Analysis of a Workload Model Learned from Past Sector Operations

David Gianazza
ENAC, F-31055 Toulouse, France
Univ. de Toulouse, IRIT/APO, F-31400 Toulouse, France
Email : {lastname}@recherche.enac.fr

***Abstract*—In this paper, we assess the performance of a workload model trained on a subset of sectors, focusing on how it generalizes on fresh sectors. The model of the air traffic controller workload is learned from historical data made of workload mesurements extracted from past sector operations and ATC complexity measurements computed from radar records and airspace data (sector geometry).**

The workload is assumed to be low when a given sector is collapsed with other sectors into a larger sector, normal when it is operated as is, and high when it is split into smaller sectors assigned to several working positions. This learning problem is modeled as a classification problem where the target variable is a workload category (low, normal, high) and the explanatory variables are the air traffic control (ATC) complexity metrics.

In previous work, we compared several classifiers on this problem. The models were trained on one week of traffic, and their generalization performance was assessed on another week of traffic, using the same sectors in both the training and test sets.

In the current work, we examine if models learned on a specific set of sectors can be performant on any other sector, or not. We also give a closer look at how the workload varies with the ATC complexity measures in our data, using bagplots of the data points for a few sector instances. The results allow us to better understand the strengths and limits of our data-driven model.

***Keywords*—Air traffic control, ATC complexity, workload, machine learning, airspace configuration**

I. Introduction

Predicting the workload of air traffic controllers is of crucial importance to the safety of the air traffic management (ATM) system at large. Overloads might lead to potentially dangerous situations where some conflicts might not be detected in time by the controllers.

Predicting the workload with good accuracy is also a question of efficiency. In day-to-day operations, the airspace is dynamically reconfigured according to the controller workload. Underloaded sectors are collapsed to form larger sectors, and overloaded sectors are split into several smaller sectors operated separately. When it is not possible to absorb the traffic simply by reconfiguring sectors, the traffic is delayed or rerouted so as to avoid the congested areas. This needs to be done well in advance, usually before the aircraft take off. Predicting with greater accuracy which ATC sectors shall be operated at what time and which of these sectors might get overloaded would improve the whole traffic regulation process. This requires a realistic and accurate workload model, which is the subject of this paper.

This problem has been addressed in several ways since the beginings of air traffic control (ATC). Depending on the context and purpose, one might count the movements on an airport, or the number of aircraft within the boundaries of an en-route sector, or the incoming flow of traffic over a time period. Such basic metrics – and the associated threshold values (capacities) – provide simple and straightforward answers to the question of deciding whether the controllers are experiencing a normal workload when handling given traffic, or if they are overloaded.

However, it has been aknowledged for a long time that simple metrics, such as *aircraft count*, do not adequately reflect the complexity of air traffic control. ATC complexity covers dynamic aspects relative to the traffic, static aspects relative to the sector geometry and route network, and aspects relative to the air traffic control procedures.

In this paper, we are interested in examining more closely the relationship between ATC complexity metrics and workload, using a gradient boosted tree model selected in our previous work [1]. This model is trained on historical data of aircraft trajectories and past sector operations. This data is made of complexity measures computed from radar tracks and sector data, and workload measures extracted from the status of the control sectors (collapsed, opened, or split into smaller sectors). We have shown in previous works that such a model provides correct predictions in more than 80% of the cases, when training the model on one week of traffic in all the french sectors, and when assessing the results on another week of traffic in approximately the same sectors.

Our aim here is to check if such a model can be trained on a subset of sectors, and still generalize well on fresh sectors. In addition, we want to verify if some intuitive notions on the relationship between complexity and workload – such as "if this complexity value increases, then the workload should be higher" – are actually true for our data. In other words, we would like to know if our model actually captures the relationship between the cognitive workload of the controller and the complexity metrics, or if our data-driven approach also captures some of the characteristics of the sectors and traffic patterns in our data.

The remainder of this paper is organized as follows: Section II gives some background on ATC complexity and air traffic controller workload, and states the objectives of the work presented in this paper. Section III is a short introduction to machine learning. Section IV describes the gradient tree boosting method used in this study. The data and experimental setup are described in section V. The results concerning the generalization performance on fresh sectors are given in section VI, and those concerning the relationship between complexity and workload are given in section VII. The paper concludes with a brief summary of our findings and the perspectives of future works, in section VIII.

II. Background and Objectives

A. ATC complexity and air traffic controller workload

In this paper, we are interested in the relationship between ATC complexity and workload. Both of these concepts are loosely defined in the literature ([2]), and before building models relating one to the other, we need to quantify them. Many ATC complexity indicators have been proposed in the literature [2], [3], [4], and this paper proposes nothing new in that matter. Quantifying the controller's workload has been done through different kinds of measures: physical activity ([5], [6]), physiological indicators ([7], [8], [9]), or subjective ratings ([10], [11]). Some of these indicators are difficult to interpret, and others are subject to biases (such as the recency effect denounced in [8], and the possibility of raters errors in the case of "over-the shoulder workload ratings" [12]). Collecting these data requires heavy experimental setups, often resulting in relatively small datasets and potential overfitting issues when trying to adjust a model on too few examples.

In order to avoid these drawbacks, we have proposed in previous work ([13], [14], [15]) to use historical records of the past sector operations to quantify the workload. These records are available in large quantity, for a large number of sectors. The information that can be extracted from the past sector operations is the following: we can assume that the workload was normal when the sector was operated, low when it was merged with other sectors to form a larger sector, and high when it was split into smaller sectors assigned to several working positions.

