Predicting Passenger Flow at Charles De Gaulle Airport Security Checkpoints

Philippe Monmousseau, Gabriel Jarry, Florian Bertosio, Daniel Delahaye, Marc Houalla

Predicting Passenger Flow at Charles De Gaulle Airport Security Checkpoints

Philippe Monmousseau*, Gabriel Jarry*, Florian Bertosio*†, Daniel Delahaye*, Marc Houalla†
* ENAC, Université de Toulouse, 7 Avenue Edouard Belin, 31400 Toulouse
Email: {jarry.gabriel, philippe.monmousseau, daniel.delahaye}@enac.fr
† Groupe ADP, Email: {florian.bertosio}@gmail.com

***Abstract*—Airport security checkpoints are critical areas in airport operations. Airports have to manage an important passenger flow at these checkpoints for security reason while maintaining service quality. The cost and quality of such an activity depend on the human resource management for these security operations. An appropriate human resource management can be obtained using an estimation of the passenger flow. This paper investigates the prediction at a strategic level of the passenger flows at Paris Charles De Gaulle airport security checkpoints using machine learning techniques such as Long Short-Term Memory neural networks. The derived models are compared to the current prediction model using three different mathematical metrics. In addition, operational metrics are also designed to further analyze the performance of the obtained models.**

***Keywords*—Airport operations, Machine Learning, LSTM networks, Airport security checkpoints, Passenger flow management, Strategic prediction**

I. Introduction

A. Motivation

Airport security checkpoints are key areas in airport operations. All passengers are checked at security checkpoint before entering the airside area. This continuous passenger flow implies an appropriate human resource management, which must satisfy two main objectives. A security checkpoint must be reliable in terms of security, while maintaining a predefined standard regarding passenger wait time. In addition, airports try to minimize their cost providing the best possible services.

At Charles De Gaulle airport, the human resources at security checkpoints are managed at two levels. The first level is a strategic level: Passenger flows at security checkpoints are predicted 20 days upstream for the following month in order to determine the appropriate number of agents required. The second level is a tactical level: In real time, the agents are distributed at the security checkpoints to provide the service. This paper investigates new learning methods such as neural networks in order to improve the prediction phase at the strategic level.

These learning methods are applied to the checkpoints within the zone of Charles De Gaulle airport corresponding to Air France's hub and named CDGE (cf. Figure 1). It contains eight security checkpoints, separated in three different categories, depending on the type of passengers going through:

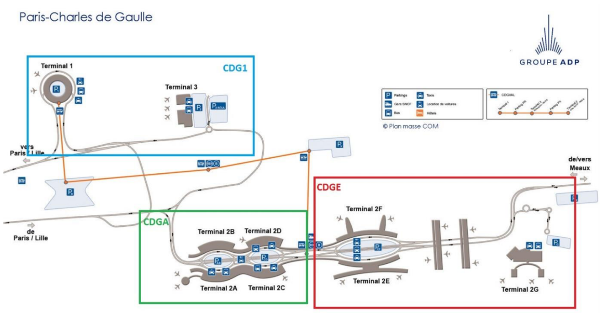


Figure 1: Overview map of Charles De Gaulle terminals

- Checkpoints handling only passengers with local flights: C2F-Centraux
- Checkpoints handling only connecting passengers: C2E-GalerieEF, C2E-Puits2E, C2E-PorteL-CNT
- Checkpoints handling passengers on both local and connecting flights: C2E-PorteK, C2E-PorteL, C2E-PorteM, C2G-Depart

Checkpoints C2E-GalerieEF and C2E-PorteL-CNT have the added particularity of linking two different terminals (E and F). C2E-Puits2E has the specificity of handling connecting passengers arriving to and leaving from Terminal E.

B. State of the art

Passenger flow prediction has been investigated for a long time in transportation areas. An exhaustive review was done by Liu et al [1]. Traffic flow prediction for public transportation was studied in [2], [3], and for air transportation in [4], [5] using various prediction methods. Time series models were developed by Kumar [6] based on Kalman filtering, while Williams and Hoel [7] and then Kumar and Vanajakshi [8] worked on auto-regressive models. In the machine learning field, regression models such as Support Vector Machines [2], [4] or Neural Networks [1], [3] were used to forecast passenger flow. So far, the models derived try to predict the passenger flow using only historical data of the flow. Nevertheless, an airport passenger flow is a complex process. Extra features could be added in order to enhance the model performance. Indeed, a model which includes information relative to the arriving and departing flights should outperform basic time

978-1-7281-5380-3/20/$31.00 ©2020 IEEE

series models. This motivates the use of machine learning models, that can fit multidimensional inputs.

Optimization of security checkpoints at a tactical level has also been thoroughly investigated. The efficiency of security checkpoint systems and organizations is discussed by Wilson et al. in [9] and by Leone and Liu in [10]. De Lange et al. suggested creating virtual queuing in order to decrease waiting time at peak periods [11]. However, to the best of the authors' knowledge, no study has been conducted around the airport security checkpoint strategic passenger flow prediction. Usually, each airport has its own process. Yet, the methodology presented in this paper is generic and could be applied everywhere. The only constraint is the availability of information regarding departing and arriving flight and their expected occupancy.

This paper is organized as follows: Section II describes the data considered, the features extracted from them and the learning models used. In Section III the different models are compared using both theoretical and operational performance measures. An in-depth analysis is performed in Section IV for two chosen checkpoints. Section V concludes this study and suggests some possible improvements and future steps.

II. Model creation

This section presents the machine learning models chosen for the following experiments as well as the data considered.

A. *Machine Learning and Long Short-Term Memory Neural Network*

A learning process consists in using data analysis methods and artificial intelligence to predict the behavior of a system. The aim is to define a model that will fit as best as possible the considered system. Machine learning algorithms define learning models h_θ, with parameters θ, that approximate the system function. The learning process is done upon a finite training set $\mathcal{D}$, and aims at minimizing the error over the training set by tuning the parameters θ of the learning model [12], [13], [14].

