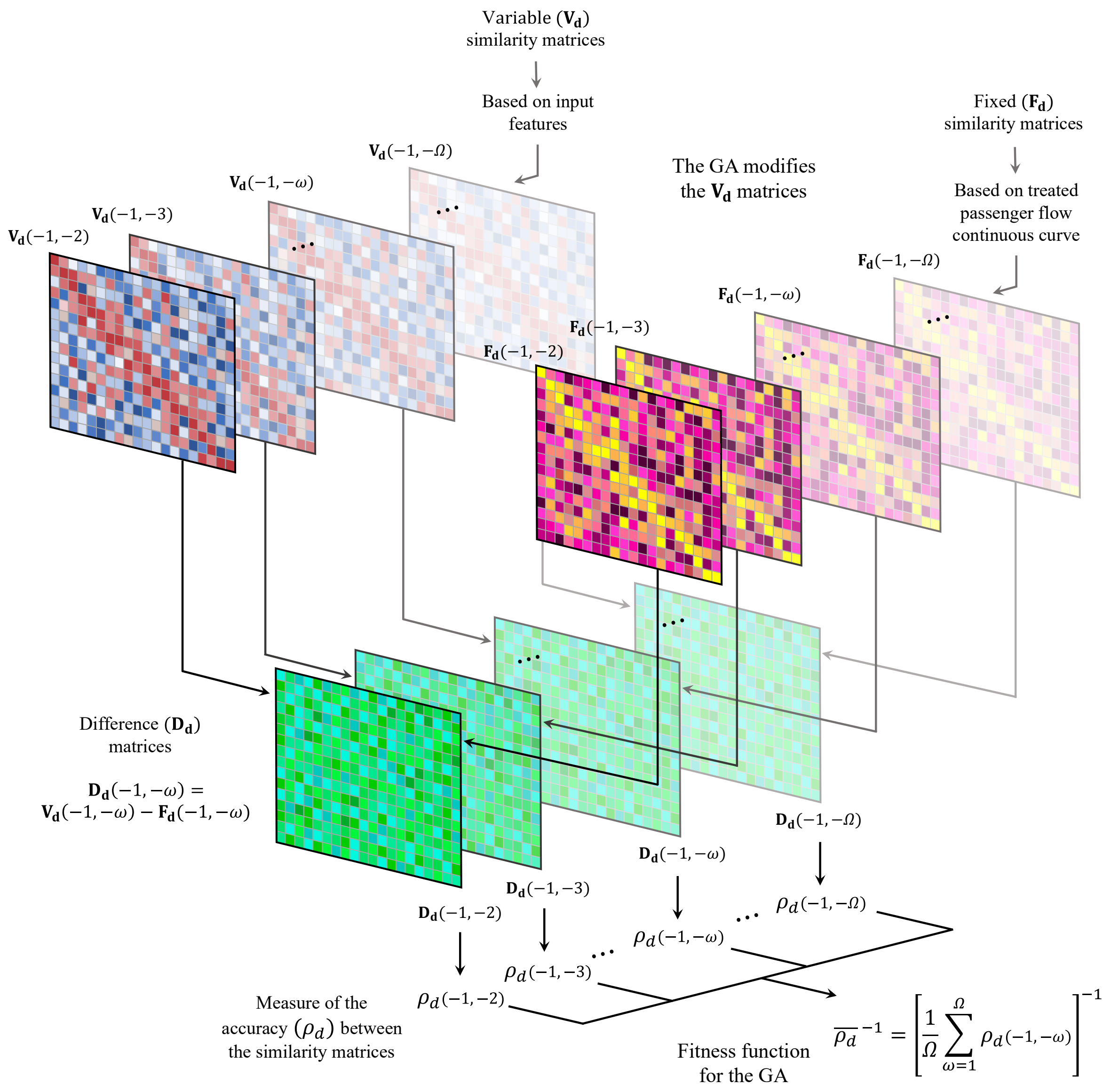
Subsequently, we obtain the difference matrices, $\mathbf{D_d}$**,** defined as

$$\mathbf{D_d}(-1,-\omega) = \mathbf{V_d}(-1,-\omega) - \mathbf{F_d}(-1,-\omega), \omega \in [-2,\Omega] \tag{10}$$

We then add the information of each entry in these matrices defining a figure of accuracy $\rho_d(-1,-\omega)$,

$$\rho_d(-1,-\omega) = \sum_{c=1}^{n_{cols}} \sum_{r=1}^{n_{rows}} \mathbf{D_d}(-1,-\omega)[r,c] \tag{11}$$

where n_{cols} and n_{rows} are the dimensions of $\mathbf{D_d}(-1,-\omega)$, which match with the length of $\boldsymbol{fl_d}(-1)$ and $\boldsymbol{fl_d}(-\omega)$ respectively. Finally, the fitness function of the problem is the inverse of the average of the accuracies $\frac{1}{\overline{\rho_d}}$. **Figure 20** shows the graphical representation of the calculation of the fitness value for the Genetic Algorithm.



Variable ($\mathbf{V_d}$) similarity matrices
Based on input features
$\mathbf{V_d}(-1,-\Omega)$
$\mathbf{V_d}(-1,-\omega)$
$\mathbf{V_d}(-1,-3)$
$\mathbf{V_d}(-1,-2)$
The GA modifies the $\mathbf{V_d}$ matrices
Fixed ($\mathbf{F_d}$) similarity matrices
Based on treated passenger flow continuous curve
$\mathbf{F_d}(-1,-\Omega)$
$\mathbf{F_d}(-1,-\omega)$
$\mathbf{F_d}(-1,-3)$
$\mathbf{F_d}(-1,-2)$
Difference ($\mathbf{D_d}$) matrices
$\mathbf{D_d}(-1,-\omega) = \mathbf{V_d}(-1,-\omega) - \mathbf{F_d}(-1,-\omega)$
$\mathbf{D_d}(-1,-\Omega)$
$\mathbf{D_d}(-1,-\omega)$
$\mathbf{D_d}(-1,-3)$
$\mathbf{D_d}(-1,-2)$
$\rho_d(-1,-\Omega)$
$\rho_d(-1,-\omega)$
$\rho_d(-1,-3)$
$\rho_d(-1,-2)$
Measure of the accuracy (ρ_d) between the similarity matrices
Fitness function for the GA
$\overline{\rho_d}^{-1} = \left[\frac{1}{\Omega}\sum_{\omega=1}^{\Omega}\rho_d(-1,-\omega)\right]^{-1}$

Figure 20. Graphical representation of the calculation of the fitness values. Input data-based similarity matrices denoted by $\mathbf{V_d}(-1,-\omega)$, output data-based similarity matrices denoted by $\mathbf{F_d}(-1,-\omega)$, and difference matrices denoted by $\mathbf{D_d}(-1,-\omega)$, where $\omega \in [-2,\Omega]$.

Each chromosome of the Genetic Algorithm has 7 genes that represent the weights for the calculation of the input data-based similarity matrices. Each chromosome generates a total of $\Omega - 1$ $\mathbf{V_d}$ matrices that are subsequently compared to the $\mathbf{F_d}$ matrices. Obviously, throughout the epochs of the evolution process, the $\mathbf{F_d}$ matrices should be stored and queried rather than re-generated at every iteration (for computational purposes). **Figure 21** shows how a generic population of chromosomes would look like, their associated matrices, and the resulting fitness values obtained after the calculation of the $\mathbf{D_d}$ matrices.