Several approaches have been tried to build models relating ATC complexity to workload. For example, taskload models ([16], [17]) compute the cumulative time required to execute control tasks. Linear regression models such as the popular dynamic density models ([18], [10]) approximate subjective workload ratings by a linear combination of a number of ATC complexity measures. Other works use a neural network instead of a linear model ([11]) to approximate subjective ratings.

In previous work, we also used neural networks, but our target variable was the workload measured from the past sector operations instead of subjective ratings. Considering an initial set of 27 complexity metrics found in the literature, we selected a subset of relevant metrics for the purpose of building a model that could be used to predict future airspace configurations ([13], [14], [15]). We showed that this concept was feasible and could be used to forecast airspace configurations that were much more realistic than the actual sector opening schedules made by the Flow Management Positions ([19], [20]). The concept was demonstrated on a mock-up HMI using static data ([21]). In [1], we compared the performances of several machine learning methods using the six selected ATC complexity metrics as input. The results showed that gradient boosted trees and neural networks performed better than basic classification methods such as linear discriminant analysis, quadratic discriminant analysis, or naive Bayes classifiers.

B. Objectives

In all our previous works, the ATC sectors from which were drawn our samples were approximately the same for both the training and test sets. The test data was simply taken in a different time interval than the training data, in order to assess the generalization performance of the trained models on fresh inputs (although in the same sectors).

In this paper, our first objective is to check if a model trained on a subset of sectors can perform well on another subset of sectors. Our concern is that the trained model might not perform well on elementary sectors, for which there are no occurences of "high" workload in the training data (as well as in the test data). A second objective is to examine more closely the relationship between the input variables (ATC complexity metrics) and the workload to determine if we actually learn a model of the cognitive workload of the air traffic controller, or if we also learn some characteristics of the observed data.

III. A Short Introduction to Machine Learning

This section is a brief introduction to machine learning. The reader may refer to [22], [23], [24] for a more thorough view of this active research field.

Learning from data with a computer can be done in different ways, through supervised learning, unsupervised learning, or reinforcement learning. In reinforcement learning, a sofware agent takes actions in a given environment so as to maximize a cumulative reward. In supervised or unsupervised learning, given some features x of an observed phenomenon, the objective is to learn a model from a set of examples $(\mathrm{x}_1, \ldots, \mathrm{x}_N)$. Unsupervised learning considers the explanatory variables x either to produce clusters of data, or to estimate the probability density of x, using the examples $(\mathrm{x}_1, \ldots, \mathrm{x}_N)$. In supervised learning, we assume a relationship $\mathrm{y} = f(\mathrm{x})$ between x and a target variable y, and we use examples of the outputs $(\mathrm{y}_1, \ldots, \mathrm{y}_N)$ associated with the inputs $(\mathrm{x}_1, \ldots, \mathrm{x}_N)$ to learn a model h approximating f.

In this paper, supervised learning techniques are used to predict the workload from ATC complexity indicators. The target variable y is here a workload category (low, normal, or high) and the input x is a vector of complexity indicators computed from the traffic or the sector geometry.

Such learning problems where the target is a categorical variable are usually referred to as classification problems, as opposed to regression problems where y is a floating-point value or a vector of floats.

Given a *loss* function ℓ such that $\ell(\mathrm{y}, \widehat{\mathrm{y}})$ is the cost of the error between the computed output $\widehat{\mathrm{y}} = h(\mathrm{x})$ and the observed data $\mathrm{y} = f(\mathrm{x})$, our objective is to choose h minimizing the following risk (i.e. the expected loss), where X and Y are the random variables from which are drawn x and y:

$$\mathcal{R}(h) = \mathbb{E}_{X,Y}\left[\ell(Y, h(X))\right] = \int_{\mathcal{X}\times\mathcal{Y}} \ell(\mathrm{y}, h(\mathrm{x}))\, \mathbb{P}_{X,Y}(d\,\mathrm{x}, d\,\mathrm{y}) \tag{1}$$

A. Learning from a finite dataset

In practice, the joint distribution of X and Y is not known and one can only approximate f using a set of examples $S = \{(\mathrm{x}_1, \mathrm{y}_1), \ldots, (\mathrm{x}_N, \mathrm{y}_N)\}$ of finite size.

The most straightforward idea is then to select the model h minimizing the following empirical risk:

$$\mathcal{R}_{emp}(h, S) = \frac{1}{|S|} \sum_{(\mathrm{x}_n, \mathrm{y}_n)\in S} \ell(\mathrm{y}_n, h(\mathrm{x}_n)) \tag{2}$$

Unfortunately, minimizing the empirical risk on S might not lead to the most desirable model. The selected model might fit the examples $\{(\mathrm{x}_1, \mathrm{y}_1), \ldots, (\mathrm{x}_N, \mathrm{y}_N)\}$ of S very well, while performing poorly on new instances of x[1].

Statistical models can be more or less "flexible" when fitting the data, depending on their analytical expression. For example, a linear model is much less likely to overfit the data than a polynomial of high degree. Selecting the best model among a collection of models of various "flexibilities" requires a *bias-variance tradeoff*. Simple models tend to have a high bias (i.e. they are far from truth) and a low variance (i.e. the response of the model is about the same, whatever the training set used to tune it). In contrast, complex models have low bias and high variance. A complex model tuned on too few examples tends to overfit these examples and to perform poorly on new inputs.