Various learning models exist in the literature and for various real-world applications, and in this paper the choice of a particular neural network named Long Short-Term Memory (LSTM) was made and compared to a Random Forest model. LSTM networks were designed as an enhancement of Recurrent Neural Networks (RNN) to perform better supervised learning task on time series data [15], [16], [17]. LSTM are capable of learning long-term dependencies, while simple RNN only learn short term dependencies. LSTM use a cell state that keeps information from the past, and three gates that update the cell state and compute the prediction. First, the forget gate enables updating the cell state in order to forget information that are no longer relevant based on the current input. Second, the input gate enables saving in the cell state relevant information from the current input. Finally, the output gate computes the prediction using the updated cell state and the current input. A simplified illustration of a LSTM structure is depicted in Figure 2.

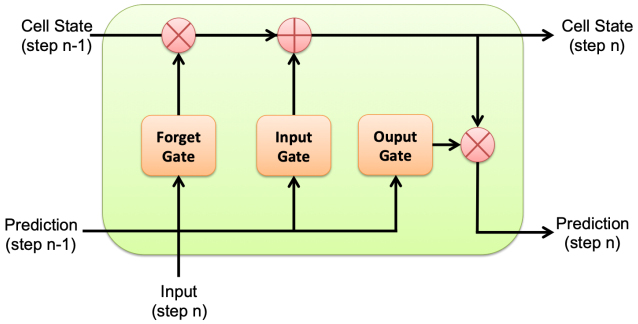


Figure 2: Simplified illustration of the structure of a LSTM cell

B. *From data to features*

In this study, the values to predict correspond to the real passenger count at each Safety Checkpoint per ten minute period. As explained in Section I, the original input dataset used by Charles De Gaulle operational experts is composed with information relative to the schedule and occupancy of arriving and departing flights aggregated per five minute periods.

The dataset starts on February, 1st 2017 and ends on March, 31st 2019. Data from both 2017 and 2018 were used for the training phase, and the data from 2019 for the validation phase. For each flight, there are three passenger count expectations corresponding to:

- the expected number of connecting passengers
- the expected number of local passengers
- the expected total number of passengers

These passenger counts are given by the airlines to the airport. In addition, there are various information such as the date, the status of the flight (departing or arriving flight), the airport terminal, the airline, the origin airport, the aircraft type, the departure geographic area, the flight range, and the check-in terminal.

Categorical features were represented using one-hot encoding. Additional features were extracted to complete the passenger count expectations. Passenger count expectations were aggregated per terminal, per status, and per terminal and status to create new features. Besides, features relative to the date were created: the month of the year, the day of the month, the day of the week, the hour of the day and the minute of the hour, and categorical variables for weekends, aeronautical weekends (including Fridays), holidays, and public holidays. Additional categories were created to capture whether a day is just before or after a public holiday or is the first or last day of a holiday.

This feature extraction yields a vector of 371 features for every five minutes of data. This vector sums-up the information over all the flights during the corresponding five minute period. The LSTM neural network was then fed with a time series corresponding to the input feature vector ranging from three hours before to five hours after the output 10 minutes time period. This time range was chosen based on two real-world considerations in order to encompass all the relevant flight and passenger information. On the one hand, airlines and airports recommend passengers on international flights to arrive about three hours before their flight departure time. On the other hand, the transfer time between two flights seldom exceeds five hours.

C. Network Architecture and Learning

This section describes the neural network architecture used in the experiments. The neural network is composed of two layers and a regression output layer. The first layer is a batch normalization. The second layer is a LSTM layer with 200 units and a sigmoïd activation function. The layer also contains a dropout to regularize the network. The output layer is a single neuron dense layer with a ReLU activation function. This architecture will be referred to as LSTM200. Figure 3 illustrates the network architecture.

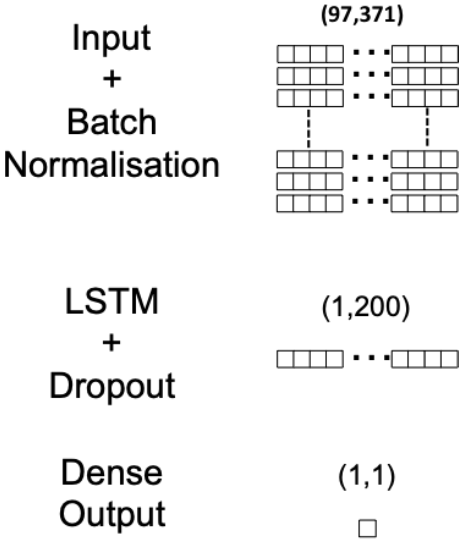


Figure 3: Description of the neural network LSTM200 architecture used in the experiments

The learning task was made using Adam optimizer [18] with a decay. The learning rate is 10^{-3} and the decay is 10^{-9}. Networks were trained during 10 epochs over the training set on a multi-GPU cluster. The cluster is composed of a dual ship Intel Xeon E5-2640 v4 - Deca-core (10 Core) 2,40GHz - Socket LGA 2011-v3 with 8 GPU GF GTX 1080 Ti 11 Go GDDR5X PCIe 3.0.

D. Penalized Loss

In practice, passenger count overestimation is costly. Therefore, a custom loss was designed. The loss aims to minimize overestimation by penalizing the positive part of the mean square error (MSE). As a reminder, the mean square error is the usual loss for regression problems. Let $\mathcal{D}$ be the training set, and h the learning model. The MSE of h over $\mathcal{D}$ is detailed in equation (1):

$$\mathbf{MSE}(h, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \cdot \sum_{(x,y)\in\mathcal{D}} (h(x) - y)^2 \qquad (1)$$

Let $E = h(x) - y$ be the error of a sample $(x, y) \in \mathcal{D}$. E_+ is the positive part of this error, and E_- the negative part. The α-Penalized MSE is defined in equation (2) with $\alpha \in \mathbb{R}$:

$$\alpha\text{-}\mathbf{PMSE}(h, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \cdot \sum_{(x,y)\in\mathcal{D}} (E_- + (1 + \alpha) \cdot E_+)^2 \qquad (2)$$

E. Model Summary

For this study, three models were used. The first model is a LSTM200 architecture trained with the MSE loss. The second model is a LSTM200 architecture trained with a 0.5-PMSE loss. The last model is a Random Forest model trained with MSE loss using the scikit-learn library [19]. The hyperparameters of the Random Forest models were set to 40 for the number of estimators, with a max depth of 10, and a minimum sample split of 2. The three models are summarized in Table I.

