EXPLORATORY RESEARCH

sesar JOINT UNDERTAKING

Reference of XAI Methods

Deliverable ID:	**D4.2**
Dissemination Level:	**PU**
Project Acronym:	**TAPAS**
Grant:	**892358**
Call:	**H2020-SESAR-2019-2**
Topic:	**SESAR-ER4-01-2019 Digitalisation and Automation principles for ATM**
Consortium Coordinator:	**CRIDA**
Edition Date:	**10 June 2022**
Edition:	**00.04.00**
Template Edition:	**02.00.02**

Authoring & Approval

Authors of the document

Name/Beneficiary	Position/Title	Date
G. Vouros / **UPRC**	WP4 Leader	07/06/2022
T. Kravaris / **UPRC**	Project Member	20/05/2021
K. Lentzos / **UPRC**	Project Member	21/05/2021
G. Santipantakis / **UPRC**	Project Member	04/04/2022
A. Bastas / **UPRC**	Project Member	11/04/2022
G. Papadopoulos / **UPRC**	Project Member	15/04/2022

Reviewers internal to the project

Name/Beneficiary	Position/Title	Date
Valle, Natividad / **CRIDA**	Project Member	08/06/2022
Lema, Florencia / **CRIDA**	Project Member	08/06/2022
Iglesias, Enrique / **CRIDA**	Project Member	08/06/2022
Rodríguez, Rubén / **CRIDA**	Project Manager	08/06/2022

Approved for submission to the SJU By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
Rodríguez, Rubén / **CRIDA**	Project Manager	09/06/2022
Cordero, Jose Manuel / **CRIDA**	Project Coordinator	09/06/2022
Scarlatti, David / **BRTE**	BRTE PoC	09/06/2022
Andrienko, Gennady / **Fraunhofer**	Fraunhofer PoC	09/06/2022
Crook, Ian / **ISA**	ISA PoC	09/06/2022

Rejected By - Representatives of beneficiaries involved in the project

Name/Beneficiary	Position/Title	Date
N/A		

Document History

Edition	Date	Status	Author	Justification
00.00.01	06/05/2021	Draft	G.Vouros	TOC
00.00.05	11/05/2021	Draft	G.Vouros	ATFCM use case
00.00.06	11/05/2021	Draft	G.Santipantakis	Data sets & Preprocessing

EUROPEAN PARTNERSHIP

Co-funded by the European Union

00.00.07	20/05/2021	Draft	T.Kravaris	AI/ML methods
00.00.09	21/05/2021	Draft	K.Lentzos	XA methods
00.01.00	27/05/2021	Complete	G.Vouros	Reviewed & Completed
00.01.05	26/07/2021	Final	G.Vouros	Revised interim version taking into account SJU comments
00.01.06	01/04/2022	Draft	G.Vouros	ToC for the CD&R Use Case
00.01.07	04/04/2022	Draft	G.Santipantakis	Data sets & Preprocessing
00.01.08	11/04/2022	Draft	A.Bastas	Conflicts detection and conformance monitoring
00.01.09	15/04/2022	Draft	G.Papadopoulos	AI/ML methods for CD&R
00.02.00	19/04/2022	Complete	G.Vouros	Reviewed & Completed
00.03.00	01/06/2022	Complete	G.Vouros	Revised taking into account SJU comments
00.04.00	07/06/2022	Complete	G.Vouros	Revised taking into account round 2 SJU comments

Copyright Statement

EUROPEAN PARTNERSHIP

Co-funded by the European Union

TAPAS

TOWARDS AN AUTOMATED AND EXPLAINABLE ATM SYSTEM

This document is part of a project that has received funding from the SESAR Joint Undertaking under grant agreement No 892358 under European Union's Horizon 2020 research and innovation programme.

Abstract

This document provides a description of the XAI systems that have been implemented in the context of TAPAS, addressing the requirements of the ATFCM use case and the requirements of the CD&R use case. Specifically, the document describes the ATFCM and CD&R use cases and the specific problems that are considered, the data sets that are exploited by the corresponding systems, the AI/ML methods implemented and the explainability paradigm followed by each method. Experimental results show the quality of solutions provided.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Table of Contents

EUROPEAN PARTNERSHIP

Co-funded by the European Union

EUROPEAN PARTNERSHIP
Co-funded by the European Union

List of Figures

EUROPEAN PARTNERSHIP

Co-funded by the European Union

EUROPEAN PARTNERSHIP

Co-funded by the European Union

List of Tables

EUROPEAN PARTNERSHIP

Co-funded by the European Union

1 Executive Summary

This document is the final report that serves as a reference for the AI/ML and XAI methods implemented, addressing the requirements towards explainable Artificial Intelligence methods for the ATFCM and CD&R use cases [1] [2] [3], in the context of the TAPAS project. Therefore, for reasons of conciseness the document succinctly describes the ATFCM and CD&R use cases and the roadmap of the functionalities decided for each use case, up to realizing the needs for implementing automation level 2. Then it describes in detail the AI/ML method implemented per use case, and the types of solutions provided, as well as the explainability paradigm followed and the XAI method implemented. Experimental results from the AI/ML and XAI methods show the quality of the solutions computed.

Section 2 of this document specifies the ATFCM use case and problem addressed, and specifies the ATFCM prototype system, providing details on the AI/ML method implemented, the types of solutions provided and the explainability paradigm and functionality implemented.

Section 3 specifies the CD&R use case and problem addressed, and specifies the CD&R prototype system, providing details on the AI/ML method implemented, and the explainability (operational transparency) provided.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

2 Introduction

2.1 Intended Readership

This is a PU document.

2.2 Terminology and Acronyms

Term	Definition
AI	Artificial Intelligence
AoR	Area of Responsibility
AREDS	AoR Exit point to Downstream Sector
ATC	Air Traffic Control
ATCo	Air Traffic Controller
ATFCM	Air Traffic Flow and Capacity Management
ATM	Air Traffic Management
AU	Airspace User
CD&R	Conflicts Detection and Resolution
CPA	Closest Point of Approach
CTDE	Centralized Training Decentralized Execution
DCB	Demand and Capacity Balancing
DGN	Deep graph convolutional NNs
DQN	Deep Q Networks
DRL	Deep Reinforcement Learning
EC	European Commission
HEC	Hourly Entry Counts
LoSs	Loss of Separation
LTM	Local Traffic Manager
MAE	Mean Absolute Error
MDV	Multivariate Data Visualization
MED	Maximum Error Difference
ML	Machine Learning
NM	Network Manager
NMs	Nautical Miles

EUROPEAN PARTNERSHIP

Co-funded by the European Union

NOP	Network Operations Plan
OCC	Occupancy Counts
PER	Prioritized Experience Replay
RAoR	Relevant Area of Responsibility
RoC	Rate of Closure
SESAR	Single European Sky ATM Research
SGTs	Stochastic Gradient Trees
TAPAS	Towards an Automated and Explainable ATM System
UI	User Interface
VA	Visual Analytics
Vis	Visualization
XAI	Explainable AI

EUROPEAN PARTNERSHIP

Co-funded by the European Union

3 ATFCM Use Case

3.1 ATFCM Use Case Specification

The scope of this use case is limited to the detection, declaration and resolution of imbalances during the Pre-Tactical Phase, and particularly, focused on D-1 (day before operations). This includes not only imbalances induced by existing differences between demand and capacity, but also induced by the opportunity to improve performance levels.

The end user is the Local Traffic Manager (LTM).

The other involved stakeholders are the Air Traffic Service Unit Supervisor and the Network Manager.

Traffic demand from D-7 (seven days before operation) to D-1 (day before operations), that is Pre-Tactical Phase, is available, allowing the Imbalance Monitoring and Prediction Service to calculate imbalances using Hourly Entry Counts.

3.2 ATFCM Problem Specification

The DCB problem (or process) considers two important types of objects in the ATM system: *trajectories* and *airspace sectors*.

The *capacity of sectors* is of utmost importance: this quantity determines the maximum number of flights within a sector during a specific time interval.

Let there be **N** trajectories in a set **T** that must be executed over the airspace in a total time period of duration **H** (in hours: typically in 24 hours). The set of sectors in all possible airspace configurations is denoted by **S**. Time can be divided in intervals of duration Δ_t.

Each trajectory is a sequence of timed positions in airspace. This sequence can be exploited to compute the series of active sectors that each flight crosses- depending on the open airspace configurations, together with the entry and exit time for each of these sectors. Given a specific spatial region (in our case the Spanish FIR) the first (last) sector of the flight, where the departure (resp. arrival) airport resides, the entry (resp. exit) time is the departure (resp. arrival) time. However, there may exist flights that cross the airspace but do not depart and/or arrive in any of the sectors of our airspace: In that case we only consider the entry and exit time of sectors within the airspace of our interest.

Thus, a trajectory *T* in the set of trajectories considered, **T,** is a time series of elements of the form:

T=[($sector_1$, $entry_{t1}$, $exit_{t1}$) .... ($sector_m$, $entry_{tm}$, $exit_{tm}$))],

where $sector_i$ is in an open airspace configuration in **S**, $i=1,...m$.