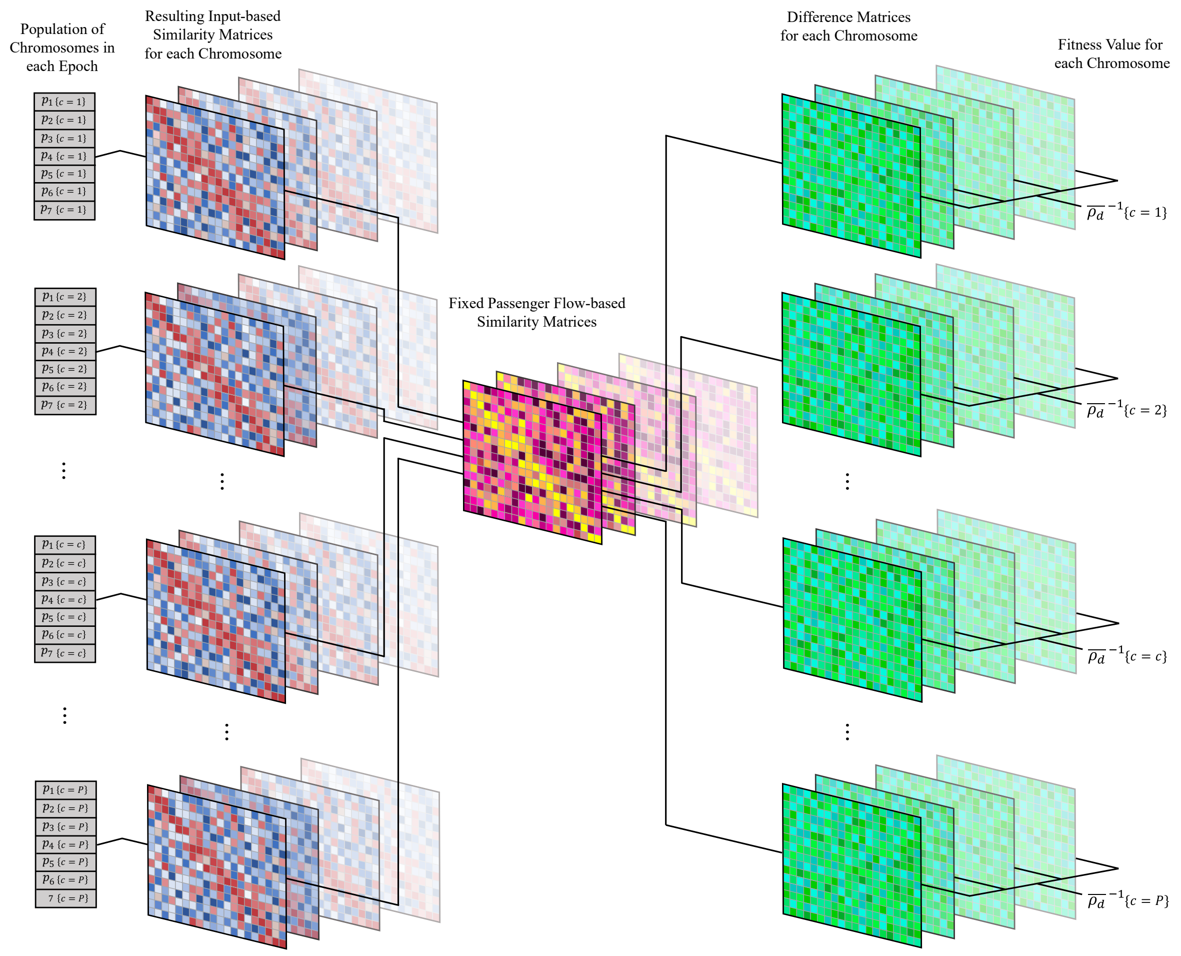
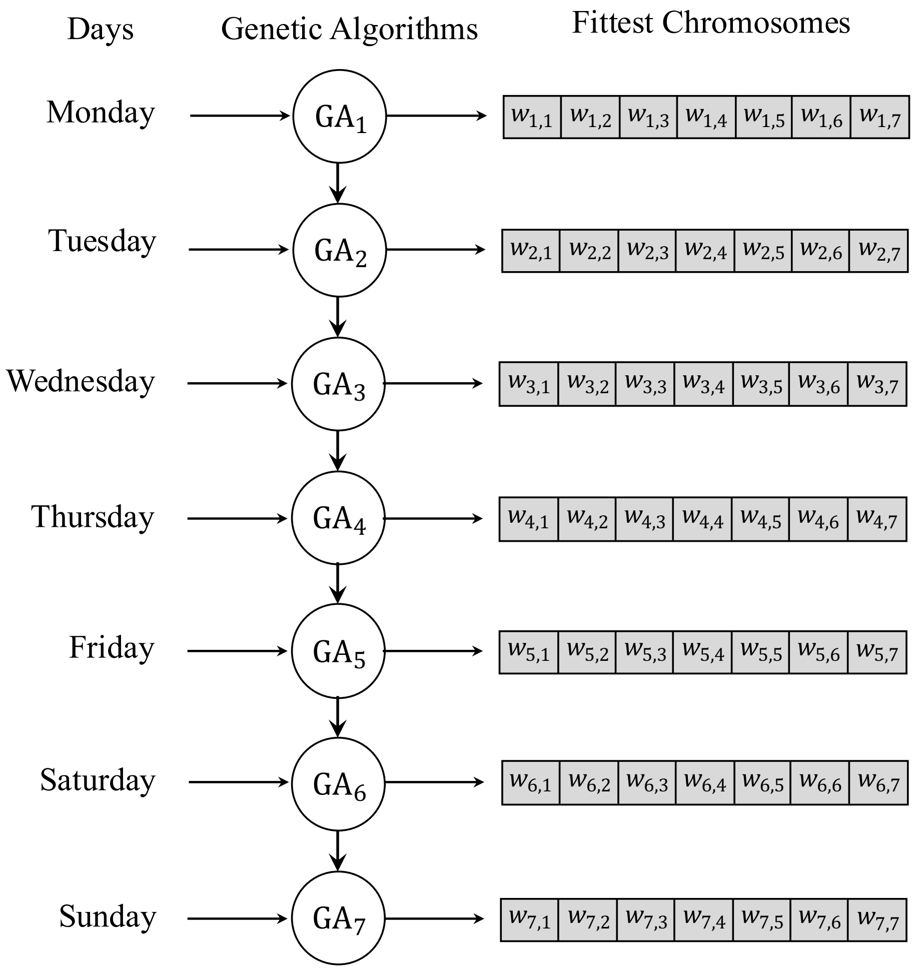

Population of Chromosomes in each Epoch
Resulting Input-based Similarity Matrices for each Chromosome
Fixed Passenger Flow-based Similarity Matrices
Difference Matrices for each Chromosome
Fitness Value for each Chromosome
$p_1\{c=1\}$ … $p_7\{c=1\}$
$p_1\{c=2\}$ … $p_7\{c=2\}$
$p_1\{c=c\}$ … $p_7\{c=c\}$
$p_1\{c=P\}$ … $p_7\{c=P\}$
$\overline{\rho_d}^{-1}\{c=1\}$
$\overline{\rho_d}^{-1}\{c=2\}$
$\overline{\rho_d}^{-1}\{c=c\}$
$\overline{\rho_d}^{-1}\{c=P\}$

Figure 21. Generic population of chromosomes (chromosome indexes identified by {c=index}). Each chromosome generates $\Omega-1$ matrices of $\mathbf{V_d}$ type and $\Omega-1$ matrices of $\mathbf{D_d}$ type, in the end the information of the $\mathbf{D_d}$ matrices is converted into a fitness value that is used in the optimization process of the Genetic Algorithm.

