Predicting sector configuration transitions with autoencoder-based anomaly detection

Thomas Dubot
ONERA / DTIS, Université de Toulouse
F-31055 Toulouse - France
thomas.dubot@onera.fr

Abstract—One of the key challenges of airspace configuration is to ensure “smooth” transitions between consecutive sector configurations, optimized at each time period. Optimization models therefore need to assess the appropriate transition cost from one configuration to another. The aim of the research presented in this paper is to develop a machine learning algorithm, trained with historical data of realized sector configuration transitions, to evaluate this cost. An anomaly detection model is used to quantify the abnormality of sector configuration transitions based on the reconstruction error of an autoencoder (or Replicator Neural Network). Results obtained on a French Area Control Center (Bordeaux) show that the model is able to predict promising transitions never realized in the past and poor transitions very unlikely to happen.

Keywords: sector configuration; transition cost; machine learning; anomaly detection; autoencoder; reconstruction error

I. INTRODUCTION

Each Area Control Center (ACC), in charge of providing air traffic control services to controlled flights within its airspace, is subdivided into elementary sectors that are used or combined to build control sectors, operated by a team of air traffic controllers. Airspace sectorization consists in partitioning the overall ACC airspace into a given number of these control sectors. In most centers, the set of control sectors deployed, the sector configuration, varies throughout the day. Basically, sectors are split when controllers’ workload increases, and merged when it decreases. We call dynamic airspace configuration the process of building this schedule of reconfigurations. The initial configuration plan (or opening scheme) is hence updated, based on the latest information on traffic demand and controllers’ availability.

Today, control centers rely deeply on the human operational expertise of flow management positions to compare the traffic demand with the sector capacity and choose the adequate sector configurations throughout the day. This way, only a small subset of predefined configurations is used, instead of exploring all the possible combinations of sectors. In addition, the usual hotspot resolution method consists in splitting the overloaded sector into two sectors, which is not optimal (increase of the number of control positions) and sometimes impossible, e.g. if the hotspot occurs at the level of an elementary sector.

Many optimization methods have been explored in past decades to address the sectorization problem, such as constraint programming [1,2], mixed integer programming [3], evolutionary algorithms [4,5], graph partitioning algorithms [6] or tree search methods [7]. If most of these methods succeed in providing optimized sector configurations at a given time period, very few have been integrated in decision support tools to help operational experts to optimize the airspace configuration process. One of the main explanations is at the heart of the airspace configuration problem: juxtaposing optimized sector configurations most often leads to an unacceptable sector configuration plan, as each reconfiguration implies an abrupt change of sectors. Too many changes can lead to confusion and potential safety issues: air traffic controllers may have difficulties to adapt to unfamiliar new sectors; some dangerous traffic situations could be missed by the new controller in charge; some flights could reenter the same sector, etc.

The stability (or continuity) between consecutive sector configurations is hence a key parameter to consider in the optimization process. Some airspace configuration models have introduced a transition cost linked to each reconfiguration. It is commonly agreed that reconfigurations should mirror the current reality of operations: minimizing changes based on split/merge processes or maximizing the number of airspace entities shared between two consecutive configurations. Nevertheless the definition of a distance between two sector configurations is a challenging issue as many rules/exceptions are inherent to each operational use case. Moreover, the analysis of historical data shows that some transitions with minor changes, e.g. based on a single split, are never realized. Inversely, some transitions are sometimes deployed by operational experts despite their apparent complexity.
In this paper, we present a machine learning method which evaluates the abnormality of sector configuration transitions, based on the analysis of historical data. This paper is organized as follows: Section II presents an overview of related works. Section III explains how autoencoders can be used to quantify an abnormality. Section IV presents the model developed to predict sector configuration transitions and some results obtained with operational data of a French ACC. Section V explains how to use this model, especially to evaluate the “distance” between two distant configurations from the reconstruction error of the autoencoder. Finally, conclusions are presented in Section VI.

II. PREVIOUS RELATED WORKS

Whereas some studies such as [8] have identified the need to take into account the airspace reconfiguration workload [9] in the sectorization process, and some others [10,11] have introduced reconfiguration complexity metrics, most airspace configuration models developed in the past [12] ignore the transition cost, or only control the frequency of reconfigurations [13].

Two main approaches have been proposed to take into account the stability of consecutive configurations. The first one consists in ensuring that some sectors or airspace volumes remain identical in source and target configurations. In [14], Gianazza addresses this constraint of stability by keeping the current airspace configuration as long as it remains acceptable. Besides, when a reconfiguration is triggered, it can be limited to a local recombination. Only a few sectors are selected, e.g. the sectors with too low or too high workload evaluation (and potentially their neighboring sectors), and a new optimal partition is recomputed from this set of airspace modules. In [15], non-sharable airspace volumes are defined as root nodes in the first stage of the genetic algorithm. For successive time periods, the same root nodes are used as sectors’ centers, ensuring that controllers still control at least this volume of airspace.