Airspace sectorization changes frequently during the day, given different operational conditions and this should be considered when a flight expands in temporal periods with different sectorizations and/or when applying regulations to flights to resolve DCB problems: It must be noticed that given different regulations imposed to a trajectory, sectors crossed may differ, due to the changing sector configurations. This may result to a number of alternative representations of a single trajectory (each representation crossing a different set of sectors) one for each possible delay.

In addition to the above, re-routings have a direct effect on sectors crossed. Considering level capping measures, flights may cross an alternative set of sectors, compared to those planned. Thus, alternative

EUROPEAN PARTNERSHIP

Co-funded by the European Union

flight plans per flight can be considered in this case, prescribing alternative trajectories of the form specified above.

This information per trajectory suffices to measure the demand $D_{s,p}$ for each of the sectors s in the airspace in any period p of duration Δ_t.

Trajectories that co-occur in sector s and time period p and contribute to hotspots, are defined to be *interacting trajectories* for the period p and the sector s.

Given the demand per sector, the DCB problem consists of these cases where the demand exceeds capacity:

Each sector s in **S** has a specific capacity C_s. Imbalances of sectors' demand and capacity occur when $D_{s,p} > \%T*C_s$, for any period p of duration Δ_t in **H**, where %T gives a level of tolerance to exceeding capacity (currently set to 110%). These cases result to *hotspots* in the airspace.

In case of capacity violation for a period p and sector s, the interacting trajectories are defined as *hotspot-constituting trajectories*: one or more of these trajectories must be regulated in order to resolve the imbalance in s and p.

Imposing measures to trajectories may propagate hotspots to subsequent time periods for the same and/or other sectors crossed by that trajectory, or may direct trajectories to other sectors (e.g. due to level capping): In any case, the sets of interacting trajectories in different periods and sectors may change, and thus, in case of demand-capacity imbalances, hotspot-constituting trajectories may change as well. This can be done in many ways, either via imposing delays or by re-routing the trajectory to different sectors.

Towards the agent-based formulation of the problem, we consider the following:

Each agent A_i is specified to be the aircraft performing a specific trajectory in a specific date and time. Thus, we consider that **agents** and trajectories coincide in our case and we may interchangeably speak of agents A_i, trajectories T_i, or agents A_i executing trajectories T_i. Agents, as it will be specified, have own interests and preferences, although they are assumed to be collaborative, and take autonomous decisions on their measures: It must be noted that agents do not have communication and monitoring constraints, given that imbalances are resolved at the pre-tactical phase, rather than during operation.

Therefore, agents have to learn *joint* measures (ground delays and/or re-routings) to be imposed to their trajectories w.r.t. the operational constraints concerning the capacity of sectors crossed by these trajectories.

It must be noted that agents –although considered collaborative- have conflicting preferences, minimizing costs, while also executing their planned trajectories safely and efficiently.

Agents with interacting trajectories have to jointly decide on their measures: The decision of one of them affects the others.

The options available in the inventory $\mathbf{Ac_i}$ of any agent A_i for contributing to the resolution of hotspots may differ between agents: These, for agent A_i are the ground delay options $\mathbf{D}_i$ from 0 to $MaxDelay_i$, in conjunction to the re-routing options at any time step due to level capping measures, resulting to alternative flight plans $\mathbf{F}_i$. We consider that (a) delays may be combined with re-routings to provide joint measures for a single agent, thus $\mathbf{Ac_i = D}_i \times \mathbf{F}_i$, and (b) possible measures may be ordered by the preference of agent A_i to any such option, according to the function $\gamma(i)$: $\mathbf{Ac_i} \rightarrow \Re$. We do not assume that agents other than A_i have any information about $\gamma(i)$. This represents the situation where airlines set own options and preferences for measures even in different individual own flights, depending on

EUROPEAN PARTNERSHIP

Co-funded by the European Union

operational circumstances, goals and constraints. However, we expect that the order of preferences should be decreasing from 0 to $MaxDelay_i$, although, with a different pace for different agents.

Problem statement: Considering *any* two peers A_i, and A_j in the society (**A, E**), with A_j in $\mathbf{N}(A_i)$-{ A_i } (i.e. in the set of trajectories interacting with A_i] these agents must select among the sets of available options in $\mathbf{Ac}_i$ x $\mathbf{Ac}_j$, so as to contribute to the DCB problem solution quality (i.e. reduce the number of hotspots) w.r.t. their preferences on options $\gamma(i)$ and $\gamma(j)$.

According to the problem formulation stated above, and using the model of collaborative multi-agent MDP framework we assume:

- The *society of agents* **A**.
- A *time step* $t=t_0, t_1, t_2, t_3,..., t_{max}$, where t_{max}- t_0=**H and** t_{i+1} - t_i is constant for any i=0,…max and defines the time granularity of the agent-based simulation**.**
- A *local state* per agent A_i at time t, comprising state variables that correspond to (a) the measures applied to that $agent_i$ up to time point t, ranging to the sets of options assumed by A_i, (b) the hotspots and corresponding periods in hotspots in which A_i is involved in (for any of the sectors it crosses) and (c) time period in minutes where the agent is in each of the sectors crossed. Other parameters that contribute to problem awareness and provision of explanations on agent's decision-making logic in certain problems can be included, as these are also indicated by transparency/ explainability requirements (D3.1).

  Such a local state is denoted s^t_i.

  The *joint state* $\mathbf{s}^t_{Ag}$ of a set of agents A_g at time t is the tuple of the state variables for all agents in A_g. A *global (joint) state* $\mathbf{s}^t$ at time t is the tuple of all agents' local states.

  The set of all joint states for any subset A_g of **A** is denoted $\mathbf{State}_{Ag}$, and the set of joint society states is denoted by **State**.
- The *local strategy* for agent A_i at time t, denoted by str^t_i is the action that A_i performs at that specific time point: Such an action, in case the agent is still on ground, is whether to add delay until the next time step, according to time granularity. Simulating its flight (at the pre-tactical phase), the agent may decide, also in conjunction to delay, on possible re-routings (i.e. crossing different sectors instead of those initially intended to cross).

  The *joint strategy of a subset of agents* A_g of **A** executing their trajectories (for instance of $\mathbf{N}(A_i)$) at time t, is a tuple of local strategies, denoted by $\mathbf{str}^t_{Ag}$ (e.g. $\mathbf{str}^t_{\mathbf{N}(Ai)}$).

  The *joint strategy* for all agents **A** at any time instant t is denoted $\mathbf{str}^t$.

  The set of all joint strategies for any subset A_g of **A** is denoted $\mathbf{Strategy}_{Ag}$, and the set of joint society strategies is denoted by **Strategy.**
- The *state transition function Tr* gives the transition to the joint state $\mathbf{s}^{t+1}$ based on the joint strategy $\mathbf{str}^t$ taken in joint state $\mathbf{s}^t$. Formally

  Tr: **State** $\times$ **Strategy** $\rightarrow$ **State**.

  It must be noticed that although this transition function may be deterministic in settings with perfect knowledge about society dynamics, the state transition per agent is stochastic, given that no agent has a global view of the society, of the decisions of others, and/or of changing sector configurations, while its neighbourhood gets updated. Thus, no agent can predict how the joint state can be affected in the next time step. Thus, for agent A_i this transition function is actually

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Tr: **State**$_{Ai}$ × **Strategy**$_{Ai}$ × **State**$_{Ai}$ → [0,1], denoting the transition probability $p(s^{t+1}{}_i | s^t{}_i, str^t{}_i)$.

- The *local reward* of an agent A_i, denoted Rwd_{Ai}, is the reward that the agent gets by executing its own trajectory in a specific joint state of its peers in **N**(A_i), i.e. with any agent executing a trajectory in **Traffic**(A_i), according to the sectors' capacities, and the joint strategy of agents in **N**(A_i). The *joint reward*, denoted by Rwd_{Ag}, for a set of peers A_g specifies the reward received by agents in A_g by executing their actions in their joint state, according to their joint strategy.

  The reward Rwd_{Ag} for and subset *Ag* of **A** depends on the participation (contribution) of agents in hotspots occurring while executing their trajectories according to their joint strategy $\mathbf{str}^t{}_{Ag}$ in their joint state $\mathbf{s}^t{}_{Ag}$, i.e. according to their decided measures.

- *A (local) policy* of an agent A_i is a function π_i: **State**$_{Ai}$ → **Strategy**$_{Ai}$ that returns local strategies for any given local state, for A_i to execute its trajectory. The objective for any agent in the society is to find an optimal policy π_i that maximises the expected discounted future return

$$V_{Ai}\ (s) = max_{\pi i} E\left[\sum_{t=0}^{\infty} \delta^t * Rwd_{Ai}\left(s_{Ai}^t, \pi_i\left(s_{Ai}^t\right)\right) \vee \pi_i\right]$$

  for each state s_{Ai}^t, while A_i executes its trajectory, given a set of measures. The discount factor δ ranges in [0,1].

This model assumes the Markov property, assuming also that rewards and transition probabilities are independent of time. Thus, the state next to state *s* is denoted by *s'* and it is independent of time. Subsequently, subscripts and superscripts are avoided in cases where it is clear where a state or strategy refers to.