Table I: Summary of the three models used in the paper

Model Name	Model Type	Loss
LSTM (MSE)	LSTM200	MSE
LSTM (0.5-PMSE)	LSTM200	0.5-PMSE
RF	Random Forest	MSE

Additionally, in order to assess the effect of the hour of the day on the robustness of the chosen models, these models were trained twice: a first time with the hour of the day as a feature, and a second time without that feature.

III. Model comparison

A. Performance metrics

1) Theoretical metrics: In order to compare the performance of the different models, three different indicators were used: the R^2 score, the mean-absolute error (MAE) and a daily Pearson correlation score (DPC).

The R^2 score, also known as the coefficient of determination, is defined as the unity minus the ratio of the residual sum of squares over the total sum of squares:

$$\mathbf{R^2}(h, \mathcal{D}) = 1 - \frac{\sum_{(x,y)\in\mathcal{D}} (y - h(x))^2}{\sum_{(x,y)\in\mathcal{D}} (y - \bar{y})^2} \qquad (3)$$

where y is the value to be predicted, $\bar{y}$ its mean and $h(x)$ is the model prediction and $\mathcal{D}$ the dataset. It ranges from $-\infty$ to 1, 1 being a perfect prediction and 0 meaning that the prediction does as well as constantly predicting the mean value for each occurrence. In the case of a negative R^2, then the model has a worse prediction than if it were predicting the mean value for each occurrence and therefore yields no useful predictions.

Regarding the mean-absolute error, the smaller its value is, the more accurate the prediction is. It is calculated using the following formula:

$$\mathbf{MAE}(h, \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} |h(x) - y| \tag{4}$$

The daily Pearson correlation score is an average of the usual Pearson correlation score applied to non-overlapping subsets $\mathcal{D}_d$ of $\mathcal{D}$, with each subset $\mathcal{D}_d$ containing the data from an entire day d and $\mathcal{D} = \bigcup_{d\in D} \mathcal{D}_d$. It gives an indication of how well the curve of the predicted number of arriving passenger follows the actual curve of arriving passengers. The closer the score is to 1, the better the prediction is. It is calculated using the following equations:

$$r(h, \mathcal{D}_d) = \frac{\sum_{(x,y)\in\mathcal{D}_d} (h(x) - \bar{h}_d)(y - \bar{y}_d)}{\sqrt{\sum_{(x,y)\in\mathcal{D}_d} (h(x) - \bar{h}_d)^2} \sqrt{\sum_{(x,y)\in\mathcal{D}_d} (y - \bar{y}_d)^2}} \tag{5}$$

$$\mathbf{DPC}(h, \mathcal{D}) = \frac{1}{|D|} \sum_{\mathcal{D}_d} r(h, \mathcal{D}_d) \tag{6}$$

where $\bar{h}_d$ (resp. $\bar{y}_d$) is the average of $h(x)$ (resp. y) over $\mathcal{D}_d$.

2) Operational metrics: Airport management being a balance between minimizing costs and maximizing the service given to passengers, two additional metrics were introduced based on these operational considerations. These metrics are simplified versions of reality since the security agent providers do not share their calculation processes and the actual staffing of checkpoints is decided at a tactical level.

From a cost perspective, the key figure is the number of security agents necessary for a smooth operation. Agents being paid per hour, the cost metric considered is the total number of agent-hours induced by the predicted passenger arrivals. A smooth operation is here defined as a nominal passenger flow f_N, which has a unit of passengers per line per ten minutes. These flows are specific to each security checkpoint and are determined by the airport management. Airports also define a peak-time passenger flow f_P that security agents should be able to cope with when needed. From these nominal flows and the number of expected passengers p_t at time step t, it is then possible to compute the number of lines n_t required to achieve this flow: $n_t = \frac{p_t}{f_N}$. Assuming that each line is staffed by five security agents yields the number of agents required at each time step t. Each time steps being of ten minutes, it is then necessary to divide the resulting cost by six to obtain the agent-hour cost. The total cost metric C_T can be resumed by the following equation:

$$C_T = \frac{5}{6} \sum_t \frac{p_t}{f_N} \tag{7}$$

From a quality perspective, the key figure is the average waiting time at the security checkpoints. In order to estimate it at each time step, the following simplified queuing model is considered. At time step t, y_t passengers arrive at the checkpoint SC adding to the r_{t-1} passengers not processed during the previous time step. Under nominal conditions, $n_t \cdot f_N$ passengers are processed during a ten minute time step, where n_t is the number of lines estimated for the cost calculation. Peak-time conditions were defined here as time steps where the remaining number of passengers r_{t-1} was greater than the nominal flow f_N. Under peak-time conditions, the number of processed passengers becomes $\max(n_{t-1}, n_t) \cdot f_P$, i.e. the number of lines kept open stays the same if it was initially supposed to become smaller. If the prediction indicated that no lines should be open and that there are in fact passengers, then either the lines open in the previous time step are kept open if any, or one line is opened.

The processed number of passengers π_t at time step t can therefore be calculated as followed:

$$\pi_t = \begin{cases} \max(n_{t-1}, n_t) \cdot f_P & \text{if } r_{t-1} > f_N \text{ and } n_t > 0 \\ n_{t-1} \cdot f_N & \text{if } n_t = 0 \text{ and } n_{t-1} > 0 \\ f_N & \text{if } n_t = 0 \\ n_t \cdot f_N & \text{otherwise} \end{cases} \tag{8}$$

The average wait time τ_t during a time step t can be computed using the following equation:

$$\tau_t = \sum_{i=1}^{y_t + r_{t-1}} \frac{i}{\pi_t} = \frac{y_t + r_{t-1} + 1}{2\pi_t} \tag{9}$$

The overall quality metric Q_T is then calculated by taking the average of all τ_t.

The passenger flow model at a checkpoint is represented as an automata in Figure 4.