Up to this point, we have referred all the variables with respect to a specific day of the week, d. Therefore, it is necessary to carry out this same optimization process for each of the 7 calendar days of the week (Monday through Sunday). That is, we use 7 Genetic Algorithms to obtain different optimal parameters for each day (**Figure 22**). On the other hand, it is necessary to correct and update these 49 parameters (7 weights for each of the 7 days) from time to time. We have set the step for re-optimization at 1 week. Although the differences in the values of the weights between one week and another may not be very large, the error that we can accumulate due to a lack of update in one or two months may significantly jeopardize the overall prediction. Therefore, it is advisable to carry out as many updates as possible, which in this case is when we apply a weekly separation between updates. Finally, we do also incorporate a recirculation of some of the fittest chromosomes from the prior GA to the next to reduce the computational cost.



Days
Genetic Algorithms
Fittest Chromosomes
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
GA$_1$
GA$_2$
GA$_3$
GA$_4$
GA$_5$
GA$_6$
GA$_7$
$w_{1,1}$ $w_{1,2}$ $w_{1,3}$ $w_{1,4}$ $w_{1,5}$ $w_{1,6}$ $w_{1,7}$
$w_{2,1}$ $w_{2,2}$ $w_{2,3}$ $w_{2,4}$ $w_{2,5}$ $w_{2,6}$ $w_{2,7}$
$w_{3,1}$ $w_{3,2}$ $w_{3,3}$ $w_{3,4}$ $w_{3,5}$ $w_{3,6}$ $w_{3,7}$
$w_{4,1}$ $w_{4,2}$ $w_{4,3}$ $w_{4,4}$ $w_{4,5}$ $w_{4,6}$ $w_{4,7}$
$w_{5,1}$ $w_{5,2}$ $w_{5,3}$ $w_{5,4}$ $w_{5,5}$ $w_{5,6}$ $w_{5,7}$
$w_{6,1}$ $w_{6,2}$ $w_{6,3}$ $w_{6,4}$ $w_{6,5}$ $w_{6,6}$ $w_{6,7}$
$w_{7,1}$ $w_{7,2}$ $w_{7,3}$ $w_{7,4}$ $w_{7,5}$ $w_{7,6}$ $w_{7,7}$

Figure 22. The seven Genetic Algorithms for each of the days of the week, with the fittest chromosomes for each optimization process.

Scaling to account for Seasonality

We are using the information of past years to scale the curves of the current year and make fair comparisons between them. First, we will distinguish between two load factors:

Load factor $\xi_y(d(-\omega))$ for the day d (Monday, Tuesday, …, or Sunday) of a given week $-\omega$ (ω weeks before the current week for which we are making the prediction), for a given year y. We define this variable as the ratio between the total number of passengers that came into the airport that day of the year, and the total number of seats available aggregating the seats for all flights for that day of the year (**Figure 23** shows an example of a daily evolution of $\xi_y(d(-\omega))$ for a given airport),

$$\xi_y(d(-\omega)) = \frac{\text{Passengers for day } d(-\omega) \text{ in year } y}{\text{Seats for day } d(-\omega) \text{ in year } y}. \tag{12}$$

Load factor $LF(f)$ for the flight f, defined as the ratio between the number of passengers that boarded into flight f and the total number of seats available for that particular flight,

$$LF(f) = \frac{\text{Passengers boarded in } f}{\text{Seats available in } f}. \tag{13}$$

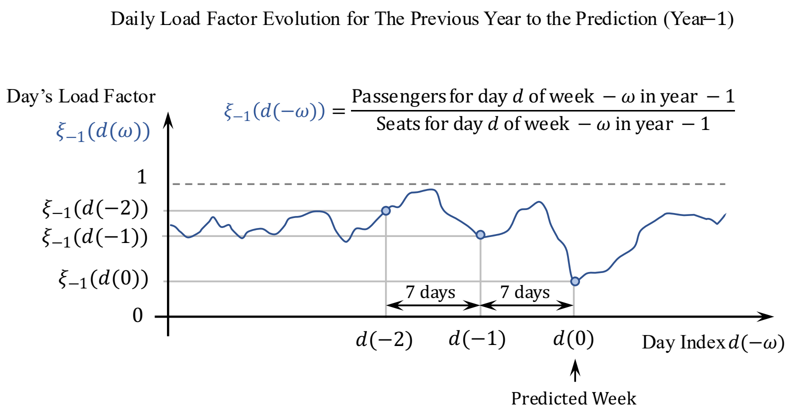


Figure 23. Evolution of the airport's load factor for each day (bounded between 0 and 1). In the horizontal axis we use the day index $d(-\omega)$ that represents the ω^{th} week before the 0^{th} week, being the 0^{th} week the week for which we are calculating the passenger flow prediction.

The load factor for the day can in fact be approximated as the average of the load factors of each individual flight that same day. If $\boldsymbol{fl}_{\boldsymbol{d},\boldsymbol{y}}(-\omega)$ represents the indices of the flights for that given day, and [f] identifies a specific entry in the vector, then

$$\xi_y(d(-\omega)) \approx \frac{1}{n_{fligths}} \sum_{f=1}^{n_{flights}} LF\big(\boldsymbol{fl}_{\boldsymbol{d},\boldsymbol{y}}(-\omega)[\mathrm{f}]\big), \quad \text{with} \;\; n_{flights} = \big|\boldsymbol{fl}_{\boldsymbol{d},\boldsymbol{y}}(-\omega)\big|. \tag{14}$$

Thus, when we are using the continuous passenger flow curves of a flight $\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)$ [f] for the prediction of $\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(0)$ [g] (for a given g) we can actually correct the load factor with the transformation $\beta(\cdot)$, considering the information from the previous year to the prediction ξ_{-1},

$$\beta\big[LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)\big] = LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[\mathrm{f}]\big)\frac{\xi_{-1}(d(0))}{\xi_{-1}(d(-\omega))}. \tag{15}$$

Nevertheless, this transformation may result in a value that is bigger than 1, and all the load factors should be bounded between 0 and 1. Thus, we have to apply the following correction,

$$LF'\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big) = \vartheta\left[\beta\big[LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)\big]\right] \tag{16}$$

where

$$\vartheta(x) = \begin{cases} 1 & \text{if } x > 1 \\ 0 & \text{if } x < 0 \end{cases} \tag{17}$$

The corrected number of passengers $P'(\cdot)$ for that flight can be calculated from the total number of seats $S(\cdot)$,

$$P'\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big) = S\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big) \cdot LF'\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big). \tag{18}$$

Finally, we scale the curve with the factor κ

$$\kappa = \frac{LF'\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)}{LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)} = \frac{\vartheta\left[\beta\big[LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)\big]\right]}{LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)} = \frac{\vartheta\left[LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[\mathrm{f}]\big) \cdot \frac{\xi_{-1}(d(0))}{\xi_{-1}(d(-\omega))}\right]}{LF\big(\boldsymbol{fl}_{\boldsymbol{d},\mathbf{0}}(-\omega)[f]\big)}. \tag{19}$$

For the scaling, we multiply each of the segments of the curve by the factor κ. Below we prove that this scaling will not affect on the fractions described in **Figure 16**.