3.3 ATFCM Functional Roadmap

Based on the TAPAS ATFCM Use Case description, for reasons of conciseness we repeat here the functions / tasks have been identified. For each function / task, the description is provided, as well as the input, output and step of the ATM Master Plan Automation Levels Task Breakdown.

1. Traffic Demand Monitoring (pre-tactical phase, focused on D-1)

This function refers to the continuous traffic demand monitoring. This includes the monitoring of demand indicators.

2. Identification of imbalances

Once the traffic demand exceeds the established traffic monitoring values or thresholds, an imbalance between the traffic demand and the available capacity is identified.

3. Analysis of the imbalances detected

This task is devoted to the analysis of the characteristics of each one of the imbalances detected based on the violation of traffic demand indicators thresholds. The final objective is to determine if the imbalance is safety-critical or not.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

4. Identification of hotspots / optispots

Based on the nature of the imbalance, this function refers to the identification of the appropriate kind of spot, i.e. hotspot or optispot.

5. Declaration of hotspots / optispots

Once a hotspot / optispot is identified, the LTM will declare it to the Network Manager.

6. Preparation of DCB measures to solve the hotspot

This task refers to the preparation of the required DCB measures to be taken in order to solve the hotspot.

This preparation task also includes the selection of the candidate flights to be impacted by the DCB measures considered and the analysis of the DCB measure impact.

7. Decision on the DCB measure and flights impacted

Once the preparation of the DCB measure is finished, local criteria are considered to make the final decision on which measure should be applied and which flights are going to be impacted.

8. Implementation of the DCB measures

This function refers to the implementation, by the corresponding stakehoders (Network Manager, Airspace Users, etc.) of the DCB measures decided to be applied.

9. Hotspot Resolution monitoring

For the different hotspots declared and the multiple DCB measures taken, the status of the hotspot, that is, if it is resolved or not, should be monitored in order to detect any possible deviation due to the volatility of the air traffic demand.

Considering the functions identified the following table depicts the allocation of tasks between the human and the machine for the ATFCM Use Case following the Automation Levels considered by the European ATM Master Plan. For the definition of this functional roadmap, it should be noted that mainly the opinion of the Advisory Board and operational experts from TAPAS consortium have been taken on board.

ATFCM Functions / Tasks	Automation Level 1	Automation Level 2	Automation Level 3
Traffic Demand Monitoring	Machine	Machine	Machine
Identification of imbalances	Machine	Machine	Machine
Analysis of the imbalances detected	Human	Machine	Machine
Identification of hotspots / optispots	Human	Machine	Machine
Declaration of hotspots / optispots	Human	Human	Machine
Preparation of DCB measures to solve the hotspot • Capacity Measures (sector configuration) • ATFCM Scenarios • Trajectory Measures • ATFCM Regulation • Selection of candidate flights • Analysis of the DCB measure impact – What-if	Human	Machine	Machine
Decision on the DCB measure and flights impacted	Human	Human	Machine

EUROPEAN PARTNERSHIP

Co-funded by the European Union

ATFCM Functions / Tasks	Automation Level 1	Automation Level 2	Automation Level 3
Implementation of DCB measures	Human	Human	Machine
Hotspot resolution monitoring	Human	Machine	Machine

Table 1. TAPAS ATFCM Functional Roadmap

The rationale behind the allocation of tasks between the human and the machine illustrated in Table 1 is highly based in TAPAS Advisory Board and operational expert meetings. The following justifications for the TAPAS ATFCM Roadmap proposal have been considered:

1. Automation Level 1 reflects the current general situation at ECAC level in terms of ATFCM, although some ANSPs might be already in Automation Level 2 for some tasks.
2. The allocation of tasks for Automation Level 2 was proposed as the logic follow up of Automation Level 1, supporting the human in action implementation for the tasks for which a further step in automation was considered as needed and beneficial.
3. The distribution of tasks between human and machine for Automation Level 3 was done considering that all the tasks addressed in the TAPAS ATFCM use case could be initiated by automation in nominal cases, leaving Automation Level 4 and Automation Level 5 out of the scope of TAPAS with the consideration of non-nominal cases, as explained below.

Regardless the automation level and the environment conditions (nominal or non-nominal), the human might be able to take control of the system at any time. In addition, for non-nominal situations, the human will continue to have automated support for executing the different tasks, but the control will rely on the human and not in the machine.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

3.4 Data Sets

Data sets include the Flight Plan and Sector Configuration data sets, briefly introduced in the next paragraphs.

ALL\FT+ (Traffic) Files

- Spatial Coverage: Worldwide
- Temporal Coverage: 18th July to 14th August 2019 (28 days)
- Format: CSV files compressed to 7z. Each file corresponds to a separate day (each uncompressed file is approximately 1.9GB). Total number of records (in all files) are 1,074,111.

Sector Configuration

- Spatial Coverage: Worldwide (more detailed in European airspace)
- Temporal Coverage: (AIRAC 1908) Wednesday, July 17, 2019 9:00:00 PM UTC to Wednesday, August 14, 2019 8:59:00 PM UTC
- Format: A set of CSV files.

The Sector Configuration data set contains information about 50551 sectors.

3.4.1 Flight Plans

This data set provides the data about intended (FTFM), regulated (RTFM), and actual (CTFM) flight trajectories. It has worldwide spatial coverage for a complete year (2019). Data is provided in a compressed archive (7z compression) for each month (varying between 3 to 5GB per month). The compressed data set for entire 2019 is approximately 48.5GB. Each compressed archive contains a set of CSV files, one for each day of the archived month. The CSV files are in the ALLFT+ format, described in "DDR2 Reference Manual For General Users 2.9.7".

Finally, we categorize the trajectories found in the flight plans data set, according to their spatial relation to the geometry of LETOT (the part that covers Iberian Peninsula). Specifically,

a. if the geometry of a trajectory is completely covered by the LETOT geometry, the trajectory is classified as "**closed**" (i.e. both departure and destination airports are within Spanish airspace);
b. the trajectory is classified as "**disjoint"** if no common point between the two geometries exists and it is classified as
c. "**incoming"** or "**outgoing**", if the geometry of the trajectory is partially covered by LETOT and the trajectory ends in LETOT or originates from LETOT, respectively;
d. The trajectory is classified as "**overflight**" when the geometry of the trajectory is at least partially covered by LETOT but neither the trajectory ends in LETOT nor it originates from LETOT; and finally,
e. a subset of incoming trajectories are distinguished as being "**close to the Spanish airspace**", if they penetrate LETOT geometry during their take off phase.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

3.4.2 Sector Configuration

This data set describes the airspace configuration, i.e. the airspace volumes composed by 3D "airblocks", as well as their capacities, and activation intervals. The data set spatially covers the whole world, however it is more detailed in the European airspace. The first sample of the data source that have been provided, temporally covers AIRAC 1908, from Wednesday, July 17, 2019 9:00:00 PM UTC to Wednesday, August 14, 2019 8:59:00 PM UTC.

Since we address the ATFCM use case in the Spanish airspace only, we need only the sectors that their 2D geometry is spatially covered by the interior and the perimeter of the geometry of LETOT (at Iberian Peninsula). The 397 sectors that satisfy this spatial filter, are illustrated in Figure 1.

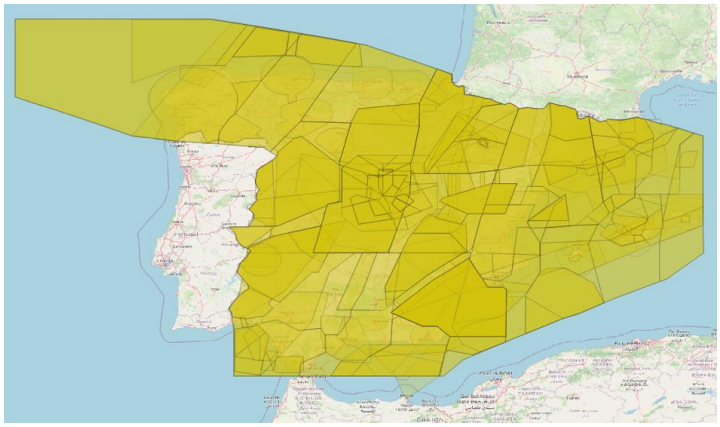


Figure 1. Sectors used in the trajectory enrichment task

The capacity of an airspace sector is evaluated only during the activation intervals of the sector. Formally, a capacity C defined in the data source for some interval I for an airspace sector S, if S is active for some interval A and the intersection of intervals A and I is not empty. The sectors that satisfy the LETOT spatial filter and also have a capacity value defined for at least one of their activation intervals are 132 for AIRAC1908.