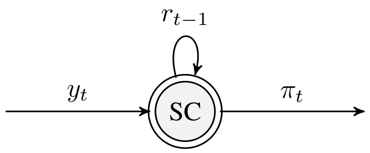


Figure 4: Model of the passenger flow at a security checkpoint

B. First results

All three models presented in Section II were trained using data from February 2017 to December 2018 and tested on the months of January to March 2019 using the performance metrics presented in Section III-A. These metrics were also applied to the current model in use at Charles De Gaulle airport for comparison. Based on operational observations, the output of the neural nets was forced to 0 when the hour of the day was between 00:00 and 04:00.

1) Hour of the day: Table II summarizes the performances of the three developed learning models based on two of the three mathematical metrics. This table enables a quick comparison of the use of the hour of the day as a feature. For the upcoming analysis, only one LSTM model and one Random Forest regressor were kept per checkpoint based on their MAE. The kept models have their performance cells highlighted in green, while the best of all models are also highlighted in bold. A first observation is that the influence of the hour of the day is not the same for Random Forests and for neural networks. For seven checkpoints over eight, using the hour of the day highly improves the Random Forest's

Table II: Comparison of the models using or not the hour in the training set. Green color cells correspond to the model kept in the following study. Bold cells correspond to the best models

	LSTM (MSE)				LSTM (0.5-PMSE)				Random Forest			
	With Hour		Without Hour		With Hour		Without Hour		With Hour		Without Hour	
	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE
C2G-Depart	0.736	4.02	0.732	4	**0.768**	**3.75**	0.754	3.85	0.721	4.27	0.712	4.68
C2E-PorteM	0.822	11.08	**0.887**	**9.02**	0.796	11.92	0.866	9.82	0.853	11.68	0.808	17.34
C2E-PorteL	**0.674**	**13.96**	0.641	14.58	0.578	15.8	0.636	14.67	0.684	14.21	0.558	19.58
C2F-Centraux	0.834	18.34	0.861	16.79	0.851	17.34	**0.86**	**16.62**	0.788	21.48	0.826	21.08
C2E-PorteL-CNT	0.436	16.54	**0.474**	**16.14**	0.426	16.6	0.425	16.49	0.471	16.36	0.475	17.62
C2E-GalerieEF	**0.667**	**15.32**	0.667	15.46	0.656	15.46	0.626	15.95	0.68	15.61	0.608	20.03
C2E-Puits2E	0.411	3.77	**0.37**	**3.76**	0.395	3.78	0.375	3.77	0.405	4.09	0.381	4.56
C2E-PorteK	0.688	18.37	0.758	15.63	0.662	15.67	**0.769**	**15.26**	0.726	16.73	0.647	23.53

performance. On the other side, six checkpoints over eight with LSTM (MSE and 0.5-PMSE) have better scores without the hour of the day.

2) Mathematical Performance Metrics: Figure 5 presents the performance of the current model and the kept models from Section III-B1 using the mathematical metrics introduced in Section III-A1. From a R^2 score perspective, both the LSTM and Random Forest models outperform the current prediction model with improvements ranging from 0.01 for C2E-PorteM to 0.3 for C2E-PorteL-CNT. Regarding the mean-absolute error performance, the LSTM nets outperform once more the current model while the Random Forest regressors have higher errors for two of the checkpoints (C2E-PorteM and C2E-Puits2E). The LSTM reduces the mean-absolute errors from 5.6% (C2E-Puits2E) to 18.9% (C2E-PorteM) compared to the current model: LSTM net have a mean-absolute error of less than seventeen passengers per ten minutes for all checkpoints while the current model has an error greater than seventeen passengers per ten minutes for half of the checkpoints. Finally, regarding the daily Pearson correlation score, LSTM model outperforms the current prediction model at every checkpoint, while the Random Forest regressor outperforms it for seven checkpoints out of eight.

3) Operational Performance Metrics: Using the simplified operational metrics introduced in Section III-A2, the difference in performance is less straightforward. Figure 6 shows the comparison of the cost metric (i.e. the number of agent-hour over the three months) per checkpoint as well as the comparison of the quality metric. Figure 6a presents the comparison of the total number of predicted passengers per checkpoints along with the actual number of passengers for comparison. A first observation is that the LSTM nets tend to underestimate the number of passengers regardless of the loss function considered, while the Random Forest regressors overestimate the number of passengers.

Since LSTM nets tend to underestimate the number of passengers more than the current model, it is also reflected from a cost perspective in Figure 6b: For seven of the checkpoints, the number of agent-hours required based on the neural nets is less than the number required based on the current model. For C2E-Puits2E, the number of required agent-hours is greater than the current model, a paradox illustrating the specificity of that terminal and further analyzed in Section IV.

4) Synthesis: Figure 7 shows the performance difference between the neural networks and the current prediction model, for all the metrics, and all the security checkpoints. All the metrics are normalized by the current prediction model value, except for the R^2 score, and the correlation score since they already have consistent magnitude and a norm lower than 1. The normalization enables comparison between security checkpoints. The difference is explained in percentage of improvement relative to the current model, except for the R^2 score and the correlation score where it is the improvement difference in percentage (norm lower than 1). In addition, the performance sign is selected such that a positive sign corresponds to a metric improvement. Finally, the performance difference is displayed with color from green when the improvement is greater than 20% to red when the best model deteriorates the performance more than 20%