The second approach relies on the introduction of a transition cost in the objective function, such as the difference of the number of control sectors in consecutive configurations. Other models use the number of new control sectors compared to the previous time step or inversely the number of control sectors that do not appear anymore [16]. In [17], the EUROCONTROL Improved Configuration Optimization (ICO) is described as a multi-objective optimization with a criterion called continuity ratio, defined as the number of airspace entities in common between two configurations at different time steps.

We described in [18] an algorithm to generate smooth sector configuration plans. In a first phase, a large number of good solutions were saved for each time period, with the use of the Pareto frontiers of a multi-objective optimization. Then, we built a sequence of configurations with a shortest path algorithm based on a shared-cell metric using the notion of partition-distance [19]. This distance was defined as the total density workload of the building blocks that are moved from a subset to another while switching from one configuration to another. In [20], we modified this function to favor split/merge operations, assuming that such operations did not introduce any reconfiguration workload for the controllers, and hence could be evaluated as perfect.

All the optimization methods presented in this section rely on the implementation of logical rules to mirror current operational habits. The common objective is to minimize the number of changes between consecutive configurations, and to use regular transformations, such as split and merge operations. Nevertheless, the fine-tuning of these logical rules requires multiple feedbacks with operational experts to model specific operational customs. This process is costly and often leads to algorithms that cannot be generalized to other ACCs. Additionally, the chosen distance functions prove to be often insufficient to correctly classify new transitions. For example, if the transition cost is limited to the difference of number of control sectors, some reconfigurations may be favored despite the fact that all sectors are changed from a time step to another. In the case of the cell-based metric described in [20], any transition based on a split or merge operation will be favored even if the analysis of historical data highlights that they have never been realized. Inversely some transitions are filtered in spite of their regular use.

This paper presents a method to quantify the abnormality of a transition, based on the analysis of past historical data, in order to enrich the transition cost with operational experience.

III. QUANTIFYING ABNORMALITY WITH AUTOENCODER RECONSTRUCTION ERROR

A. The Machine Learning approach

Machine learning methods allow to directly learn from historical data, i.e. sector configuration transitions realized in the past. A first intuition would be to use a binary classification method, considering all possible transitions between configurations previously deployed. When the transition occurred in the past, the transition is labeled as a positive example, and inversely a transition never realized is labeled as a negative example. It would hence be possible to predict the feasibility of a new transition (between configurations never deployed) with the classifier model. Nevertheless this model relies on the principle that a transition which has never been realized is a bad transition and this assumption is denied by operational experts. Then, the use of regression methods to predict the occurrence of a transition is also excluded, as two transitions which are deemed comparable by operational experts might be used differently depending on many other factors, such as center or training habits.

Unsupervised anomaly detection methods seem more adapted to the airspace configuration problem. In anomaly detection, the system is trained with normal instances (in our
case the transitions realized in the past) and when it “sees” a new instance (transition), it can tell whether it looks like a normal one or whether it should be considered as an anomaly or outlier. Many mathematical models, such as those presented in [21], can predict whether a new observation belongs to the same distribution as existing observations. Most of the time, these methods allow to filter new instances but cannot quantify their degree of abnormality. The method presented in this paper is based on the use of autoencoders, or Replicator Neural Networks (RNN), for anomaly detection, as initially proposed by [22]. Anomalous transitions can be detected and quantified. The degree of abnormality of each transition can then be used to predict the probability of use of each sector configuration transition.

B. Artificial Neural Networks

Artificial Neural Networks (ANNs) are machine learning systems inspired by biological neural networks. An ANN is a network of elements called neurons, which receive input, change their internal state (or activation) according to that input, and produce output depending on the input and activation. The network formed by connecting the output of certain neurons to the input of other neurons can be represented as a directed weighted graph. If the graph is acyclic, the network is called feedforward. Each ANN has an input layer, an output layer, and one or several middle layer(s) called hidden layer(s). We call Deep Neural Networks ANNs with multiple hidden layers. Each layer (except the input layer) is associated with an activation function computing the activation of each neuron based on inputs \( p_i \) from its predecessor neurons.

\[
a_j = f\left( \sum_i W_i p_i + b \right)
\]

with \( W_i \) representing the weight given to the input \( p_i \) and \( b \) the intercept term or bias.