3.5 Overall ATFCM Prototype

Figure 2 provides the overall architecture of the system, including, data preparation, AI/ML, Explainability module(s) and Explainability Logic as part of the Visual Analytics component.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

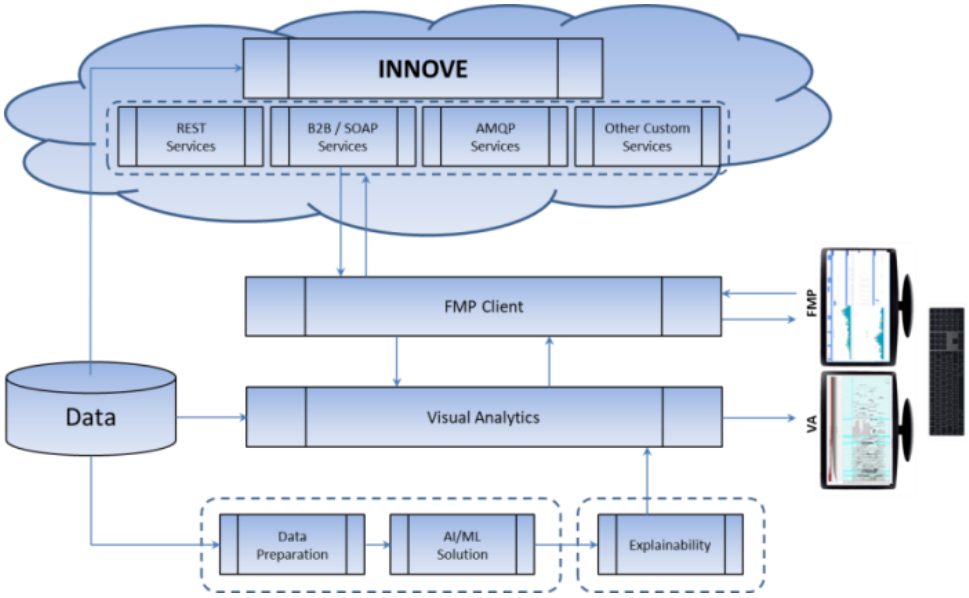


Figure 2. Overall System and Interaction with FMP

Given the **data sources** (as these are specified in section 3.2), these are exploited

(a) Via the **Visual Analytics** component, that provide data exploration facilities, and

(b) The **data preparation component** for clearing data and linking data sets to provide the representations of flights' intended trajectories, as these are specified in Section 2.2., and in the form required from the AI/ML module.

The building blocks for the ATFCM use case from our point of view, are aircraft trajectories compiled from the point profile of their flight plans (FTFM), and the airspace configurations.

Airspace configurations define the geometrical boundaries of airspace sectors, the activity intervals (i.e. temporal intervals for which the airspace is active), as well as their capacity.

It is essential that airspace sectors that are active at a time instance to be associated to corresponding positions of the trajectory. We call this process as **(trajectory) data enrichment**, since it associates each position with contextual information.

The data enrichment, in addition to the above, involves the computation of entry/exit positions for each crossed active sector, and the corresponding entry/exit time instances. The computed positions are also added in the trajectory and annotated as "inferred" positions, to indicate the transitions between sectors.

The **AI/ML** algorithms train deep models for solving the DCB problems identified, and provides, together with potential solutions, information for explaining decision making.

It must be noted that in Automation level 2 the system may provide more than one alternative solutions with appropriate explanations, while in automation level 3 the system provides one solution.

The **explainability components** exploit AI/ML deep models' output to train transparent models that provide explanations' content to the explanation logic.

The **explanation logic** realizes surface representations and renders textual and visual information in appropriate displays: together with **visual analytics** provide advanced facilities for exploring,

EUROPEAN PARTNERSHIP

Co-funded by the European Union

transforming, comparing, filtering and/or projecting data from the data sources and from the explainability components.

The purpose of the document is to describe the AI/ML methods used, as well as the explainability components. Explanation logic and visual analytics methods are described in D4.3 to be delivered in parallel to this document. Also, interactions among components are specified in D4.1 Tapas Integrated Prototype.

3.6 AI/ML component

This component implements the core and most of the functionality regarding the ATFCM prototype:

- Traffic Demand Monitoring (pre-tactical phase, focused on D-1)
- Identification of imbalances
- Analysis of the imbalances detected
- Identification of hotspots / optispots
- Declaration of hotspots / optispots
- Preparation of DCB measures to solve the hotspot
- Decision on the DCB measure and flights impacted

Actually, it prescribes the measures to be taken towards resolving imbalances in critical cases (i.e. in cases where the demand, measured by means of the Hourly Entry Count measure, exceeds %T of sector capacity). To do that, it implements an agent-based simulation, where agents (corresponding to flights) decide on level capping measures and minutes of ground delay. Specifically, at each simulation step agents decide on additional delay of 0…10 minutes to be taken according also to the ATFCM problem specification specified in Section 2.2.

Agents, having been notified by the demand per active sector they cross, identify the imbalances and hostpots, prepare types of measures to be taken to resolve any hotspot, and finally, they individually decide on the DCB measures to be taken.

The types of measures considered by the agents correspond to the three different types of solutions provided by this component:

- Solution type 0: These are solutions where agents consider only delaying at gate.
- Solution type 1: In these solutions agents decide whether level capping regulations will be applied.
- Solution type 2: In this case agents consider combinations of level capping and delay at gate measures.

3.6.1 Deep Q-Networks (DQN): Background knowledge

Tabular function representation in Reinforcement Learning (RL) has many successes [4] in relatively low dimensional problems, but it has two major drawbacks:

- The designer of the application has to hand-craft the state representations incorporating mostly important state features.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

- Methods learn/store each state or state-action value (V -value or Q-value, respectively) separately, resulting in unreasonably slow learning in large state-action spaces.

To resolve these problems, the deep Q-network [5] method successfully combined RL with neural networks (NNs). Deep Q-Network (DQN) uses a NN to approximate Q-values, modelling the agent's policy. The goal of a DQN agent is to select actions in a fashion that maximizes cumulative future reward. More formally, it uses a deep neural network to approximate the optimal action-value function

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots | s_t = s,\ a_t = a,\ \pi\right],$$

which is the maximum sum of rewards *r* discounted by *γ* at each timestep *t*, achievable by a behaviour policy *π*, after making an observation *s* and taking an action *a,* at any time step.

Reinforcement learning is known to be unstable or even to diverge when a nonlinear function approximator such as a neural network is used to represent the action-value function [6]. This instability has several causes: the correlations present in the sequence of observations, the fact that small updates to *Q* values may significantly change the policy and therefore change the data distribution, and the correlations between the action-values and the target values.

1. Observe environment state (*s*)
2. Choose random / policy action (*a*)
3. Receive resulting state (*s'*) and reward (*r*)
4. Store all transitions (*s*, *a*, *r*, *s'*) in Memory
5. Sample prioritized batch
6. Predict Q values from Policy and Target Networks
7. Calculate DQN Loss
8. Optimize in order to minimize Loss
9. Periodic Target Network updates

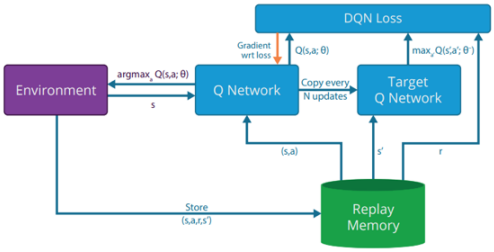


Figure 3. Deep Q Network (DQN) method

As shown in Figure 3, two vital elements of the success of DQN, that address these issues, are the addition of a target network and an experience replay memory. The target network mitigates the effect of constantly moving update targets, by incorporating a second network from which the update targets are sampled. This network is periodically updated with the weights of the original "online" network. The addition of uniform experience replay memory de-correlates the samples collected during rollouts, by randomizing over the data, thereby smoothing over changes in the data distribution.

During learning, the method applies Q-learning updates, on samples (or minibatches) of experience, drawn uniformly at random from the pool of samples stored. The Q-learning update at iteration *i* uses the following loss function:

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s') \sim \mathrm{U}(D)}\left[\left(r + \gamma \max_{a'} Q(s',a';\theta_i^-) - Q(s,a;\theta_i)\right)^2\right]$$

In which *γ* is the discount factor determining the agent's horizon, θ are the parameters of the Q-network at iteration *i* and θ^- the network parameters used to compute the target at iteration *i*.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

In our implementation of DQN we incorporated three established extensions of the method, in order to improve stabilization and enhance the quality of results.

The first extension is Double DQN [7]. Double Q Learning was originally introduced in [8] and aimed at improving the performance of Q learning, by eliminating the overestimation of action values, a phenomenon inherent to the method. It utilized two independent tabular approximations of the Q function during training, where each Q function is updated with [9] targets produced from the other Q function. Double DQN transferred this approach in the Deep RL setting, by using the online network to choose the best next action *a'*, but evaluate its Q value with the target network.

The second extension is the prioritized experience replay. In the original approach, experience transitions were uniformly sampled from a replay memory. However, this approach replayed transitions at the same frequency that they were originally experienced, regardless of their significance. Prioritized experience replay prioritizes experience sampling, so as to replay important transitions more frequently, and therefore learn more efficiently. In particular, the method proposes to more frequently replay transitions with high expected learning progress, as measured by the magnitude of their temporal-difference (TD) error.