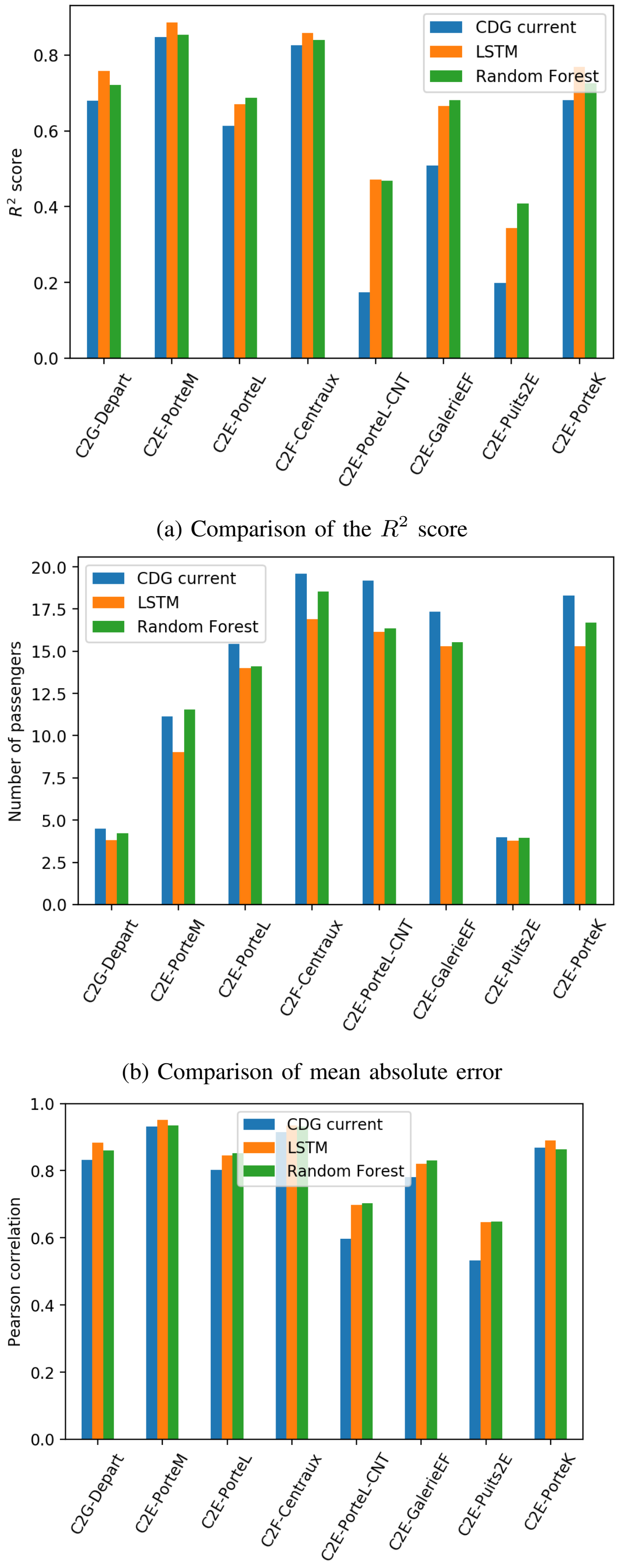
For three security checkpoints over eight (C2G-Depart, C2E-PorteM, C2F-Centraux), LSTM models outperforms the current prediction model for all the performance metrics. For four over eight (C2E-PorteL, C2E-PorteL-CNT, C2E-GalerieEF, C2E-PorteK) LSTM models outperform current model for all the metric excepted the waiting time metric, which is deteriorated more than 10% in half of the cases (C2E-GalerieEF, C2E-PorteK). Finally, at security checkpoint C2E-Puit2E, the performance metric is highly deteriorated for the agent number (-29%) and waiting time (-12.9%). This particular behavior will be explained in Section IV.

IV. Case Study

In this section, two security checkpoints were selected with respect to their performances for a further analysis. C2G-Depart was chosen to illustrate the good results of the LSTM model while C2E-Puits2E was chosen to better understand why the LSTM model does not outperform the current model from an operational perspective.

A. Daily analysis

A first step in understanding the differences in performance is to analyze the performances of the two models (LSTM and current) on a less aggregated level such as the different days of the week. Figure 8 shows the distribution of the daily Pearson correlation score per day of the week for the two chosen security checkpoints. It confirms the previous observation that

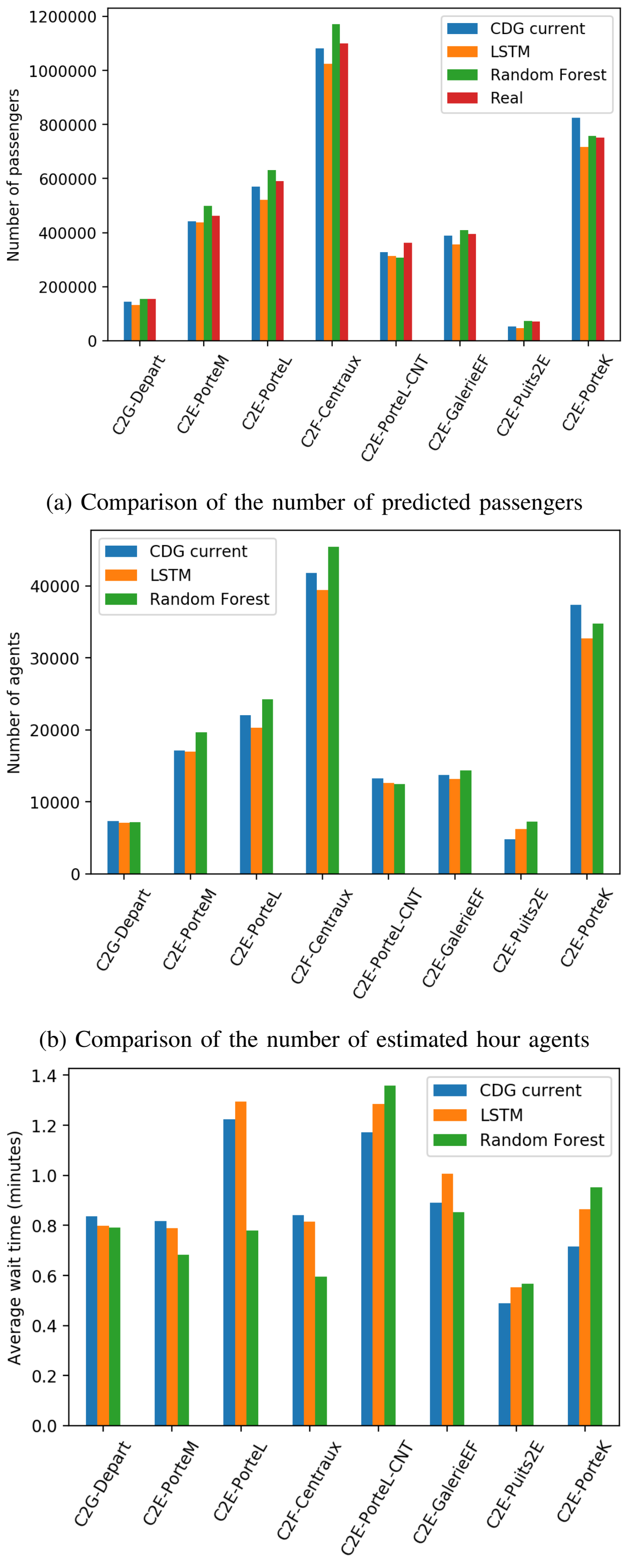


(a) Comparison of the R^2 score

(b) Comparison of mean absolute error

(c) Comparison of daily Pearson correlation score

Figure 5: Comparison per checkpoints of different mathematical metrics for the three considered models