Considering a training example \((x, y)\), the ANN can be used to define a complex, non-linear form of hypotheses \( h_{W,b}(x) \) with parameters \( W, b \) computed to approximate \( y \). A cost function such as the mean squared error (MSE) is then used to assess how well the neural network maps training examples to correct output \( y \). The objective of the training phase is to find the parameters minimizing the cost function for all training examples. The weights are hence modified by using backpropagation [23] to calculate the gradient of the loss function.

C. Autoencoders

Autoencoders are feedforward ANNs, with an input layer, an output layer with the same number of nodes, and one or more hidden layers, built so that they reconstruct their own inputs through an encoding and a decoding part.

As the output reproduces the input, autoencoders can be used with unlabeled training data \( x \), in an unsupervised learning context. Fig. 1 gives an illustration of an autoencoder with six nodes in input and output layers (plus bias unit) and one hidden layer with only three nodes.

Let us formalize such an autoencoder with an input layer of \( d \) nodes and one hidden layer with \( d' \) nodes. First the autoencoder encodes an input \( x \in \mathbb{R}^d \) to a hidden representation \( y \in \mathbb{R}^{d'} \) through a deterministic mapping:

\[
y = f(W_1 x + b_1)
\]

where \( W_1 \) is the weight encoding matrix, \( b_1 \) an encoding bias vector and \( f \) an activation function such as the sigmoid function:

\[
\sigma(z) = \frac{1}{1 + e^{-z}}
\]

The latent representation \( y \), or code, is then mapped back with the decoder into a reconstruction \( \hat{x} \) of the same shape as \( x \), with a similar transformation:

\[
\hat{x} = g(W_2 y + b_2)
\]

where \( W_2 \) is the weight decoding matrix, \( b_2 \) a decoding bias vector and \( g \) another activation function, which can be the same as \( f \).

The parameters of this model are optimized such as to minimize an average reconstruction error \( L \):

\[
(W_1, b_1, W_2, b_2) = \arg \min_{W_1, b_1, W_2, b_2} L(x, \hat{x})
\]
IV. APPLICATION TO THE PREDICTION OF SECTOR CONFIGURATION TRANSITIONS

A. Feature selection

This machine learning method is implemented with historical data from the French ACC of Bordeaux. Several years of records are used to obtain a set of realized sector configurations and a set of realized sector configuration transitions.

Based on the state of source and target configurations, or the nature of the transition leading from one to another, a vector of 55 normalized features $x^{(i)} \in [0,1]^{55}$ is then extracted from each transition, as illustrated by Fig. 2. For example, the following features are used:

- Number of control sectors in each configuration
- Number of elementary sectors in each configuration
- Size of control sectors used, i.e. the number of elementary sectors grouped together (min, max, mean)
- Shape of control sectors used, i.e. the altitude/latitude/longitude range of control sectors (min, max, mean)
- Cell-based distance between configurations, i.e. the number of airspace volumes neither in common nor included in each other
- Number of control/elementary sectors covering each ACC cluster, i.e. predefined geographic groups of sectors

These features have been selected to be as generic as possible, in order to easily apply the model to another ACC. However, the definition of clusters would have to be adapted, depending on the organization of the ACC.

![Figure 2. Selection of transition features](image)

B. Neural network hyperparameters and performance measure

As introduced in section III, Hawkins et al. [25] demonstrate the possibility to use autoencoders in the framework of anomaly detection to produce an anomaly score from the reconstruction error. Their neural network was an autoencoder with three hidden layers using the activation function for the two outer hidden layers and a staircase like activation function in the middle hidden layer. However, several studies, such as [26], demonstrate that for other use cases, a network with only one single layer could perform as well, if not better. The model presented here is a simple autoencoder with one hidden layer, as a first approach. In [27], Dau et al. show that on different data sets, the number of neurons in the hidden layer equal to the input-output layers yields good detection rate. We discuss here on such a solution with the use of the sigmoid activation function for all layers. The reconstruction error $L$ is finally assessed using the mean squared error estimator.

C. Training the model

Let us call $n$ the total number of data samples, i.e. the transition vectors described in IV.A and $m$ the number of samples used for the training:

$$\{x^{(1)}, x^{(2)}, x^{(3)}, ..., x^{(m)}\}$$

where $x^{(i)} \in \mathbb{R}^d$, $i \in [1, m]$

The number of samples $n$ is equal to 2617 and $m$ is chosen so that 80% of data are used in the training phase, and 20% are kept to check the generalization of the model during the test phase. The number of nodes $d$ is equal to 55, the size of the features described in IV.A.