The total sum of the segments before the scaling, A_{old} is defined as $\sum_{i=1}^{n_{segs}} b_{old_i}$ where b_{old_i} represents the value before scaling of i^{th} segment and n_{segs} is the total number of segments. After scaling, each segment is $b_{new_i} = \kappa \cdot b_{old_i}$, then A_{new} is $\sum_{i=1}^{n_{segs}} b_{new_i}$.

The fraction of a given portion of the segments from $i = i_0$ to i_f before the scaling is $frac_{old} = \frac{\sum_{i=i_0}^{i_f} b_{old_i}}{A_{old}}$ and the fraction for that same portion after scaling is

$$frac_{new} = \frac{\sum_{i=i_0}^{i_f} b_{new_i}}{A_{new}} = \frac{\sum_{i=i_0}^{i_f} b_{new_i}}{\sum_{i=1}^{n_{segs}} b_{new_i}} = \frac{\sum_{i=i_0}^{i_f} \kappa \cdot b_{old_i}}{\sum_{i=1}^{n_{segs}} \kappa \cdot b_{old_i}} == \frac{\sum_{i=i_0}^{i_f} b_{old_i}}{A_{old}} = frac_{old}. \quad (21)$$

To optimize the space utilized, we apply the scaling factors as we are populating the similarity matrices without storing the scaled curves. After choosing the λ first flights (most similar flights), we then perform again the scaling to those curves that will be used for the prediction.

Prediction
Once we have selected the past flights that will serve for the inference of each scheduled future flight, we perform the following steps:

1. Group the data of the selected flights, (as defined by the similarity function that was optimized by the GA). Note that we already accounted for seasonality when selecting the flights and rescaling them.
2. Query the weight of each of these past flights.
3. Perform the weighted average at each point of the continuous curves to obtain the final prediction of the scheduled flight.
4. Transform the continuous curve of predicted scans density to the histogram.

After the histogram predictions for all the scheduled flights have been performed, we then merge the histograms of all the scheduled flights for a given day, to obtain the final prediction of the entire day.

RESULTS

The results obtained from the execution of all the aforementioned phases have been collected in **Table 2**. For the calculation of the performance, we used the figures of merit defined in this paper.

Table 2. Average figures of merit for the months of October 2021 to February 2022 (FOM1 and FOM2). Outlier days (which show unusual lack of scans) were excluded for the calculation (some months show up to 2 days where very few scans or even no scans were recorded).

Month	FOM 1	FOM 2
October 2021	70.37 %	68.34 %
November 2021	70.08 %	73.22 %
December 2021	70.15 %	72.36 %
January 2022	70.64 %	73.59 %
February 2022	68.56 %	65.82 %

Next, we showcase some examples of the predictions made. **Figures 24**, **25**, **26**, and **27** represent the comparison between the predictions and the ground truth for 4 consecutive days of January of 2022, from the 16th to the 19th. On the other hand, **Figures 28** and **29** show the similarity matrices of 2 days, May 13th 2021, and May 20th 2021.

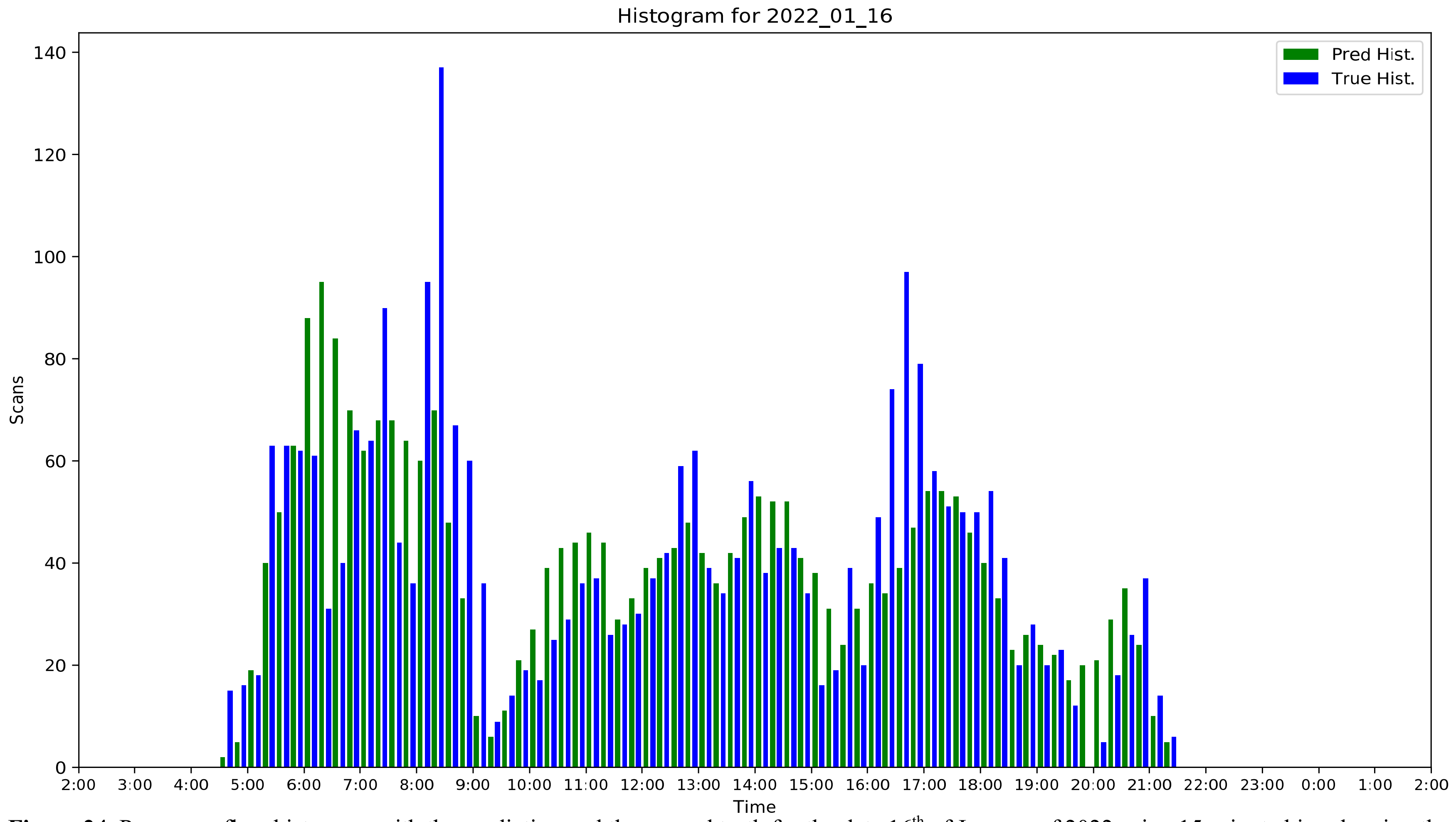


Figure 24. Passenger flow histogram with the prediction and the ground truth for the date 16th of January of 2022 using 15-minute bins showing the entire day (starting at 2:00am the 16th until 2:00am the 17th). With an accuracy of 79.65 % for FOM 1 and of 79.71 % for FOM 2.