(a) Comparison of the number of predicted passengers

(b) Comparison of the number of estimated hour agents

(c) Comparison of the number of the estimated average wait times

Figure 6: Comparison per checkpoints of different operational metrics for the three considered models

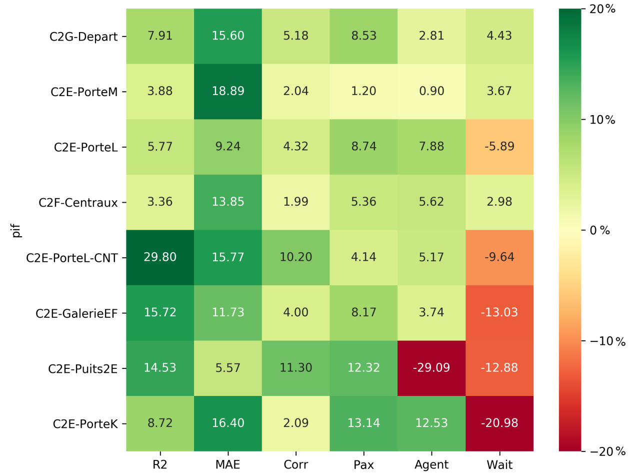


Figure 7: Heatmap visualization of the performance difference between the LSTM models and the current predictive model

the LSTM model is overall better than the current model with this metric, while adding some information on how this improvement is structured.

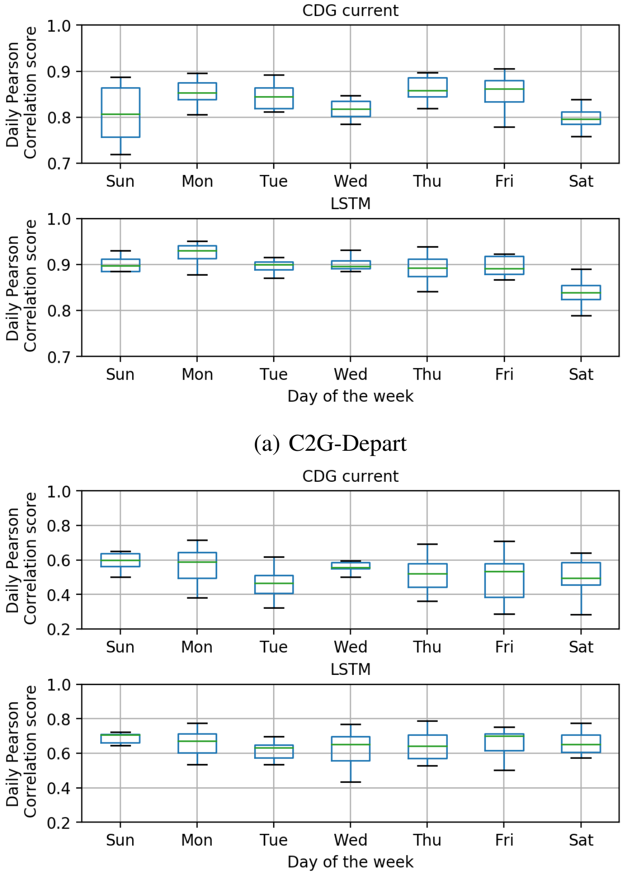
Regarding C2G-Depart, Figure 8a shows that both models are less precise on Saturdays compared to other days, though the LSTM model reduces the score variability on that day. An important improvement can be seen for Sundays: the current model has a lower score with a large variability, whereas the LSTM model reduces drastically that variability and improves the median score of 0.1.

Regarding C2E-Puits2E, Figure 8b shows that the LSTM model manages to reduce variability on most days, with an important reduction on Fridays. Wednesdays show an opposite behaviour: though the LSTM model does increase the median correlation score, it also triples the score variability.

B. Hourly analysis

A similar analysis can be conducted by aggregating the performance metrics per hour of the day. Figure 9 shows the hourly distribution of the error in predicting the number of arriving passengers for the current model and the LSTM model at the two chosen security checkpoints. It confirms the LSTM's tendency to underestimate the number of passengers: All medians are at or below zero for the LSTM while the current model tend to overestimate for five hours out of the sixteen considered hours for C2G-Depart. For C2E-Puits2E, both models have median errors at or below zero, however the LSTM model variations are shifted towards the negative with a smaller tendency to overestimation, which is indicated by smaller upper whiskers. This underestimation can be seen as a lower cost, since the predicted number of passengers determines the number of required agents.

Figure 10 shows the hourly distribution of the average wait time using the predictions from the current model and the LSTM model. Combining Figures 10 & 9 makes the impact of underestimation clearer on the quality of service. Underestimations in the number of passengers is associated with a higher median average wait time, which is then propagated in the following hours. For C2G-Depart, the underestimations at 3pm and 7pm on Figure 10a are clearly associated with a rise and propagation of the average wait time on Figure 9a. For C2E-Puits2E, it is most visible for the underestimation at 7am for both models.