The parameters are optimized to minimize the loss, i.e. the reconstruction error of all the training samples, which can be noted, following the III.C paragraph:

$$(W_1, b_1, W_2, b_2) = \arg \min_{W_1, b_1, W_2, b_2} \frac{1}{m} \sum_{i=1}^{m} \|x^{(i)} - \hat{x}^{(i)}\|^2$$

where $\hat{x}^{(i)} = \sigma(W_2(\sigma(W_1 x^{(i)} + b_1)) + b_2)$

with $\sigma(z) = \frac{1}{1+e^{-z}}$

and $W_1, W_2 \in \mathbb{R}^{d \times d}$

The optimization is realized using the adaptive moment estimation ADAM [28] method, which relies on exponentially decaying average of past gradients and past square gradients. Partial derivatives are calculated by reverse-mode automatic differentiation. These methods are implemented in the open-source TensorFlow library [29]. Fig. 3 shows the evolution of the reconstruction error of the training set (blue curve) during the learning optimization process.
In the same time, we evaluate the reconstruction error of the test set consisting of the data samples that were not used during the training phase.

\[
x^{(m+1)}, x^{(m+2)}, \ldots, x^{(n)}
\]

where \( x^{(i)} \in [0,1]^d, \forall i \in [m+1,n] \)

Training and test curves show a convergence after a maximum of \(10^3\) iterations.

It has to be noted that if the reconstruction error tends to zero, each data sample (training set and test set) has a reconstruction error strictly greater than zero.

\[
L(x^{(i)}, x^{(i)}) \in [0,1]
\]

This ensures that even if the number of neurons in the hidden layer is the same as in the input/output layers, the model does not only reproduce the identity function.

D. Validation of the model

The model is finally applied to all sector configuration transitions: realized transitions used in the training set, realized transitions not used during the training phase, and transitions never realized, and therefore not in the training set.

<table>
<thead>
<tr>
<th>Set</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Realized transitions used in training</td>
<td>Min error</td>
</tr>
<tr>
<td>(training set)</td>
<td>7.21e-6</td>
</tr>
<tr>
<td>Realized transitions not used in training</td>
<td>7.17e-6</td>
</tr>
<tr>
<td>Transitions never realized</td>
<td>7.13e-6</td>
</tr>
</tbody>
</table>

Table I displays some statistics on the reconstruction errors obtained for all these transitions. The scores obtained by realized transitions not used in training are comparable to those of the training set. Only 0.77% of the transitions have a reconstruction error higher than the maximum reconstruction error of the training set. Besides, if we analyze the transitions with highest errors, we observe that these transitions occurred very rarely in the last years, as illustrated by Fig. 4. It has to be noted that the model was built without the knowledge of the number of occurrences of each transition in the training set. Inversely, if we analyze the scores of all the transitions never realized, mean and max reconstruction errors are significantly above the training scores, and 43.58% of the transitions are above the maximum error of the training set. Such results show that the model generalizes well and that the reconstruction error could be used to classify or filter transitions never realized.

In a first approach, the maximum reconstruction error of the training could be used as a threshold to filter relevant transitions never realized. Inversely, a very low score could highlight promising transitions.

E. Experimental evaluation

In order to assess the usability of the reconstruction error score, we select a set of transitions with one single change based on the split of one sector. The target configuration has hence two additional sectors, only one missing from the source configuration, and all the others identical. As analyzed in section II, most models developed in the past have a transition cost based on the minimization of changes and the use of split/merge operations. Such transitions would therefore be assessed as valid by these models, without any possible differentiation between them.
As the anomaly detection model presented in this paper can quantify and classify such transitions through the reconstruction error, we asked one operational expert of the ACC to assess these transitions, with four different levels:

- **Level 1**: “I am pretty sure that this transition has already been realized”
- **Level 2**: “This transition seems realistic but I am not completely sure that it has been realized”
- **Level 3**: “There is nothing wrong with this transition but I doubt that it has already been realized (for various reasons)”
- **Level 4**: “I am pretty sure that this transition has never been realized, and will never be realized (for various reasons)”

As illustrated by Fig. 5, popular transitions already realized (in yellow) were immediately assessed as valid. Transitions never realized with a very low error (in green) were assessed as probable, even if some of them were judged surprising. Inversely, transitions with high reconstruction error score, above or below the maximum training threshold, were deemed unlikely to happen. It is difficult to totally exclude a transition, i.e. to evaluate it as level 4, because even if they seem clearly improbable, they could be used in exceptional circumstances, e.g. ash clouds. These results confirm the relevance of the reconstruction error score provided by the anomaly detection model, and show how it could be used to classify, if not filter, some transitions that would be considered as classic valid reconfigurations by other models.