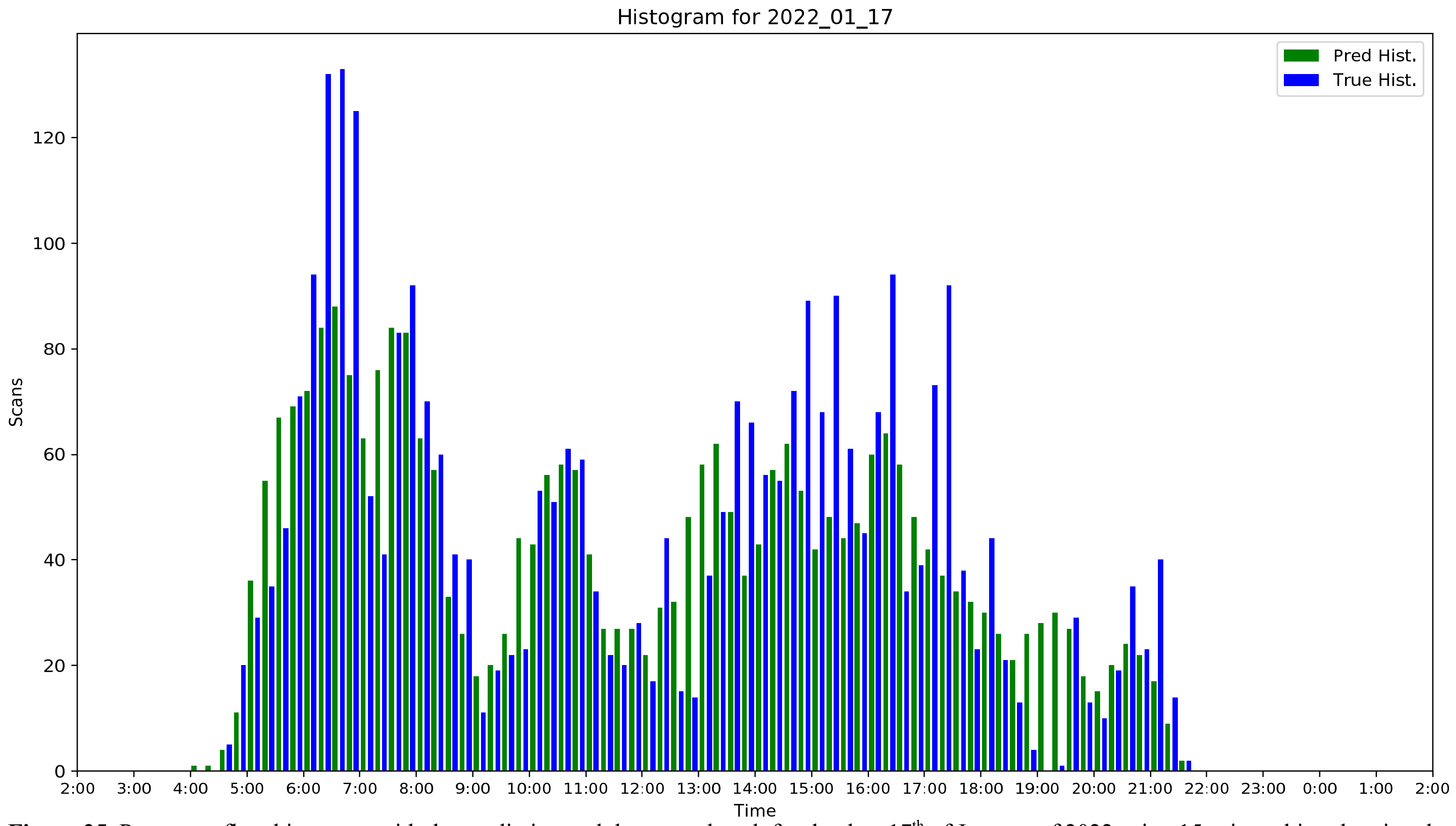


Figure 25. Passenger flow histogram with the prediction and the ground truth for the date 17th of January of 2022 using 15-minute bins showing the entire day (starting at 2:00am the 17th until 2:00am the 18th). With an accuracy of 77.46 % for FOM 1 and of 78.09 % for FOM 2.