B. Model assessment and selection

There are several ways to control overfitting and to find a suitable bias-variance tradeoff. One can use an information theory criterion, such as AIC (Akaike's "An Information Criterion") or BIC (Schwartz's Bayesian Information Criterion). These asymptotic criteria add a penalty P depending on the model complexity to the empirical risk $\mathcal{R}_{emp}(h, S)$ defined in equation (2). The model having the lowest value of $\mathcal{R}_{emp}(h, S) + P$ is selected.

Another way to proceed is to assess empirically the generalization error. Let us denote $\mathcal{A}$ the algorithm used to learn a model from a dataset S. In holdout cross-validation, the initial dataset S is split into two sets: a training set S_T used to learn the models, and another set S_V used to assess the holdout validation error Err_{val} as defined by the equation below:

$$Err_{\text{val}}(\mathcal{A}, S_T, S_V) = R_{\text{emp}}(\mathcal{A}[S_T], S_V). \tag{3}$$

The model having the lowest holdout validation error is selected.

[1]For example, let us assume we fit a polynomial curve on 10 points. For this regression problem, a polynom of degree 9 will fit exactly the examples, but will give poor predictions at other points. For a classification problem, the same overfitting problem might occur when using a K-nearest-neighbours method with $K = 1$.

K-fold cross-validation is another popular empirical method, where the dataset S is partitioned into k folds $(S_i)_{1\leq i\leq k}$. Let us denote $S_{-i} = S\backslash S_i$. In this method, k separate predictors $\mathcal{A}[S_{-i}]$ are learned from the k training sets S_{-i}. The mean of the holdout validation errors is computed, giving us the cross-validation estimation below:

$$CV_k(\mathcal{A}, S) = \sum_{i=1}^{k} \frac{|S_i|}{|S|} Err_{\text{val}}(\mathcal{A}, S_{-i}, S_i). \tag{4}$$

When used for model selection, cross-validation can be performed successively on a collection of models. The model having the best cross-validation error is selected.

C. Hyperparameter tuning

In many methods, the bias-variance tradeoff is controlled through one or several parameters. For example, one can think of the number of hidden units in a neural network, or the weight decay hyperparameter. Hyperparameter values can be selected through cross-validation.

Let us denote λ the vector of hyperparameters of an algorithm $\mathcal{A}_\lambda$. In this paper, a 5-fold cross-validation has been used to tune hyperparameters, as described in algorithm 1.

Algorithm 1 Hyperparameter tuning for an algorithm $\mathcal{A}_\lambda$ and a set of examples T (training set).

```
function TUNEGRID($\mathcal{A}_\lambda$,$grid$)[$T$]
    $\lambda^* \leftarrow \underset{\lambda\in grid}{argmin}\ CV_{10}(\mathcal{A}_\lambda, T)$
    return $\mathcal{A}_{\lambda^*}[T]$
end function
```

IV. The Gradient Tree Boosting Method

In our experiments, we used the statistical software `R`, and more specifically the `Xgboost` library for gradient boosted trees.

The stochastic gradient boosting tree algorithm was introduced in [25], [26], [23]. It applies functional gradient descent to classification or regression trees ([27]).

The functional gradient descent is a *boosting* technique. The model h is iteratively improved. Denoting h_m the current model at iteration m, we consider the opposite gradient of the loss $g_i = -\frac{\partial\ell(\hat{y}, y_i)}{\partial\hat{y}}\left(h\left(x_i\right), y_i\right)$. A model g is then tuned to fit this opposite gradient, using a set of examples $(x_i, g_i)_{1\leq i\leq n}$. The model h is then updated as follows : $h_{m+1}(x) = h_m(x) + \rho g(x)$, where ρ is a constant minimizing the empirical risk. The next iteration repeats the same procedure for h_{m+1} until a maximum number of iterations is reached. In stochastic gradient boosting, the dataset is randomly resampled at each iteration.

In the Gradient Tree Boosting, the machine learning algorithm boosted by the functional gradient descent is a classification or regression tree algorithm. Before continuing our description of gradient boosted trees, let us say a few words on classification and regression trees (CART) which were introduced by Breiman in [27]. In this algorithm, a binary tree is used to represent a binary recursive partition of the input space. At each node, the input space is split in two regions according to a condition $x_j \leq s$. The J leaves of this tree describe a partition $(R_j)_{1\leq j\leq J}$ of the input space. Each region R_j is associated to a constant γ_j. In the case of regression, it will be a constant float value (usually the average value of the examples in region R_j). In classification trees, γ_j will be a class (the most represented class among the examples in R_j). When the tree is used to make a prediction on a new input x, the value γ_j is returned when x falls into R_j.

CARTs have some advantages. For example, they are insensitive to input monotonic transformations: Using x_j, $\log(x_j)$ or $\exp(x_j)$ leads to the same model. As a consequence, this algorithm is robust to outliers. It can easily handle categorical variables and missing values. However it is known to have a poor performance in prediction. This performance is greatly improved however when applying gradient boosting to CART.