Finally, we incorporate the exploration scheme of Adversarially-Guided Exploration, as proposed in [10, 11]. As reported in [12] and [11], training on adversarial examples enhances the resilience of the policy to perturbations crafted using the same technique. This phenomenon can be explained from the perspective of regularization: adversarial example perturbations of states provide the means for regularization of the policy (or value function) through data augmentation. Therefore, training the policy over adversarial examples of states generated with a certain attack mechanism results in the enhancement of resilience and robustness of the policy to perturbations crafted via that mechanism. The AGE mechanism extends the classical ε-greedy exploration mechanism by adjusting the probability of sampling actions for each state *s* according to the adversarial state-action significance, defined as follows:

$$\zeta_{adv}^{\pi_i}(s,a) = \frac{\exp\left(\max_{a'} Q^{\pi_i}(s,a') - Q^{\pi_i}(s,a)/\epsilon\right)}{\sum_{\alpha \in A} \exp\left(\max_{a'} Q^{\pi_i}(s,a') - Q^{\pi_i}(s,\alpha)/\epsilon\right)}$$

3.6.2 AI/ML Method for DCB using delay regulations (Type 0 Solutions)

3.6.2.1 Overall method

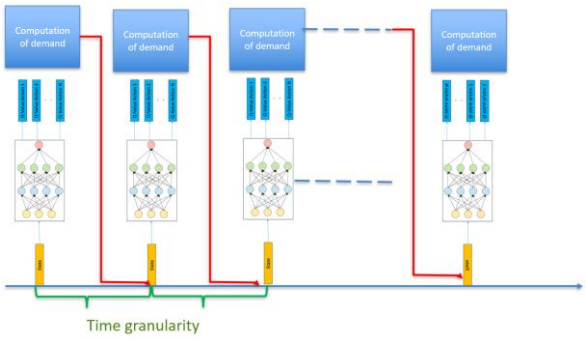


Figure 4. Simulation

EUROPEAN PARTNERSHIP
Co-funded by the European Union

The DQN model is trained by interacting with the environment, as is the case with all RL methods. For this interaction we have implemented a simulation of the environment, where agents (corresponding to flights) can make different decisions independently from the others and observe the resulting state: Thus the environment of an agent includes the airspace configuration and constraints, as well as other flights.

According to our methodology, we do not regulate a flight with a number of minutes at once (e.g. 14 min of delay), but by adding additional minutes of delay according to needs. Decisions are taken during an (simulation) episode, corresponding to a day where a specific scenario of 1439 minutes is simulated. For each agent, each episode comprises a series of rounds. Each round corresponds to a number of minutes of the day (mentioned as simulation timestep) and where each agent takes a concrete decision for additional delay from one up to that number of minutes (or none). I.e. in case an agent (flight) has 13 minutes of delay, then there should be a number of distinct time steps in the simulation where it has decided to take additional minutes of delay that sum up to 13. The method takes into account the existing daily traffic, given all the initial flight plans and decisions from individual flights.

3.6.2.2 Traffic demand monitoring

In order to calculate demand, we use the Hourly Entry Count with 60 minutes periods and 20 minutes period steps. Entries where the flights remain to the sector less than 60 seconds are ignored. Given the fact that we use 20 minutes of period steps, at any given time there are 3 active periods, so each sector entrance is calculated for those three periods. For example, an entry at 13:05 is considered for the periods starting at 12:20, 12:40 and 13:00. Given the demand and the capacities of each sector, we consider that if the demand exceeds 110% of the capacity, a hotspot occurs.

During the simulation, agents decide on delay regulations, thus altering their take-off times and subsequent entries to sectors they cross. Each timestep each agent makes a specific decision on the measure to be applied, the demand contribution of the specific flight is recalculated in order to acquire the new - up to date overall demand and the resulting hotspots, and so on. These recalculations occur at every timestep during simulation, following the delay decisions that agents take, as shown in Figure 4.

3.6.2.3 Addressing the scalability challenge: Centralized Training & Decentralized Execution

Beyond DRL instability issues, we had to resolve also scalability issues regarding the number of agents participating in a single scenario (typically, more than 6000). A technique which has many variants and is vital in understanding Deep MARL is called centralised training and decentralized execution (CTDE).

A naive way to provide agents with a joint policy is to train in parallel independent policies, one for each agent, following the independent learners paradigm. This approach can produce results in low-dimensional problems, but in real-world scenarios is inefficient and unstable. In order to alleviate these problems produced by the non-stationarity of a MAS environment, many algorithms employ some form of centralized training, thus exploiting information that is available during training but often unavailable during execution.

Parameter sharing [13] is the extreme case of CTDE: It learns a single policy shared by many agents. The main idea is that this single policy should be able to adequately describe the behaviour of different agents with the same goals. The resulting policy can be more robust, given the fact that it has been trained with samples that potentially belong to different parts of the state space, explored by different agents. We consider the parameter sharing approach to be the cornerstone of any scalable algorithm.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

Since scalability is a vital aspect of our use case, we opted for the parameter sharing version of DQN. At every timestep of our simulation, transition samples are collected from every agent and stored in the prioritized experience replay memory. These samples contain the state of the agent *s*, the action decided *a*, the resulting state s' and reward r (usually mentioned as s,a,r,s'). At the end of each simulation episode a minibatch of 200k samples is used to update the policy. During training, each agent has a probability to pick an action according to the AGE methodology (described in section 5.1.1), instead of a greedy one. This probability starts from 90% and diminishes as training continues, until it reaches 4%. During execution phase this probability is set to 10%, a common technique to ensure stability.

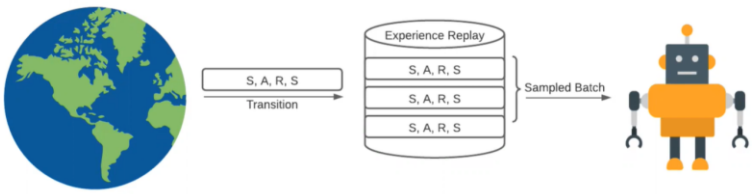


Figure 5. Gathering samples and providing minibatches

The reward function utilized has two distinct parts. If an agent participates in no hotspots it receives a positive scalar value minus the amount of delay minutes it has decided on. If the participates in one or more hotspots it receives a negative reward which depends on the total amount of minutes it spends within a hotspot.

3.6.2.4 Experimental Setting & Results

For the network architecture we use 4 fully connected layers, with relu activation, consisting of 512 nodes each. Each training batch consists of 200000 samples. Discount factor gamma is set to 0.99.

The replay memory has a capacity of 3000000 samples.

At the start of exploration epsilon is set to 0.9 and is diminished by 0.01 every 15 episodes, until it reaches the minimum of 0.04. The total number of episodes is 3150. During testing epsilon is set to 0.1.

Below we present indicative results from scenarios on specific days with heavy traffic: 01/08/2019 and 02/08/2019. In the following table we present statistics regarding the traffic of each scenario. Flights column indicates the total amount of flights that cross the Spanish FIR during the specific day. Hotspots column indicates the total number of occurring Hospots. Flights in Hotspots column indicates the number of flights that participate in at least one Hotspot.

Scenario	Flights	Hotspots (baseline)	Flights in Hotspots (baseline)
01/08/2019	6682	97	2122
02/08/2019	6625	59	1298

Table 2. TAPAS ATFCM Indicative Scenarios

EUROPEAN PARTNERSHIP

Co-funded by the European Union

In the following figure we can see the traffic in a specific sector during the 02/08/2019 day. With blue we indicate the demand and with orange the capacity. Traffic is severe enough to produce 21 consecutive hotspots.

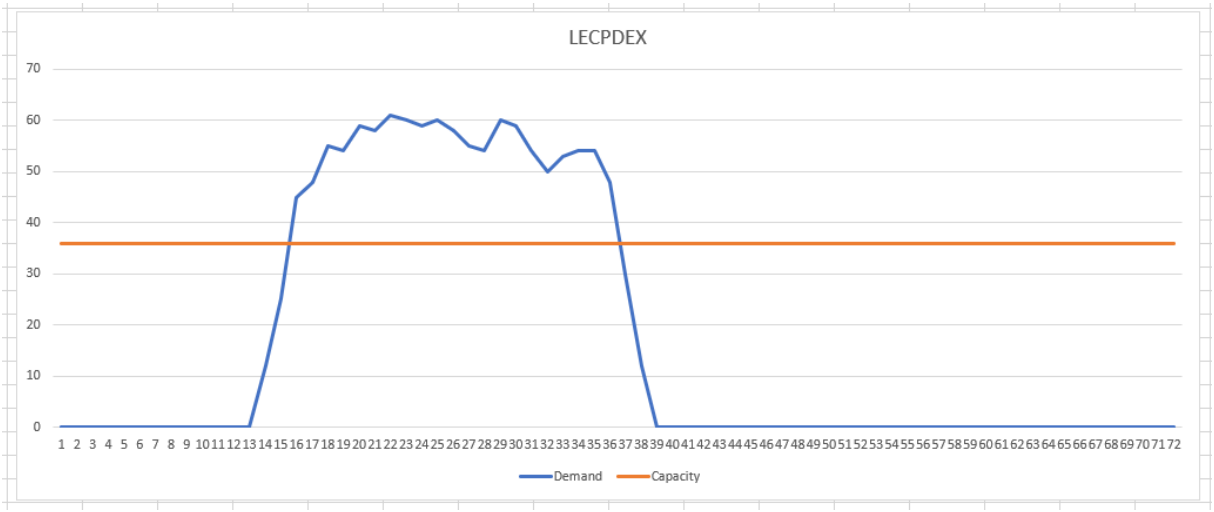


Figure 6. Traffic in LECPDEX in 02/08/2019

In the following table we present statistics regarding the results of each scenario. Final hotspots column indicates the remaining hotpots after applying the proposed measures. Average delay column is the total minutes of delay (ignoring delays less than 5 minutes), divided by the number of all flights. Delayed Flights column indicates the number of flights which got regulated for ≥5 minutes. The last column is the total minutes of delay (ignoring delays < 5 minutes), divided by the number of flights which got regulated for ≥5 minutes.