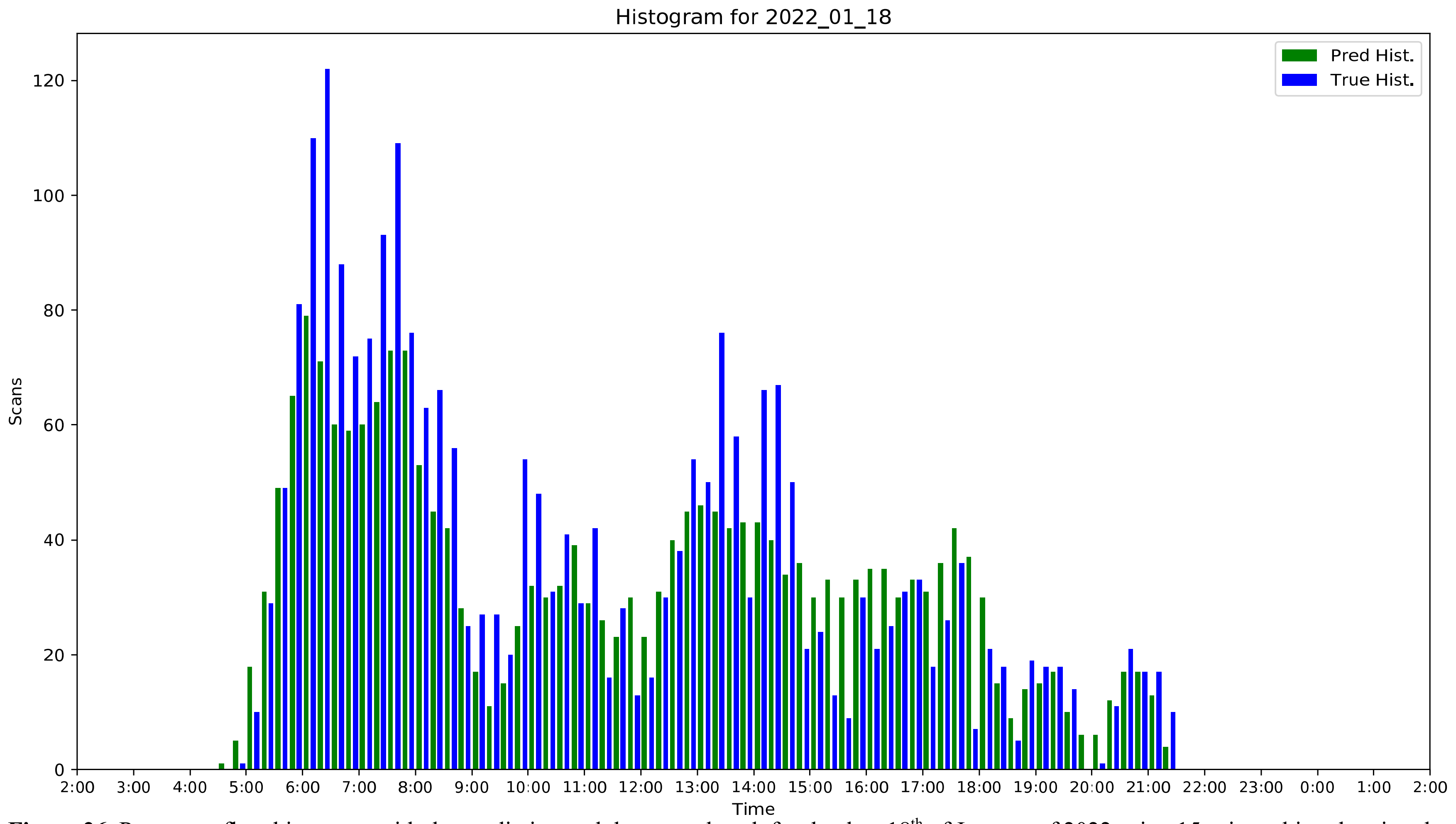


Figure 26. Passenger flow histogram with the prediction and the ground truth for the date 18th of January of 2022 using 15-minute bins showing the entire day (starting at 2:00am the 18th until 2:00am the 19th). With an accuracy of 77.73 % for FOM 1 and of 81.03 % for FOM 2.

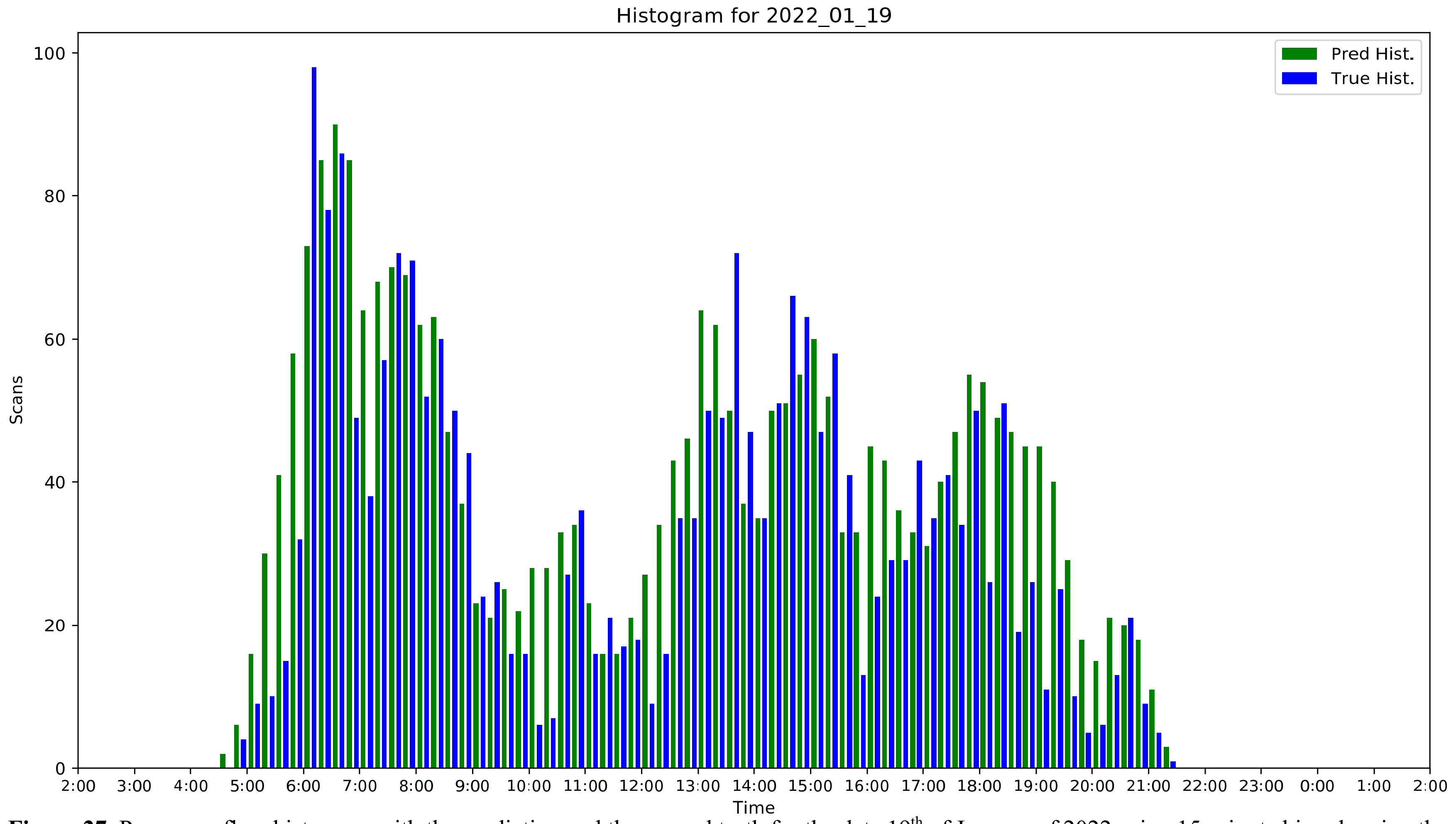


Figure 27. Passenger flow histogram with the prediction and the ground truth for the date 19th of January of 2022 using 15-minute bins showing the entire day (starting at 2:00am the 19th until 2:00am the 20th). With an accuracy of 71.08 % for FOM 1 and of 76.06 % for FOM 2.