### V. USE OF THE ANOMALY DETECTION MODEL

#### A. Analysis of sector configuration transitions

Another interesting benefit of this method is that it can be used as a data mining tool to analyze why some reconfigurations seem more probable than others.

![Figure 5. Operational evaluation versus model predictions](image1)

![Figure 6. Example of reconstruction error per subdimension](image2)

![Figure 7. Reconstruction error and variation of sectors](image3)
Transitions implying a large difference of sectors’ number are very unlikely to happen. This makes sense from an operational viewpoint, as usual reconfigurations mainly imply a single split/merge operation, as illustrated by Fig. 8.

In the model presented in this paper, we did not filter any transition based on such operational knowledge, in order to check if the machine learning model could find usual logical rules from scratch. The identification of such logical rules can accelerate the classic optimization process, as discussed in Section II, by analyzing the feedbacks from operational experts on logical associations automatically detected.

**B. Towards a distance function between sector configurations**

In the end, if the transition between two configurations seems improbable according to the value of the reconstruction error, it would be interesting to evaluate how far the configurations are, i.e. in how many intermediate steps it would be probable to transit from one configuration to another. The reconstruction error could be used, within other parameters such as the frequency of use (if already used), to compute a probability \( w_i \) for each transition.

We can then build a probabilistic graph \( G = (V, E) \) with all sector configurations as vertices and edges between configurations (transitions) valued with the probability based on the reconstruction error. In order to find the most probable path between a source configuration and a target configuration, we use a classic problem reduction method using the logarithm function transformation [30].

Our objective is to determine the path maximizing the product of transitions probabilities:

\[
P_v = \arg\max_w \prod_{w_i \in w} w_i
\]

Let us consider the graph \( G' = (V', E') \) so that \( G \) and \( G' \) are isomorphic:

\[
G' \cong G
\]

and \( w'(u,v) = -\log(w(u,v)), \forall (u,v) \in E' \)

We can then reduce the initial problem as a shortest path problem in \( G' = (V', E') \) as:

\[
\arg\min_{w'} \sum_{w_i \in w'} w_i' = \arg\max_w \sum_{w_i \in w} \log(w_i) = \arg\max_{w}(\log(\prod_{w_i \in w} w_i)) = \arg\max_{w}(\prod_{w_i \in w} w_i)
\]

The use of shortest path algorithm, such as Dijkstra’s algorithm [31] hence gives a probabilistic path between the source and the target configuration.

The number of intermediate configurations can be used as a quasimetric to assess the “distance” between two sector configurations. It has to be noted that such a method can become rapidly computationally expensive if we consider all the possible transitions between sector configurations. In that case, the graph is a complete graph, as illustrated by Fig. 9 with nine nodes. Such a complete graph ensures finding a path for every pair of configurations, but its number of edges \( |E| = \frac{n(n-1)}{2} \) can prevent fast computation.

**VI. Conclusion**

In order to ensure smooth reconfigurations between sector configurations, airspace configuration models must consider the cost induced by each transition. Logical rules such as the maximization of the number of sectors in common can be used but are insufficient to reproduce operational habits. Machine learning methods based on historical data could be an alternative or a complementary approach to predict the probability of a transition based on the transitions realized in the past.

The model presented in this paper is based on an anomaly detection method able to quantify the degree of abnormality of a transition. An autoencoder is used with a set of features extracted from sector configuration data. The reconstruction error of a new sample can be used to predict its probability to be used. Results on a French ACC show the capability of this model to capture the operational habits, and to detect the transitions likely to be deployed, even in the case of usual
transitions based on a single split/merge operation that would be impossible to sort with current airspace configuration algorithms. This model used as a first approach could be enhanced in the future, with different hyperparameters or another autoencoder type. Some other features could also be introduced to better model a transition. Secondly, such approach could be improved by the use of logical rules that could reduce useless evaluation of impossible transitions, or deliberately filter some samples, such as night reconfigurations, and hence better predict complex transitions. Finally it would be interesting to verify if the machine learning method described in this paper could be easily transposable to another ACC.

ACKNOWLEDGMENT

The author would like to thank Denis Bedouet from Bordeaux ACC for his numerous feedbacks and DSNA/ONERA colleagues that provided insight and expertise on this sector configuration transition problem. In addition, the author would like to express his gratitude to Judicaël Bedouet, Julien Bourget, Antoine Joulia, Jérôme Morio, Xavier Olive and ICRAT anonymous reviewers, for their valuable and constructive suggestions before and during the reviewing of this paper.

REFERENCES