In gradient boosted trees, the equation of the model update is the following, where ν is a *shrinkage* parameter:

$$h_m : \mathrm{x} \rightarrow h_{m-1}(\mathrm{x}) + \nu \sum_{R_j \in T_m} \gamma_{mj} \mathbb{1}_{R_j}(\mathrm{x}) \tag{5}$$

We can denote $\mathrm{GBM}_{(m,J,\nu)}$ the gradient boosted tree algorithm, where m is the number of boosting iterations, J is the number of leaves of the tree and ν is the shrinkage parameter. The final model obtained after boosting is a sum of regression or classification trees. J allows us to control the interaction between variables, as we have $J-1$ variables at most in each tree. ν is the learning rate. In [23] (chap. 10), it is recommended to take small values for the shrinkage parameter ($\nu < 0.1$) and small values for J as well ($4 \leq J \leq 8$). The hyperparameter grid used for this algorithm is presented in section V-E.

V. Data and experimental setup

A. Explanatory variables

In this study, ATC complexity indicators are used as inputs to our models. In previous works ([13], [28], [15]), we selected 6 basic complexity metrics among 27 metrics found in the literature. We used a principal component analysis to reduce the dimensionality of the inputs, and then selected the most relevant metrics related to the significant components. The 6 metrics that were found to be the most relevant *for the purpose of building airspace configuration prediction models* are the following:

- *vol*, the airspace volume of the considered ATC sector,
- *nb*, the number of aircraft within the sector boundaries at time t,
- *flow15*, the incoming traffic flow within the next 15 minutes,
- *flow60*, the incoming traffic flow within a 1 hour time horizon,
- *avg_vs*, the average absolute vertical speed of the aircraft within the sector,
- *inter_hori*, the number of speed vector intersections with an angle greater than 20 degrees.

These metrics are fairly simple and can be computed from radar tracks and static sector data (geometrical boundaries).

In the current paper, we have chosen to use the same metrics. They are standardized so as to obtain explanatory variables with mean 0 and standard deviation 1. These standardized variables are used as input vector x in our models.

B. Target variable

The target variable y we are trying to predict with our models is a workload category: *low*, *normal*, or *high*. In order to build our examples, we extracted this workload variable from historical airspace configuration data. In many cases, the workload in an ATC sector s at a past time t can be quantified by considering how it was operated at that time t. We simply make the following assumptions about the relationship between sector operation and workload:

- *Low workload* when sector s is collapsed with other sectors to form a larger sector operated on a working position,
- *Normal workload* when the sector s is operated as is,
- *High workload* when s is split into several smaller sectors operated on different working positions.

The other possible states – such as when a part of s is collapsed with one sector and another part is collapsed with another sector – are useless for quantifying the workload and are not used.

C. Dataset

The datasets used in this study are built from radar tracks and recorded sector operations from two weeks in October 2016 (13th to 26th), from the five french ATC control centers (Aix, Bordeaux, Brest, Paris, and Reims). For a given ATC sector, we sample the data so as to balance the occurences among the workload classes having non-zero occurences in the initial data. For example, for elementary sectors (or other sectors) for which there is no occurence in the high workload category[2], we sample an equal number of instances in the low and normal categories.

We then select all the sectors with non-zero occurences of the “normal” workload category (i.e., sectors that were opened at one moment or another). As a result, we obtain 50389 samples in the low workload category, 57539 in the normal workload category, and 21372 samples in the high workload category. This dataset is then completed by drawing samples from the sectors having no occurences of normal workload (i.e. sectors that were never opened). The resulting dataset comprises 57539, 57539, and 32800 samples in the low, normal, and high workload categories, respectively.

This procedure is different from the one adopted in our previous work [1], where our dataset was built so as to

[2]By definition, elementary sectors cannot be split, so there is no occurence of “high” workload according to our definition of workload based on the sector status (merged, collapsed, or split).

obtain an overall balance among the three classes[3], without considering the balance in each subset corresponding to each sector. The resulting dataset contains 147878 complexity and workload samples concerning 163 elementary sectors and 369 ATC sectors made of several elementary airspace sectors.

Note that the chosen data sampling procedure does not provide an exactly equal representation of the three classes in the resulting subset. One reason is that there exists no instance of the "high" workload class in the initial data for elementary sectors[4]. Even for ATC sectors made of several airspace sectors, the sample might be imbalanced, for example when the sector is opened for very large periods throughout the day (i.e. there might not be enough data of the low or high class from which to draw samples).

D. Performance evaluation and model selection

Our aim is to assess the performance of a model trained on a dataset extracted from past ATC sector operations, checking if this model generalizes well on new sectors that were not used to train the model. Consequently, our dataset must be split so that any ATC sector present in the subset used to train a model is not represented in the subset used to assess the model.

In addition, we would like examine how the trained model performs on ATC sectors made of only one elementary airspace sector. Considering these objectives, we use a nested cross-validation procedure, stratified so that the proportion of elementary sectors and collapsed sectors is about the same in all subsets.

The nested cross-validation consists in an outer 7-fold cross-validation for model performance assessment, embedding an inner 5-fold cross validation for hyperparameter selection. The dataset S is split in 7 subsets S_i, $1 \leq i \leq 7$. For each iteration i, a model is trained on $S_{-i} = S \backslash S_i$, and its performance is assessed on S_i. The training on S_{-i} follows a 5-fold cross-validation procedure (the inner cross-validation), in order to select the best hyperparameter values for the chosen machine learning method (model selection). This is done by splitting S_{-i} into 5 subsets $S_{-i,j}$, $1 \leq i \leq 5$. A grid of hyperparameter values is used to tune several models on $S_{-i,-j} = S_{-i} \backslash S_{-i,j}$, and the performance of these models are evaluated on $S_{-i,j}$. The hyperparameter λ^* providing the best performance on the subsets $S_{-i,j}$ is then selected, and a final model with hyperparameter λ^* is trained on S_{-i}. This is the model evaluated on S_i in the outer cross-validation.