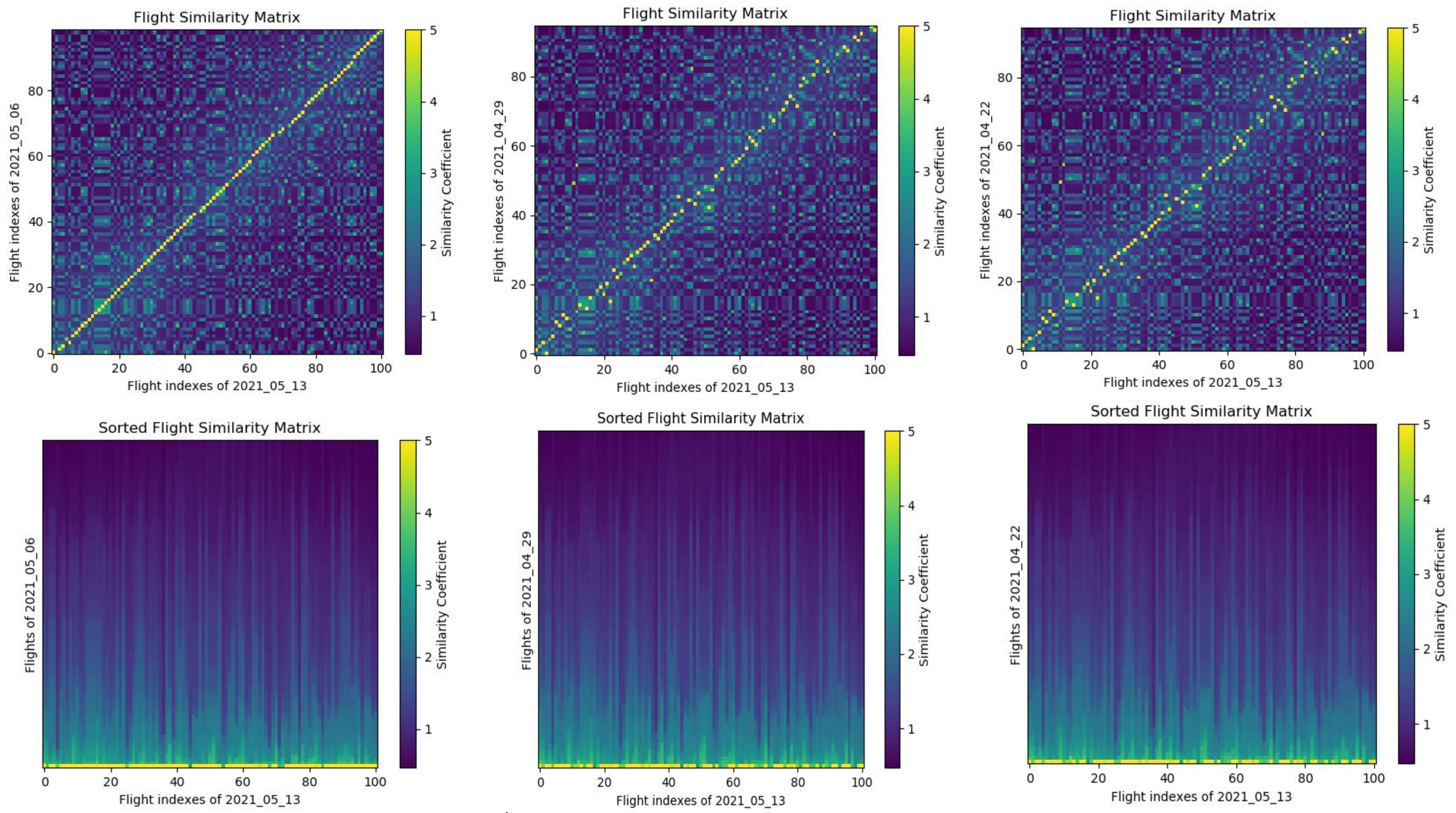


Figure 28. Flight similarity matrices using the 13th of May of 2021 as the basis for the comparison (horizontal reference indicates the flight index of a scheduled flight) and looking back 3 weeks (vertical reference indicates the flight index of a past week). Top matrices are the original similarity matrices, bottom matrices are the sorted similarity matrices.

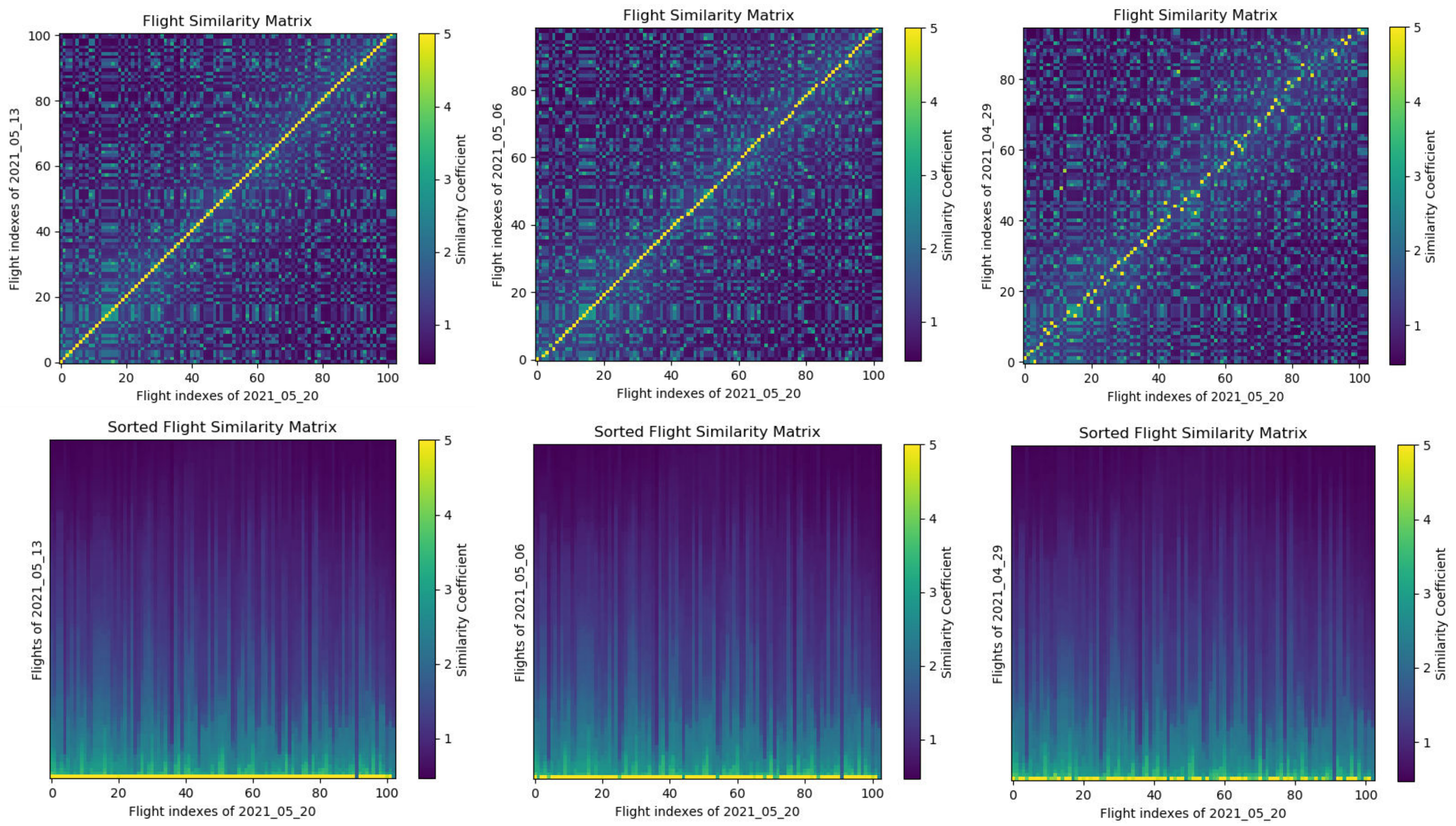


Figure 29. Flight similarity matrices using the 20th of May of 2021 as the basis for the comparison (horizontal reference indicates the flight index of a scheduled flight) and looking back 3 weeks (vertical reference indicates the flight index of a past week). Top matrices are the original similarity matrices, bottom matrices are the sorted similarity matrices.

INTERPRETATION OF THE RESULTS

The predictions and the observations follow similar trends. The algorithm is capable of correctly inferring the fluctuations that exist within the same day: the upward trend in the morning, the decrease in passengers during midday and the subsequent increase throughout the afternoon. In addition, the predictions made from one day to the next also adapt correctly to the different patterns that each day has.

In terms of consistency of results, we have seen that there is a high homogeneity throughout the year. Although it should be noted that there are isolated days where there is a low population of recorded passengers (for unknown external reasons). It is in these kinds of outliers that we see the biggest differences between the predictions and the observations. However, these cases of "corrupted" data can be resolved by improving the data acquisition process.