It must be noted that only delays equal or greater than 5 minutes are considered, given (a) expert advice that delays less than 5 minutes are usually not considered as delays, as they can be mitigated at operational time, (b) that the average delay reported will be reduced a lot given all flights with <5 minutes of delay (e.g. with 1 or 2 minutes of delay). This effect can also be seen by comparing the results reported in “Average delay” and “Average delay ≥5” columns in Table 3. Thus, results in the “Average delay ≥5” column reflect more accurately the capabilities of the AI/ML method, while being closer to the way delays are currently considered in the operational setting.

Scenario	Final Hotspots	Average Delay	Delayed Flights (≥5)	Average Delay ≥5
01/08/2019	10	20.29	2654	51.46
02/08/2019	10	8.17	1278	42.37

Table 3. TAPAS ATFCM Indicative Results

EUROPEAN PARTNERSHIP

Co-funded by the European Union

In the following figure we can see the Learning Curve of the 02/08/2019 experiment. In blue we indicate the remaining hotspots after a simulation cycle and in orange the Average Delay, as explained in the previous paragraph.

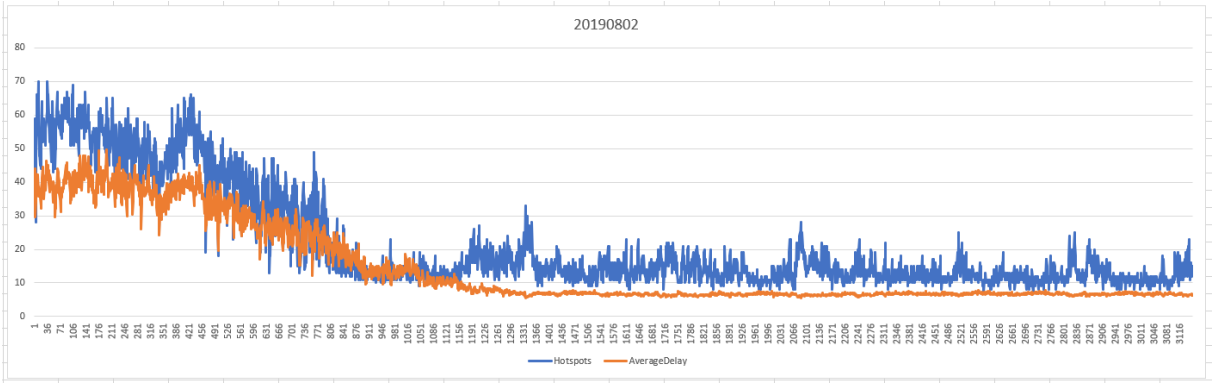


Figure 7. 02/08/2019 Learning Curve

In the indicative results we can see that the severity of the traffic renders solutions with a minimal amount of regulations infeasible. Our method manages to solve the vast majority of the hotspots, but a number of them still persists, after the proposed regulations. As expected, we can see the difference between scenarios, for example in Table 3, 3rd column, where the average delay needed to achieve the same amount of remaining hotspots is much smaller for the scenario with a smaller number of baseline hotspots.

3.6.3 AI/ML Method for DCB using level capping (Type 1 Solutions)

3.6.3.1 Overall method

Our methodology here starts with the production of alternative flight plans, produced by applying level capping measures to the original flight plans, where this is possible. Flights eligible for this specific kind of regulation are the ones which take off inside the Spanish FIR, or in nearby airports. For each eligible flight, all possible level capped flight plans are created and ranked according to the smoothness of the resulting planned trajectory.

Given all these flight plans for each flight, we retain the top-k ones. After calculating the baseline demand and hotspots of the scenario, we cross check which flights participate in hotspots that can be resolved by a level capping regulation. In practice, the majority of flights falling into this category can resolve exactly one hotspot, so the best ranking flight plan which can resolve this hotspot is selected. In the case that more than one hotspots can be resolved, the best ranking flight plan, which can resolve the most hotspots is selected (typically resolving all hotspots).

After this data preparation, the alternative flight plans, as well as all the initial ones are exploited vy the DQN algorithm: Each agent decides on its own plan.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

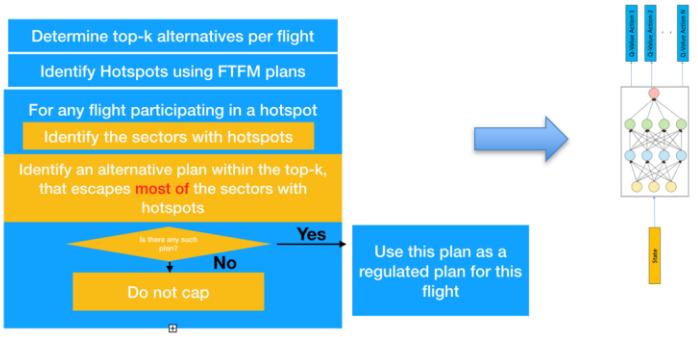


Figure 8. Selecting alternative flight plans due to level capping measures.

3.6.3.2 Traffic demand monitoring

Traffic demand monitoring is identical with the one described in section 5. 2.2. The only difference here is that there are only two states of demand, the initial (no measure has been applied) and the final (some flights have a level capping measure). As explained in the next section there are not intermediate timesteps (i.e. simulation steps, as in deciding delays) in deciding level capping measures.

3.6.3.3 Training & Decision making

Similarly to the approach described in section 5.2.1, we have designed a simulation in order to provide the necessary interaction between DQN and the environment. The main difference here is that the simulation has two steps. The initial one, where each agent observes the hotspots they participate in (if there are any) and decides whether to utilize the alternative level capped flight plan. The second and final step is the one where all decisions are evaluated, by our rewards function, and feedback is given to the DQN algorithm in the form of samples in the replay memory.

The reward function utilized has two distinct parts. If an agent participates in no hotspots it receives a positive scalar value. If the participates in one or more hotspots it receives a negative reward which depends on the number of hotspots it participates in.

3.6.3.4 Experimental Setting & Results

For the network architecture we use 2 fully connected layers, with relu activation, consisting of 100 nodes each. Each training batch consists of 32 samples. Discount factor gamma is set to 0.99.

The replay memory has a capacity of 3000 samples.

At the start of exploration epsilon is set to 0.9 and is diminished by 0.01 every 20 episodes, until it reaches the minimum of 0. The total number of episodes is 2000. During testing epsilon is set to 0.

Below we provide indicative results for the scenarios 01/08/2019 and 02/08/2019. In the following table we present statistics regarding the traffic of each scenario. Flights column indicates the total amount of flights that cross the Spanish FIR during the specific day. Hotspots column indicates the total number of occurring Hotspots. Flights in Hotspots column indicates the number of flights that participate in at least one Hotspot. Flights were level capping is allowed column indicates the total number of flights taking off inside or nearby the Spanish FIR, thus being eligible for level capping regulations. The last column indicates the number of flights that are eligible for level capping and have a level capped alternative flight plan which avoids a sector in which they participate in a Hotspot.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Scenario	Flights	Hotspots (baseline)	Flights in Hotspots (baseline)	Flights were level caping is allowed	Flights that are eligible to level capping and that participate in at least one hotspot
01/08/2019	6682	97	2122	1201	71
02/08/2019	6625	59	1298	1196	49

Table 4. TAPAS ATFCM Indicative Scenarios (with level capping measures)

In the following table we present statistics regarding the results of each scenario. Final hotspots column indicates the remaining hotpots after applying the proposed regulations. Regulated Flights column indicates the number of flights which received a level capping regulation.

Scenario	Final Hotspots	Regulated Flights
01/08/2019	92	39
02/08/2019	56	12

Table 5. TAPAS ATFCM Indicative Results (with level capping measures)

In the following figure we can see the Learning Curve of the 02/08/2019 experiment. In blue we indicate the remaining hotspots after a simulation cycle and in orange the Regulated Flights, as explained in the previous paragraph.

Figure 9. 02/08/2019 Learning Curve (with level capping measures)

For at least these two scenarios we present here, the impact of level capping regulations is limited. This is due to the fact that most traffic and thus consecutive hotspots occur in sectors which reach low altitudes. This renders level capping approaches unable to resolve high peaks of traffic. On the other hand, we can see that the level capping approach can resolve the eligible hotspots by applying a minimal amount of regulations.