This procedure is more computationally intensive than the one chosen in our previous work [1], where the outer cross-validation was a simple holdout validation, where the dataset was split into a training set and a test set only, instead of a 7-fold cross-validation. The new procedure has the advantage to provide some evaluation of the distribution of the performance results (considering the subsets S_i).

Also, in our previous paper, there were a large number of sectors overlapping in both the training and test sets. We were interested in evaluating how a model trained on one week of data would generalize on another week of data (the test set).

In the current paper, a different question is being adressed, motivating the change of procedure. We want to check if a model trained on a subset of sectors can generalize well on another subset of sectors, and if our model overfits the data for elementary sectors.

Knowing that there is no data with high workload available for the elementary sectors in our training sets, we might expect the model to overfit the training data and to generalize poorly on fresh examples, for elementary sectors. Hovever, it should generalize correctly on the other sectors.

E. Hyperparameter grids

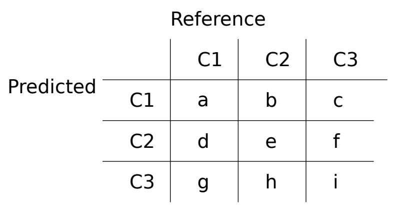
The hyperparameter selection of the inner cross-validation is performed on the training set, using function *TuneGrid* of algorithm 1 and the grids described in table I.

Method	Hyperparameter grid
GBM$_{(m,J,\nu)}$	$m = \{5000, 6000, 7000\}$ $J = \{2, 3, 4\}$ $\nu = \{$1e-4, 5e-4, 1e-3,1e-2,1e-1$\}$

Table I: Grid of hyperparameters used in our experiments.

F. Classification performance metrics

In this paper, the performance of a classifier is assessed through its accuracy, recall and precision. Accuracy is the total number of correct predictions made divided by the total number of predictions. Recall is the number of correct predictions made for one class, divided by the actual number of occurences in the considered class. Precision is the number of correct predictions made for one class divided by the total number of instances predicted to be in that class.



Predicted \ Reference	C1	C2	C3
C1	a	b	c
C2	d	e	f
C3	g	h	i

Figure 1: Illustration of a confusion matrix.

In other words, considering the example of a confusion matrix on Figure 1, accuracy, precision and recall are computed as follows:

$$Accuracy = \frac{a+e+i}{a+b+c+d+e+f+g+h+i}$$

$$Recall(C1) = \frac{a}{a+d+g}$$

[3]Note that the dataset description in section III-C of [1] is partly incorrect: although only sectors that were actually opened were initially selected, the dataset was completed using sectors that were not opened during the considered time period, in order to balance the number of instances among the three workload categories.

[4]By definition, elementary sectors cannot be split into several smaller sector, so we cannot measure high workloads for such sectors with the chosen target variable (the sector status).

$$Precision(C1) = \frac{a}{a+b+c}$$

VI. Results when generalizing on fresh sectors

In this section, we examine the performance of the GBM model when generalizing on fresh data taken from sectors that were not used when tuning the model. The model performance is assessed on several sub-populations of ATC sectors:

- All ATC sectors (elementary, or not),
- Elementary sectors, *i.e.* sectors that cannot be split into smaller sectors, and for which there is no occurence of the "high workload" class[5],
- Non-elementary sectors, *i.e.* ATC sectors made of several elementary airspace sectors, for which we detail the following results :
  - Overall performance on the non-elementary sectors
  - Model performance on non-elementary sectors for which there is no instance of the "high workload" class in the data.

A. Overall performance, all sectors included

Table II shows the performance of the GBM model, when including all sectors in the performance assessment. The first line shows the mean rates of correct classifications, and the standard deviations (within brackets). The overall rate of correct classification (2^{nd} line, 2^{nd} column) is the accuracy, and the class-specific rates (2^{nd} line, columns 3,4, and 5) are the recall. The precision of the model is given on the last line, where the overall precision is the average over the three classes.

The mean values and standard deviations are computed from the 7 folds of the cross-validation, considering only the validation subsets S_i (not used to train the model).

	Overall	Low	Normal	High
Correct classif.	0.759 (0.03)	0.757 (0.046)	0.736 (0.074)	0.808 (0.100)
Precision	0.771 (0.034)	0.817 (0.065)	0.680 (0.042)	0.815 (0.112)

Table II: Model performance averaged over 7 folds, for all ATC sectors, using the GBM method.

B. Results for elementary sectors

Table III shows the correct classification rates (overall accuracy, and class-specific recall), as well as the precision of the GBM model, for the control sectors made of only one elementary airspace sector. Such sectors cannot be split so as to alleviate the workload. As a consequence, there is no data concerning the "high" workload for these sectors.

For these results, the models trained in each fold of the cross-validation are exactly the same as in the previous subsection. They are trained on the same training subsets as before. However, the model performance is here evaluated considering only the elementary sectors in the validation subsets.

[5]With the chosen target variable, high workload can be observed only when the sector is split into several smaller sectors.

	Overall	Low	Normal	High
Correct classif.	0.82 (0.092)	0.904 (0.062)	0.507 (0.197)	NaN (NA)
Precision	0.519 (NA)	0.856 (0.112)	0.539 (0.271)	0 (NA)

Table III: Model performance averaged over 7 folds, for elementary airspace sectors only, using the GBM method.