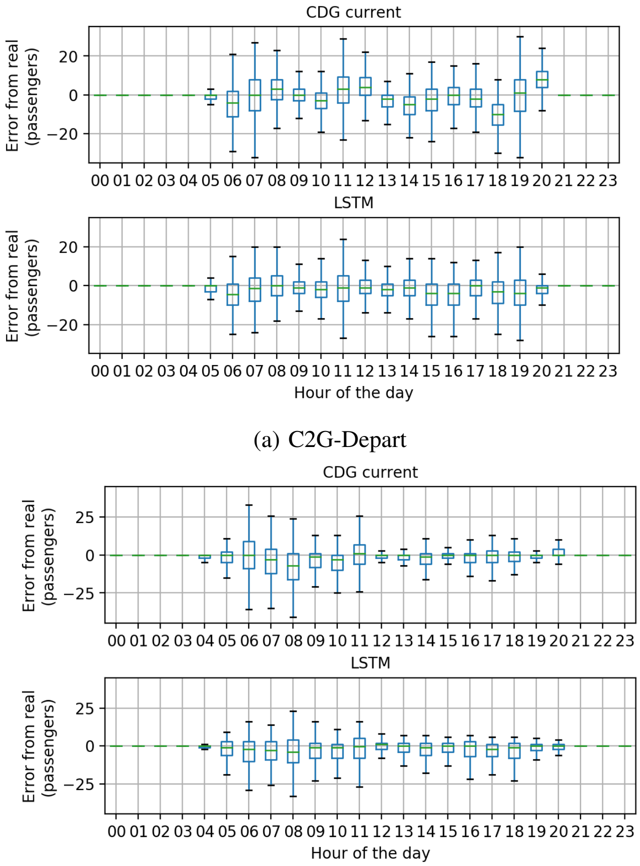
(a) C2G-Depart

(b) C2E-Puits2E

Figure 8: Daily correlation distribution per day of the week for C2G-Depart and C2E-Puits2E

This analysis could be used to further improve the derived models and the determination of the number of required agents. By highlighting hours of the days where the models are known to underestimate (resp. overestimate) the number of passengers, it should be possible to mitigate this underestimation (resp. overestimation) by adjusting the predicted value or by adapting accordingly the number of required agents for these specific periods.

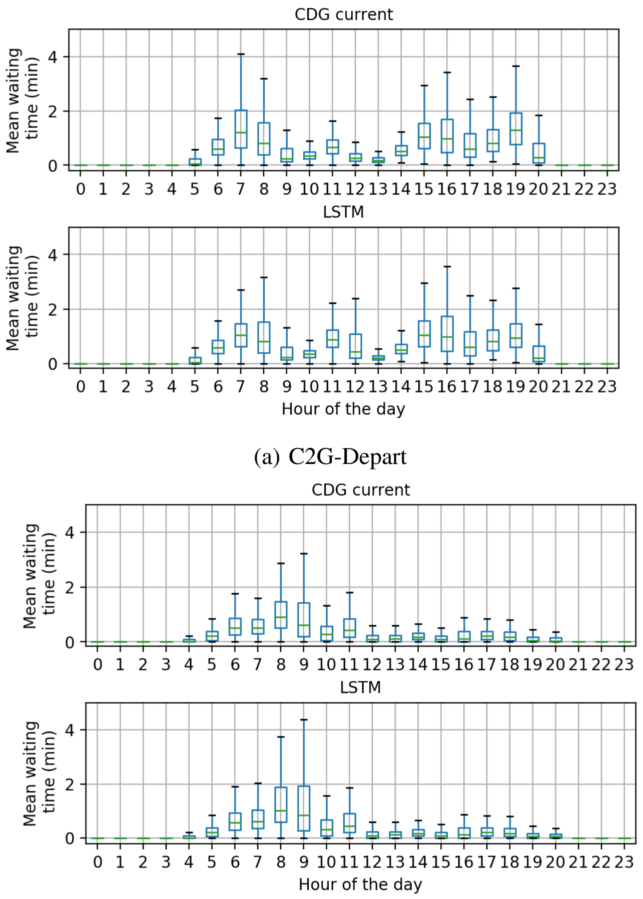
In order to better understand the differences in performance for these two checkpoints, the estimated number of passengers is plotted over a day (January 16th, 2019) in Figure 11 for C2E-Puits2E and in Figure 12b for C2G-Depart. Figure 11 highlights the difficulty of predicting the number of passengers for C2E-Puits2E: There are irregular yet continuous arrival



(a) C2G-Depart

(b) C2E-Puits 2E

Figure 9: Hourly passenger error boxplots comparison between the current model and the neural net trained with a mean squared error loss function at two different checkpoints



(a) C2G-Depart

(b) C2E-Puits 2E

Figure 10: Hourly average wait time boxplots comparison between the current model and the neural net trained with a mean squared error loss function at two different checkpoints

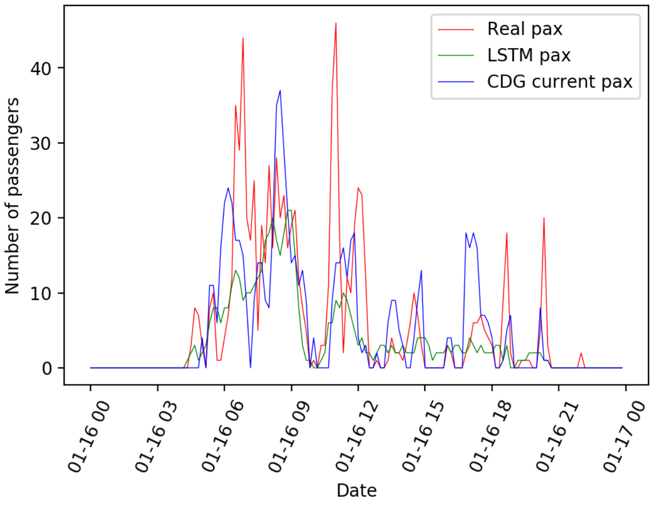
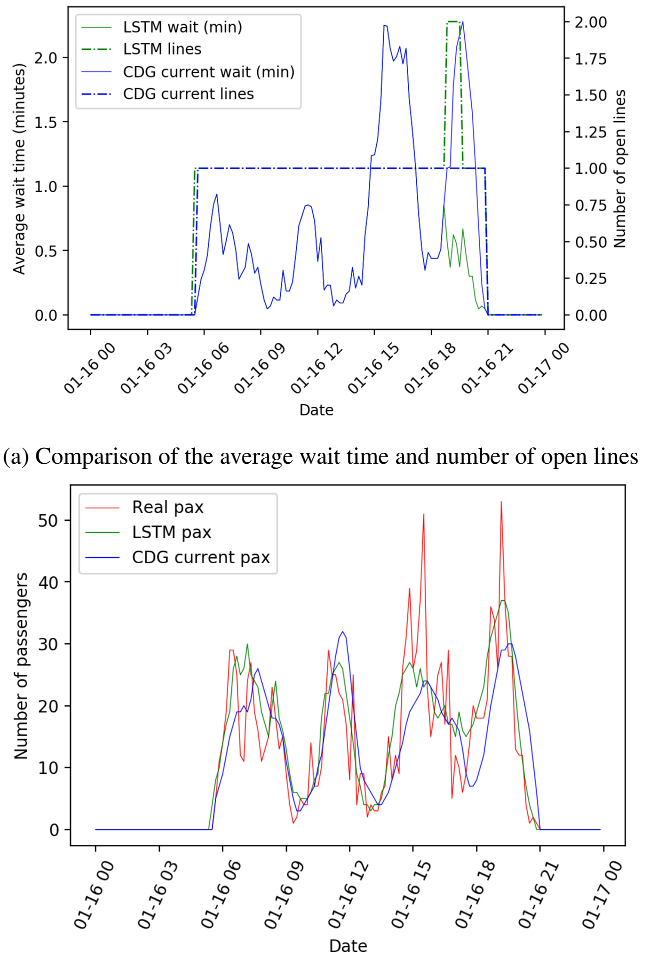