3.6.4 AI/ML Method for DCB using level capping and delay regulations (Type 2 Solutions)

EUROPEAN PARTNERSHIP

Co-funded by the European Union

3.6.4.1 Overall method

The approach here is identical to the one described in section 5.2. The only difference is the starting air traffic situation, since the flight plans decided by the previous methodology are here considered the starting point. Specifically, after calculating all alternative flight plans due to level capping measures the DQN method decides to which flights such a measure is applied. The result is a new set of flight plans (where some of the flights have a level capping measure applied) and typically a state where some hotspots have been resolved. This new set of flight plans results in a relaxed problem compared with the initial one, which is solved according to the method described in Section 5.2.

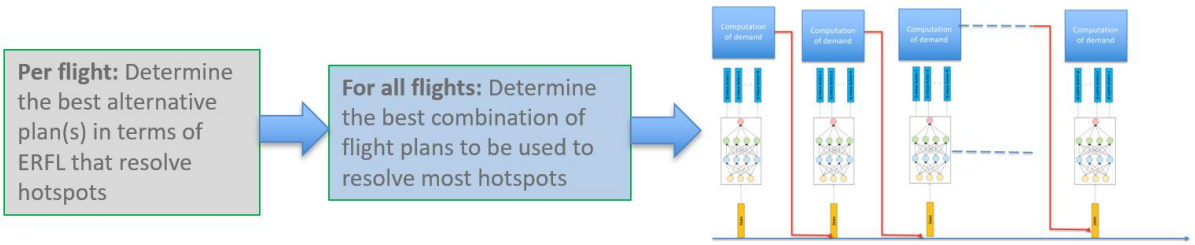


Figure 10. Combining level capping measures and ground delays.

3.6.4.2 Experimental Results

Below we provide indicative results for his type of solutions for the scenarios 01/08/2019 and 02/08/2019. In the following table we present statistics regarding the traffic of each scenario. Flights column indicates the total amount of flights that cross the Spanish FIR during the specific day. Hotspots column indicates the total number of occurring Hotspots. Flights in Hotspots column indicates the number of flights that participate in at least one Hotspot. It should be noted that the baseline for this solution is not the original problem, but the resulting one after applying level capping regulations.

Scenario	Flights	Hotspots (baseline)	Flights in Hotspots (baseline)
01/08/2019	6682	92	1981
02/08/2019	6625	56	1210

Table 6. TAPAS ATFCM Indicative Scenarios (after applying level capping measures)

In the following table we present statistics regarding the results of each scenario. Final hotspots column indicates the remaining hotpots after applying the proposed regulations. Average delay column is the total minutes of delay (ignoring delays less than 5 minutes), divided by the number of all flights. Delayed Flights column indicates the number of flights which got regulated for more than 4 minutes. The last column is the total minutes of delay (ignoring delays less than 5 minutes), divided by the number of flights which got regulated for more than 4 minutes.

Scenario	Final Hotspots	Average Delay	Delayed Flights	Average Delay >4
01/08/2019	7	18.37	2686	45.70
02/08/2019	10	8.50	1192	47.28

Table 7. TAPAS ATFCM Indicative Results (with delay and level capping measures)

EUROPEAN PARTNERSHIP
Co-funded by the European Union

In the following figure we can see the Learning Curve of the 02/08/2019 experiment. In blue we indicate the remaining hotspots after a simulation cycle and in orange the Average Delay, as explained in the previous paragraph.

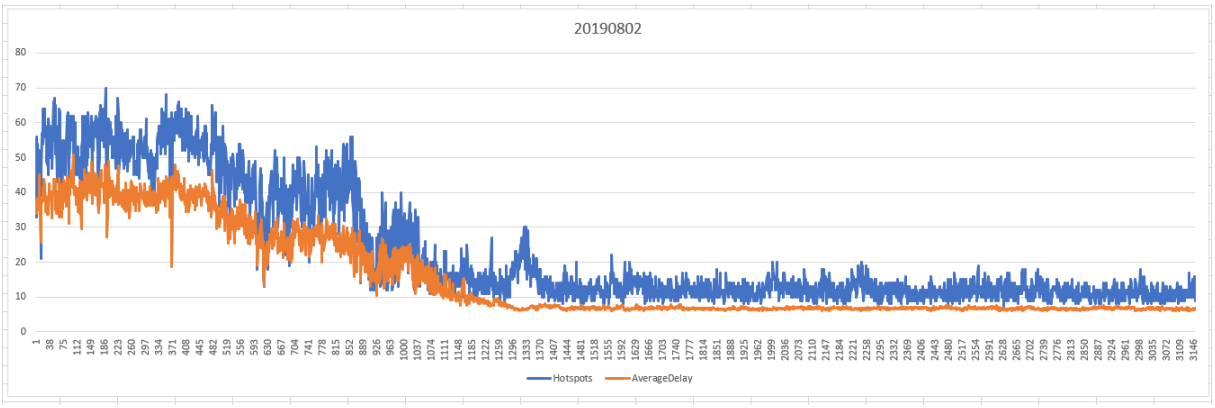


Figure 11. 02/08/2019 Learning Curve (with delay and level capping measures)

As we can see in the previous tables, the combination of level capping and delay regulations can produce better overall results. For example, in the case of scenario 01/08/2019 the remaining hotspots as well as the average delay are less. On the other hand, in the case of 02/08/2019 we can see that the effect of level capping regulations can be minimal and thus not having a great impact in the overall proposed solution.

3.7 Explainability (XAI) component

3.7.1 Overall XAI method: Mimicking DQNs

DRL models, are inherently hard to interpret. There should be a method that extracts explanations (i.e. interpret these methods) in either a local or in a global way: By “local” we mean providing explanations for decisions taken under specific circumstances, while by providing “global” explanations we consider providing information of the entire logic (means and/or ends) towards achieving goals. Interpreting DRL is an area still in its infancy, and can be addressed either by training interpretable models, or by replacing DRL methods with interpretable models. Here we follow the first approach, towards providing local interpretations.

Interpreted models using the DRL method as an oracle, can be trained either during or after training the DRL models. In any case the training process may use samples provided by the deep method, advising also the models of the DRL method.

This results into two distinct paradigms that refine the transparent box paradigm: The mimicking and the distillation paradigms.

It must be noted that although the distillation process has been introduced as a neural network model compression process through training a student neural network, and has been introduced as a term in [14], the first work that introduces it for sequential decision making (as a model compression technique, not as a method for building interpretable models) is described in reference [15] where the teacher provides samples, each comprising an observation sequence and a vector of state-action Q-values.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

For DRL methods, the distinguishing line between the distillation and the mimicking processes is not that clear in the literature. The main difference between the two paradigms concerns the input provided to the interpretation process: While the distillation process distils the knowledge acquired by the trained DRL agent by exploiting any of the constituent DRL models, the mimicking process, monitors the interaction of the reinforcement learning agent with the environment, and gathers interaction samples, recording agent decisions, state transitions and rewards that the agent got.

In this work we follow a mimicking approach, which is shown in Figure 8, in its generic form: Overall, given a well-trained DRL model, this is treated as an oracle towards training a mimic model, which learns to take decisions in a way that is shown to be faithful to the DRL decisions.

Specifically, as it is shown in Figure 8, given an observation signal I, assuming that the full state may not be observable by the agent, the DRL model predicts the values of the actions a in that state, i.e. $\hat{Q}(I, a)$. Given the best action predicted, i.e. argmax$\hat{Q}(I, a)$, together with the observation signal, and optionally, the values of all actions (if actions are discrete), the mimic models is trained to take the DRL predicted decision.

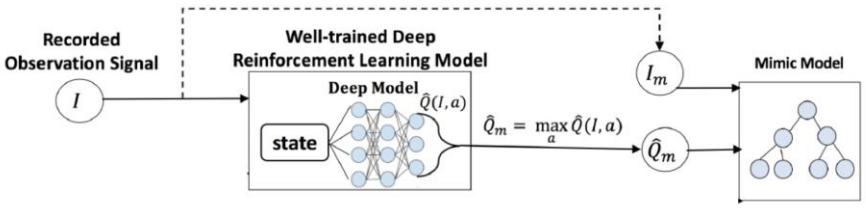


Figure 12. The overall DRL mimicking process (this figure has been inspired from the one provided in [16])

In our approach in TAPAS, the DRL model trained is provided by DQN, while the mimic model is provided by training Stochastic Gradient Trees.

3.7.2 Stochastic Gradient Trees (SGTs) Background knowledge

Stochastic Gradient Trees (SGTs) [17] is a state-of-the-art method for learning decision trees using stochastic gradient information as the source of supervision. SGTs operate in the incremental learning setting, although in our case we are using it in a batch learning setting (i.e. providing all samples at the same time), towards mimicking a well-trained DQN model.

SGTs do not require the construction of a new tree for every update, aiming to introduce splits that minimize the expected loss. SGTs is a generic method and has been demonstrated in several types of problems by changing only the loss function: classification, regression, and multi-instance learning as example applications. As our interest is to mimic the trained DQN model, we aim to perform regression on the $\hat{Q}$ function, as it has been learned by the deep model.

Our choice of SGTs is motivated and justified by the fact that it outperforms state of the art incremental regression trees.

Overall, as described in [17] , in supervised incremental learning, data is of the form $(x_t, y_t) \in X \times Y$. A new pair arrives at every time step, t, and the aim is to predict the value of y_t given x_t. Algorithms for

EUROPEAN PARTNERSHIP
Co-funded by the European Union

this setting enable prediction at any time step—they cannot wait until all instances have arrived and then train a model.