C. Results for ATC sectors made of several airspace sectors

Table IV shows the results (correct classification rates and precision) obtained for the control sectors made of several elementary airspace sectors.

	Overall	Low	Normal	High
Correct classif.	0.757 (0.029)	0.739 (0.041)	0.748 (0.060)	0.808 (0.100)
Precision	0.769 (0.031)	0.814 (0.068)	0.679 (0.047)	0.815 (0.112)

Table IV: Model performance averaged over 7 folds, for ATC sectors made of several elementary airspace sectors, using the GBM method.

Table V details the results for the non-elementary sectors for which there is no occurence of the "high workload" class in the data, similar in that respect to the elementary sectors.

	Overall	Low	Normal	High
Correct classif.	0.738 (0.033)	0.713 (0.058)	0.760 (0.089)	NaN (NA)
Precision	0.504 (0.020)	0.784 (0.087)	0.729 (0.056)	0.000 (0.000)

Table V: Model performance averaged over 7 folds, for non-elementary sectors having no data instances in the "high workload" category.

When comparing the results in tables IV, V and III, we see that the model performance drops for the elementary sectors, with a recall barely above $50\ \%$ for the "normal workload" class, but that it remains good for non-elementary sectors in general, and even those with no occurence of high workload. This tends to show that our model overfits the data taken from elementary sectors specifically, and generalizes poorly on these sectors but not on the others. One reason might be that there are no other sectors in our data that would be similar to the elementary sectors in size and characteristic, and that would be well-balanced among the three classes. By contrast, we can find a lot of data samples from well-balanced (non-elementary) sectors having sizes and characteristics similar to the non-elementary sectors having no high workload occurence. This might explain why our model still generalizes correctly on these sectors.

This is a good result, as it means we can still use our workload model to find optimal combinations of sectors, although we should replace or amend our model when assessing the workload in elementary sectors.

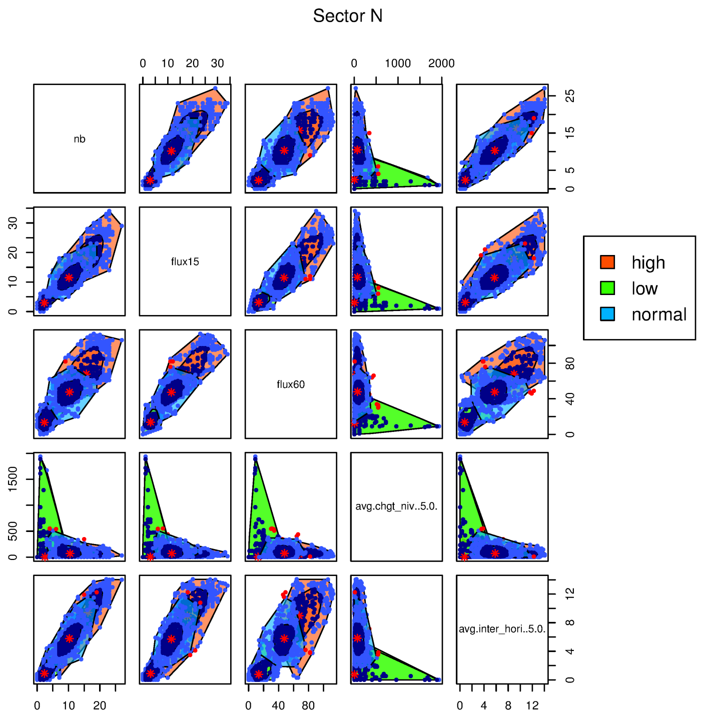


Figure 2: Bagplots for the sector N (Brest ATCC).

VII. Relationship between the input metrics and the workload classes

Figure 2 shows bagplots for sector N, for all pairs of explanatory variables and for the three classes (low, normal, or high workload). Bagplots are bivariate generalizations of the wellknown unidimensional boxplots. The centroïd corresponds to the median of the data. The inner bag is a convex hull containing 50% of the data points. The outer bag (called the loop) is obtained by expanding the inner bag by a factor 3. All points outside the loop are flagged as outliers.

Let us consider the graph on the second line and first column of Figure 2, representing the bagplots for the (nb, $flow15$) pair of explanatory variables. We see that both nb the number of aircraft (on the x-axis) and $flow15$ the flow of incoming traffic within the next 15 minutes (y-axis) are globally higher for data points in the "high" workload class than in the "normal" workload class. Similarly, the values of these explanatory values are higher in the "normal" class than in the "low" class. The centroids of the three workload classes are distributed along the diagonal axis of the graph, with the lowest workload closest to zero. The same observations can be made for all pairs of explanatory variables, looking at the other graphs of the same figure, except those involving *avg_vs*, the average absolute vertical speed of all aircraft within the sector.

Considering the graphs on the 4^{th} line (or the 4^{th} column), we see that for *avg_vs* the centroids of each workload class are more or less aligned. They are also fairly close to the x-axis, indicating that this *en-route* sector usually handles leveled traffic. Moreover, the dispersion of the vertical speeds is higher for low workload classes than for higher workload classes. Although this remains to be confirmed, this is probably because the biggest flows, inducing the highest workload in this sector, are the transatlantic flights and the northbound or southbound flights between the UK and southern europe, which contain mostly leveled flights.