Figure 11: Hourly comparison of the predicted number of passengers between the current model and the neural net at C2E-Puits2E on January 16th 2019

spikes in the early morning (5am-9am) and then the rest of the day is composed of arrival spikes of varying amplitudes with periods with no passengers at all. From a prediction performance perspective, Figure 11 clearly illustrates the paradox of predicting less passengers while requiring more agents. The LSTM model underestimates more the passenger arrival spikes in the early morning than the current model, and estimates a low number of passengers for the rest of the day though never predicting zero arrivals. This means that agents are required all day long from the LSTM perspective, while the current model captures better the periods with no arrivals, enabling an economy of agents. A potential improvement of the LSTM model would be to hardcode the periods where operational expertise indicates that transfers within Terminal E are highly unlikely.

Regarding C2G-Depart, Figure 12b is a good day example to understand the better performance of the LSTM model compared to the current model. There are four daily spikes in passenger arrival with varying amplitude, and though both models capture the number of spikes, the LSTM model yields a better estimation of the amplitude of each spike as well as



(a) Comparison of the average wait time and number of open lines

(b) Comparison of the predicted number of passengers

Figure 12: Hourly comparison between the current model and the neural net trained with a mean squared error loss function at C2G-Depart on January 16th 2019

their initial slope increase. This higher accuracy has a direct impact on the estimated wait time, as shown in Figure 12a. The average wait time is identical for both model until the fourth spike, where the better estimation of the increase in passengers triggers the opening of a second line, which reduces the wait time by half compared to the current model.

V. Discussion & Conclusion

This paper investigated predicting passenger flow at Paris Charles De Gaulle airport security checkpoints using LSTM neural networks. The models performance was evaluated over several theoretical and operational metrics. The overall results are promising since LSTM models outperform the current model for every checkpoints using the theoretical metrics and for three checkpoints out of eight, LSTM models outperform the current prediction model using all the considered metrics. Though the considered operational metrics were simplified, these results illustrate that implementing a better and accurate strategic passenger flow prediction would surely reduce operational cost while maintaining predefined standard regarding passengers waiting time.

The methodology presented in this study can still be enhanced and tuned to be efficient and dedicated on specific cases. Future works should investigate a more elaborated queuing model or simulation. In addition, the models could be validated with real experimentation in the operations. Further works could be done on the neural network architecture and learning, or with expert to tune the models bringing relevant information to improve the prediction (hybrid models).

References

[1] L. Liu 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.

[2] Y. Sun, B. Leng, and W. Guan, “A novel wavelet-svm short-time passenger flow prediction in beijing subway system,” *Neurocomputing*, vol. 166, pp. 109–121, 2015.

[3] Y. Wei and M.-C. Chen, “Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks,” *Transportation Research Part C: Emerging Technologies*, vol. 21, no. 1, pp. 148–162, 2012.

[4] G. Xie, S. Wang, and K. K. Lai, “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.

[5] C.-I. Hsu and Y.-H. Wen, “Improved grey prediction models for the trans-pacific air passenger market,” *Transportation planning and Technology*, vol. 22, no. 2, pp. 87–107, 1998.

[6] S. V. Kumar, “Traffic flow prediction using kalman filtering technique,” *Procedia Engineering*, vol. 187, pp. 582–587, 2017.

[7] B. M. Williams and L. A. Hoel, “Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results,” *Journal of transportation engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[8] S. V. Kumar and L. Vanajakshi, “Short-term traffic flow prediction using seasonal arima model with limited input data,” *European Transport Research Review*, vol. 7, no. 3, p. 21, 2015.

[9] D. Wilson, E. K. Roe, and S. A. So, “Security checkpoint optimizer (sco): an application for simulating the operations of airport security checkpoints,” in *Proceedings of the 38th conference on Winter simulation*. Winter Simulation Conference, 2006, pp. 529–535.

[10] K. Leone and R. R. Liu, “Improving airport security screening checkpoint operations in the us via paced system design,” *Journal of Air Transport Management*, vol. 17, no. 2, pp. 62–67, 2011.

[11] R. De Lange, I. Samoilovich, and B. Van Der Rhee, “Virtual queuing at airport security lanes,” *European Journal of Operational Research*, vol. 225, no. 1, pp. 153–165, 2013.

[12] V. Vapnik, *Statistical learning theory*, ser. Adaptive and learning systems for signal processing, communications, and control. Wiley, 1998. [Online]. Available: https://books.google.fr/books?id=GowoAQAAMAAJ

[13] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, “The elements of statistical learning: data mining, inference and prediction,” *The Mathematical Intelligencer*, vol. 27, no. 2, pp. 83–85, 2005.

[14] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[15] Y. Bengio, P. Simard, P. Frasconi *et al.*, “Learning long-term dependencies with gradient descent is difficult,” *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[16] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” 1999.

[17] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, and D. Cournapeau, “Scikit-learn: Machine Learning in Python,” *Machine Learning in Python*, 2011.