On the other hand, during the morning, when there is a peak of incoming passengers, is more difficult to make accurate predictions. This is due to the high waiting time that originates in the security checkpoint line as a direct consequence of a high volume of passengers. Thus, there is a bigger than usual delay between the true arrival time of the passenger to the airport complex and the scanning of the boarding pass, which then translates into more uncertainty for the prediction problem. Future improvements of the algorithm will include the integration of the line speed itself to mitigate this issue. This data is already available for the airport authorities at CVG from the Bluetooth technology installed to monitor the flow of the line.

In the similarity matrices we are comparing the similarity between the scheduled flights and the past flights of a recent day. For most of the scheduled flights there is one past flight that is very similar (substantially higher similarity than all the others). In fact, the closer this past day is to the actual day for which we are making the prediction, the bigger the one-to-one correspondence between flights. In other words, when we try to compare past flights that departed several weeks ago, it gets harder to identify similarities between the flights. This is primarily due to the accumulation of small changes in some of the flight schedules, which may not be substantial from one week to the other, but when we aggregate several weeks, they play an important role.

CONCLUSIONS

We have created an explainable algorithm capable of performing high-quality passenger flow predictions at the security checkpoint of an airport. To do so, we have used different data-processing blocks that transform the information in a transparent manner, without resorting to neural networks.

We created a collaboration between the University of Cincinnati and the Cincinnati / Northern Kentucky International Airport (CVG) to install and test the processing pipeline presented. The results obtained show that the developed algorithm is highly competitive. Based on the figures of merit defined we can state that the project was successful and that we significantly improved the predictive power of CVG systems for the prediction of the passenger flow. This not only has scientific implications that justify the use of this algorithm, but also offers the opportunity to create a commercial product that can be integrated in other US airports besides CVG. Indeed, future work should focus on the integration of this algorithm in more locations so that the corresponding authorities (TSA, airlines, airport managers, etc.) have access to an invaluable piece of information that concerns the security and the congestion at the airport.

Quoting Brian Cobb, Chief Innovation Officer of CVG airport: “At CVG, we’re redefining predictive analytics and our operational outcomes with Dr. Viaña’s Explainable AI. Aviation’s long attempt to rely on certainties of peak travel days or seasonality to schedule labor and systems has been obliterated post-pandemic where there are no absolutes to travel norms domestically or around the world. Explainable AI is the way forward to the quality and consistency we strive for in our industry and beyond.”

ACKNOWLEDGMENTS

The project that generated these results was supported by a grant from the “La Caixa” Banking Foundation (ID 100010434), whose code is LCF / BQ / AA19 / 11720045.
This paper arises from an official collaboration that we have established between the Universidad de Cincinnati, CVG Cincinnati/Northern Kentucky International Airport, and the Office of Innovation of CVG.
The project has been recognized with the Airport Cooperative Research Program Graduate Research Award, sponsored by the Federal Aviation Administration, administered by the Transportation Research Board and The National Academy of Sciences, and managed by the Virginia Space Grant Consortium.

AUTHOR CONTRIBUTIONS

The authors confirm contribution to the paper as follows: study conception and design: J. Viaña; data collection: S. Saunders, N. Marx, B. Cobb; supervision of the development: K. Cohen, S. Saunders; analysis and interpretation of results: J. Viaña; draft manuscript preparation: J. Viaña. All authors reviewed the results and approved the final version of the manuscript.

REFERENCES

1. Munasingha, K., and V. Adikariwattage. *Discrete Event Simulation Method to Model Passenger Processing at an International Airport.* Presented at 2020 Moratuwa Engineering Research Conference (MERCon), IEEE, 2020.

2. Çetek, F. A., and E. Aydoğan. *Air Traffic Flow Impact Analysis of RECAT for Istanbul New Airport using Discrete-Event Simulation*. Düzce Üniversitesi bilim ve teknoloji dergisi (Online), 2019. 7: 434–444.

3. Nikoue, H. et al. Pass*enger Flow Predictions at Sydney International Airport: A Data-Driven Queuing Approach*. arXiv preprint arXiv: 1508.04839, 2015.

4. Gu, X. et al. *Prediction of the Spatiotemporal Passenger Distribution of a Large Airport Terminal and its Impact on Energy Simulation.* Sustainable Cities and Society, vol. 78, 2022. DOI: 10.1016/j.scs.2021.103619.

5. Guo, X., Y. Grushka-Cockayne, and B. De Reyck. *Forecasting Airport Transfer Passenger Flow Using Real-Time Data and Machine Learning*. Manufacturing & Service Operations Management, 2021. DOI: 10.1287/msom.2021.0975.

6. Zhao, H., H. Zhou, and Q. Luo. *Research on Optimal Scheduling Method of Airport Rapid Transit System.* Presented at IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), IEEE, 2021.

7. Monmousseau, P. et al. *Predicting Passenger Flow at Charles De Gaulle Airport Security Checkpoints.* Presented at AIDA-AT 2020, 1st International Conference on Artificial Intelligence and Data Analytics for Air Transportation, IEEE, 2020.

8. Liu, L., and R-C. Chen. *A Novel Passenger Flow Prediction Model using Deep Learning Methods.* Transportation Research. Part C, Emerging Technologies, vol. 84, pp. 74–91, 2017.

9. Xie, G. et al. *Short-term Forecasting of Air Passenger by using Hybrid Seasonal Decomposition and Least Squares Support Vector Regression Approaches.* Journal of Air Transport Management, vol. 37, pp. 20–26, 2014.

10. Liu, X. et al. *Prediction of Passenger Flow at Sanya Airport Based on Combined Methods.* Data Science, Communications in Computer and Information Science, vol. 729, 2017. DOI: 10.1007/ 978-981-10-6385-5_61.

11. GDPR, European Commission, 2018.

12. Kuner, C. et al. *The EU General Data Protection Regulation (GDPR): A Commentary. First edition.* Oxford: Oxford University Press, 2020.

13. Kolasa, K. et al. F*uture of Data Analytics in the Era of the General Data Protection Regulation in Europe.* PharmacoEconomics, vol. 38, 2020. DOI: 1021–1029. DOI: 10.1007/s40273-020- 00927-1.

14. Gunning, D. *Explainable Artificial Intelligence (XAI).* DARPA. Tech. rep., 2017.

15. Gunning, D., and D. W. Aha. *DARPA's Explainable Artificial Intelligence (XAI) Program*. The AI magazine, vol. 40, pp. 44–58, 2019.

16. Bureau of Transportation Statistics, 2020. url: https :// www . bts . gov / browse - statistical - products-and-data/state-transportation-statistics/us-airline-traffic-airport.

17. Bureau of Transportation Statistics, 2021. url: https://www.bts.gov/newsroom/full-year-2020-and-december-2020-us-airline-traffic-data.

18. Li, Z. et al. P*assenger Flow Forecasting Research for Airport Terminal Based on SARIMA Time Series Model.* IOP Conference Series. Earth and Environmental Science, vol. 100, 2017. DOI: 10.1088/1755- 1315/100/1/012146.

19. Federal Aviation Administration, *The Economic Impact of Civil Aviation on the U.S. Economy.* Tech. rep., 2020. url: https:// www. faa. gov/ about/ plans_ reports/ media/ 2020_ jan_ economic_ impact_report.pdf.

20. Holguin, S. et al. *Why Sine Membership Functions*. Presented at Annual Conference of the North American Fuzzy Information Processing Society (NAFIPS 2022). Saint Mary's University, Halifax, NS, Canada, 2022.