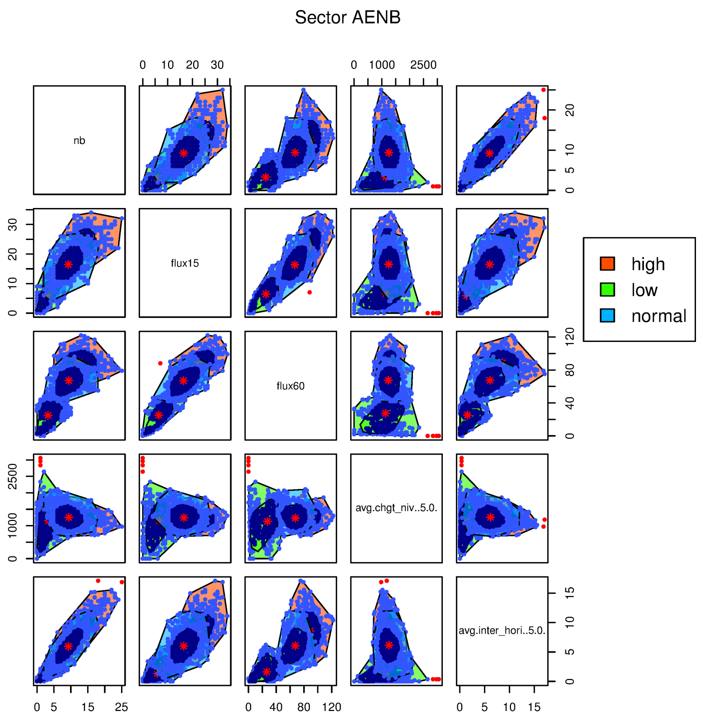


Figure 3: Bagplots for the sector AENB (Paris ATCC).

Now, considering Figure 3 representing bagplots for another sector AENB in the Paris area, we see on the graphs of the 4^{th} line that the median value for *avg_vs* is around 1200 fpm, whatever the workload class. This is probably because sector AENB is a pre-approach sector for Paris airports, in which nearly all the traffic is either climbing or descending. We see also that the three bagplots representing the low, normal, or high workload classes are more or less aligned horizontally, as for the sector N. The dispersion of the data is also higher in low workload classes than in higher workload classes.

The fact that the median value of *avg_vs* on Figures 2 and 3 remains more or less on a flat line whatever the workload class seems to contradict our intuitive notion that the cognitive workload of the controller should increase when more aircraft are climbing or descending in the sector, just as for the other explanatory variables.

However, looking more closely at the outer bag of the "normal workload" class, on Figure 2, 4^{th} line, 1^{st} column, we see that its shape is more or less triangular, with the pointy end towards high values of nb, the number of aircraft. The obliqueness of the upper boundary of this triangle tends to show that high values of *avg_vs* are more acceptable to the controllers for small values of nb than for high values, at least for this sector. A similar shape can be observed on many other sectors. So, within the normal workload category, our intuition that the cognitive workload increases with the number of climbing/descending aircraft might actually be true.

This relationship is simply difficult to observe at the macroscopic level: the median values of *avg_vs* are about the same whatever the workload category, which simply reflects the fact that there are no more climbing/descending aircraft, in our data, when the sector is split than when it is normally operated or collapsed.

In addition, the influence of *avg_vs* must be considered across all the sectors. When doing so, and when combining it with other variables such as the sector volume, the average vertical speed *avg_vs* does influence the workload categorization, as was shown in previous work on the selection of relevant explanatory variables ([13], [28], [15]). It remains to be seen if, for our purpose, it could be replaced with a categorical variable characterizing the sector (*en-route*, or *pre-approach*, for instance).

VIII. Conclusion

Let us now conclude this paper by summarizing our approach and our findings. We have looked into the performances of a workload model learned from historical data, using gradient boosted trees. The examples used to learn the model were made of ATC complexity measurements computed from radar records and sector data, and workload measurements extracted from past ATC sector operations. The three levels (low, normal, high) only give a rough indication of the workload. However, this workload measurement has the advantage of being easily available, in large quantities and for a great number of ATC sectors, because it can be directly extracted from historical records of past sector operations.

In previous works, this model showed an 82% rate of correct classifications, when training the model on one week of traffic, and assessing it on another week, considering approximately the same set of sectors in both the training and the test set. In the current work, our first objective was to look into the model's performances when the model is trained on a subset of sectors and assessed on a different subset. Our second objective was to examine more closely the relation between the input ATC complexity variables and the output (i.e. the workload class).

The results show that the overall performance of the model is slightly degraded, with a rate of correct predictions around 76%, when the training and test sets are geographically segregated (different sectors) instead of being temporally segregated like in our previous approach. The detailed results show that our model probably overfits the training data for elementary sectors, leading to poor generalization performance for these sectors. However, the model remains performant on all non-elementary sectors, even those having no occurence of the high workload class in the data.

Concerning the relations betwen the input variables and the workload output, the bagplots of section VII confirm some natural intuitions on the sense of variations of these quantities for most variables, except maybe for one, the average absolute vertical speed of all aircraft in the sector. For all variables except this one, we observe on two instances of sectors (one en-route and one pre-approach sector) that when the input ATC complexity metric increases, we are more likely to be in a higher workload category.

A natural intuition is that the cognitive workload of the controller should also increase when there are more climbing and descending aircraft in the sector, just as for the other variables. This relationship is most probably true, but it is difficult to observe in our macroscopic workload categorization, except maybe by looking closely at the dispersion of the data in the normal workload category.

To conclude, the model obtained with the GBM machine learning method cannot be interpreted only as a model of the cognitive workload of the controller. The model expresses relations among variables emerging from the data, and it can only be as good as the data that was used to train it. It cannot be transposed to any context without precautions. The fact that our workload model remains performant for all non-elementary sectors confirms that it can actually be used to predict optimal configurations of ATC sectors (sector opening schemes), where we only search to split or merge sectors optimally. This model should be completed or replaced by a more simple model when evaluating the workload in elementary sectors.

In future works, we might try to produce some artificial data samples of the "high" workload class for elementary sectors. This would force our model to correctly assess the boundary between normal and high workload for these specific sectors. Another approach that we could try is to apply one-class classification methods on the "normal" workload class, to detect when non-normal instances (underloads, or overloads) occur in elementary sectors. Other work might consider the seasonal variability in our data. It would be interesting to compare the performances of a same model tuned several times on data samples of different months.

References

[1] D. Gianazza. Learning air traffic controller workload from past sector operations. In *Proceedings of the 12th USA/Europe Air Traffic Management R & D Seminar*, 2017.

[2] R.H Mogford, J. A. Guttman, S. L. Morrow, and P. Kopardekar. The complexity construct in air traffic control: A review and synthesis of the literature. Technical report, FAA Technical Center: Atlantic City, 1995.

[3] P. Kopardekar. Dynamic density: A review of proposed variables. FAA WJHTC internal document. overall conclusions and recommendations, Federal Aviation Administration, 2000.

[4] B. Hilburn. Cognitive complexity in air traffic control, a litterature review. Technical report, Eurocontrol experimental centre, 2004.

[5] I. V. Laudeman, S. G. Shelden, R. Branstrom, and C. L. Brasil. Dynamic density: An air traffic management metric. Technical report, 1999.

[6] A. Majumdar, W. Y. Ochieng, G. McAuley, J.M. Lenzi, and C. Lepadetu. The factors affecting airspace capacity in europe: A framework methodology based on cross sectional time-series analysis using simulated controller workload data. In *Proceedings of the 6th USA/Europe Air Traffic Management R & D Seminar*, 2005.

[7] J.H. Crump. Review of stress in air traffic control: Its measurement and effects. *Aviation, Space and Environmental Medecice*, 1979.

[8] P. Averty, S. Athènes, C. Collet, and A. Dittmar. Evaluating a new index of mental workload in real ATC situation using psychological measures. Note CENA NR02-763, CENA, 2002.

[9] Caroline Martin, Julien Cegarra, and Philippe Averty. Analysis of mental workload during en-route air traffic control task execution based on eye-tracking technique. In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pages 592–597. Springer, 2011.

[10] P. Kopardekar and S. Magyarits. Measurement and prediction of dynamic density. In *Proceedings of the 5th USA/Europe Air Traffic Management R & D Seminar*, 2003.

[11] G.B. Chatterji and B. Sridhar. Measures for air traffic controller workload prediction. In *Proceedings of the First AIAA Aircraft Technology, Integration, and Operations Forum*, 2001.

[12] C. Mannings, S. Mill, C. Fox, E. Pfleiderer, and H. Mogilka. The relationship between air traffic control events and measures of controller taskload and workload. In *Proceedings of the 4th Air Traffic Management Research & Developpment Seminar*, 2001.

[13] D. Gianazza and K. Guittet. Evaluation of air traffic complexity metrics using neural networks and sector status. In *Proceedings of the 2nd International Conference on Research in Air Transportation*. ICRAT, 2006.

[14] D. Gianazza and K. Guittet. Selection and evaluation of air traffic complexity metrics. In *Proceedings of the 25th Digital Avionics Systems Conference*. DASC, 2006.

[15] D. Gianazza. Smoothed traffic complexity metrics for airspace configuration schedules. In *Proceedings of the 3nd International Conference on Research in Air Transportation*. ICRAT, 2008.

[16] GM Flynn, A Benkouar, and R Christien. Adaptation of workload model by optimisation algorithms. Technical report, Eurocontrol, 2005.

[17] Jerry D Welch, John W Andrews, Brian D Martin, and Banavar Sridhar. Macroscopic workload model for estimating en route sector capacity. In *Proc. of 7th USA/Europe ATM Research and Development Seminar, Barcelona, Spain*, 2007.

[18] B. Sridhar, K. S. Sheth, and S. Grabbe. Airspace complexity and its application in air traffic management. In *Proceedings of the 2nd USA/Europe Air trafic Management R&D Seminar*.

[19] D. Gianazza, C. Allignol, and N. Saporito. An efficient airspace configuration forecast. In *Proceedings of the 8th USA/Europe Air Traffic Management R & D Seminar*, 2009.

[20] D. Gianazza. Forecasting workload and airspace configuration with neural networks and tree search methods. *Artificial Intelligence Journal, Elsevier*, 174(7-8):530–549, may 2010.

[21] N. Saporito, C. Hurter, D. Gianazza, and G. Beboux. A participatory design for the visualization of airspace configuration forecasts. In *Proceedings of the 4th International Conference on Research in Air Transportation*, 2010.

[22] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 6. Springer, 2013.

[23] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[24] Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 4. springer New York, 2006.

[25] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[26] Jerome H Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002.

[27] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[28] D. Gianazza. Airspace configuration using air traffic complexity metrics. In *Proceedings of the 7^{th} USA/Europe Seminar on Air Traffic Management Research and Development*, 2007. best paper of "Dynamic Airspace Configuration" track.