In our setting, as also discussed in the mimicking paradigm above, x_t is a (state, action) pair and y_t is the $\hat{Q}$-value for that pair, as predicted by the deep model. These samples are tagged with the simulation time step and provided from each of the agents at each time step during simulation.

Below, we describe how SGTs incrementally construct decision trees that can be trained to optimise arbitrary twice-differentiable loss functions. The major ideas behind SGTs are (a) evaluating splits and computing leaf node predictions using only gradient information, and (b) using standard one-sample t-tests to determine whether enough evidence has been observed to justify splitting a node.

Solving a regression problem, we aim to predict $\hat{Q}$-values. Usually in such settings, a single SGT is used to generate predictions, but in our setting we use multiple SGTs, one for each of the possible actions: The aim to predict the Q-value for each potential action, so as (a) to learn how to rank alternatives, (b) to be able to explain the main reasons behind each ranking (i.e. provide the features that contributed to that decision) and (c) be able to mimic faithfully (with fidelity) the decisions of the deep model, by selecting the prediction that maximizes the predicted Q-value, and thus, by selecting the action which corresponds to that tree and which maximizes agent's expected reward.

We train SGTs using a squared error loss function

$$l(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

measuring how well our predictions, $\hat{y}$, match the predictions y provided by the deep model.

Predictions are generated using an SGT (realizing function Q_{SGT}), optionally composed with an activation function, σ,

$$\hat{y} = \sigma(Q_{SGT}(\boldsymbol{x}))$$

The objective during training is to minimize the expected loss, as estimated from the data observed between the current time step, t, and time step, r, at which the tree was previously updated. Assuming i.i.d. data, the expectation can be stochastically approximated using the most recent observations,

$$\mathbb{E}[l(y, \hat{y}))] = \frac{1}{t - r} \sum_{i=r+1}^{t} l\,(y_i,\, \hat{y}_i)^2$$

Towards this objective at each time step, we aim to find a modification, to the tree that takes a step towards minimising the expected loss. Such a modification in a decision tree, is a split to one of its leaf nodes, or an update to the prediction made by a leaf.

Therefore, for each possible split we aim to find the one that yields the maximum reduction in loss. Thus, given $\boldsymbol{x_t}$ and the leaf node where this «lands», the methods considers not splitting and updating the prediction made by that leaf node, or splitting on each attribute (state feature in our case), while deciding on the values to be assigned to any newly created leaf nodes.

The empirical expectation of the loss function can be approximated using a Taylor expansion around the unmodified tree at time t (indicated by $Q_{SGT\,t}$):

EUROPEAN PARTNERSHIP

Co-funded by the European Union

$$\mathcal{L}_t(u) \approx \sum_{i=r+1}^{t}\left[l(y_i,\ Q_{SGT}(\boldsymbol{x_i})) + g_i u(\boldsymbol{x_i}) + \frac{1}{2}\, h_i u^2(\boldsymbol{x_i})\right],$$

Where, u denotes the potential split given a new instance, g the first derivative of the loss function w.r.t. $Q_{SGT}(\boldsymbol{x_i})$, i.e. in our case $\hat{y} - y$, and h the second derivative, i.e. 1.

Optimization can be expressed by eliminating the constant first term resulting in

$$\Delta\mathcal{L}_t(u) \approx \sum_{i=r+1}^{t}\left[g_i u(\boldsymbol{x_i}) + \frac{1}{2}\, h_i u^2(\boldsymbol{x_i})\right] = \sum_{i=r+1}^{t}\Delta l_i(u)$$

Describing the change in loss due to a potential split u.

The optimal splitting, given all possible splits is

$$v_u^*(j) = -\frac{\sum_{i\in I_u^j} g_i}{l+\sum_{i\in I_u^j} h_i},$$

Where I_u^j is the set of indices of instances reaching the new leaf node identified by *j*, and *v* maps these new leaf node identifiers to the difference between their predictions and the prediction made by their parent.
SGTs use student's t-test to determine whether a split should be made, with the hypothesis that no split should be made. This avoids knowing in advance the range of values that can be taken for *n* terms $\Delta l_i(u)$.

The process used to determine whether enough evidence has been collected to justify a split is prohibitively expensive to carry out every time a new instance arrives. In practice, SGTs check whether enough evidence exists to perform a split when the number of instances that have fallen into a leaf node is a multiple of some user specified parameter. As in [17] this value is set to 200 by default.

3.7.3 XAI method per type of solution

For solutions of type 0 and 2, i.e. involving ground delay regulations, we follow an approach which is similar to training the DRL process: Centralized training and decentralized execution.

Specifically, the mimic model is trained to predict the Q values of all delay options (i.e. adding 0 to 10 minutes of delay) at each simulation time step and for any of the agents. Aiming to take decisions that are faithful to those taken by the DQN model (i.e. minimizing the loss, as described above), the mimic model must predict the best action, i.e given a state *s*, it must hold that

$$a^* = \text{argmax}\hat{Q}(s,a) = \text{argmax}Q_{SGT}(s,a) = \ \text{argmax}_a Q_a(s)$$

EUROPEAN PARTNERSHIP

Co-funded by the European Union

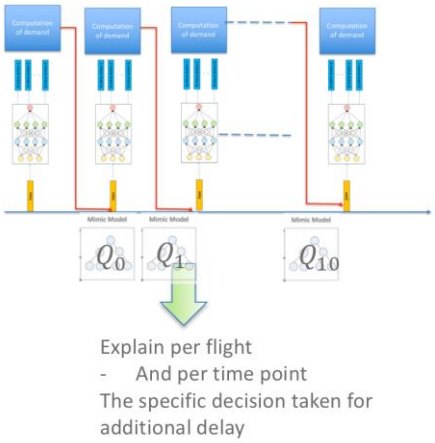


Figure 13. The mimicking process for explaining delay decisions in solutions of type 0 and 1

As noted before, in our setting we use multiple SGTs, one for each of the possible actions: These models are trained in a centralized manner to solve a regression problem for predicting Q_{SGT} per action (denoted as $Q_i, i = 0 \dots 10$), given any state *s*. Then, during execution (i.e. providing local explanations) they provide explanations per agent and simulation timestep, explaining the DRL decision of 0…10 additional minutes of delay at simulation time step *t*. The overall approach is shown in Figure 9.

For solutions involving level capping measures, i.e. solutions of type 1 and 2, we aim to explain (a) why a level capping measure *may* be applied: This provides information on the qualities of the alternative flight plans considered, mostly in terms of sectors being crossed instead of others towards avoiding hotspots; (b) the decision of each agent to choose a flight plan with level capping, instead of the original flight plan, or not.

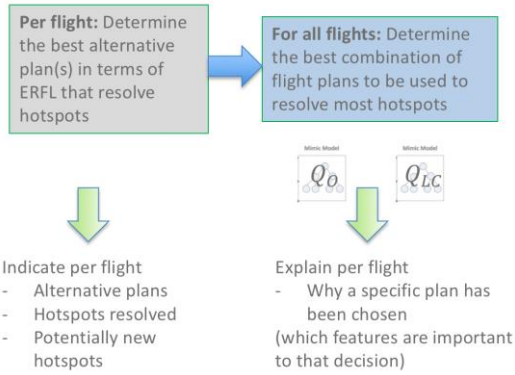


Figure 14. The mimicking process for explaining delay decisions in solutions of type 0 and 1

As in explaining delay regulations, the mimic model is trained to predict the Q values of all options (i.e. applying a level capping measure or not) and for any of the agents. Aiming to take decisions that are faithful to those taken by the DQN model (i.e. minimizing the loss, as described above), the mimic model must predict the best option.

As done previously, in our setting we use multiple SGTs, one for each of the possible actions: These models are trained in a centralized manner (i.e. given samples from all agents) to solve a regression problem for predicting Q_{SGT} per option (denoted as $Q_i, i = O, LC$), where O denotes original plan, while LC denotes one with level capping measures, given any state *s*. Then, during execution (i.e. providing local explanations) these models provide explanations per agent, explaining the DRL decision of taking the appropriate measure. The overall approach is shown in Figure 10.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

3.7.4 Provision of explanations' content

Each agent decision at a simulation timestep corresponds to a number of explanations: one for each possible option. In case of delays only, following the agent-based simulation we provide per agent and per simulation timestep, explanations regarding agents' decision on additional minutes of delay. In case of level capping regulations, we provide explanations only for agents' decisions on the flight plan chosen (thus, this does not indicate any time step).

In any case, we provide explanations regarding:

- Why the agent has taken a specific decision.
- What the agent considers as important for taking an alternative decision (this is contrastive explanation given the above, showing what the agent considers as counter arguments for its actual decision).

In case of delays only, each agent can take a decision among 11 different options, corresponding to 0-10 minutes of additional delay at each timestep. In case of level capping measures the agent can take a decision among different flight plans' options. In case of level capping measures, we use 2 trees corresponding to decision for **not** changing the original flight plan and be regulated by means of level capping. In case of applying combinations of delays with level capping, each agent, having chosen whether it will take a level capping regulation, it can decide among 11 different options for additional delay at any time step.

Counterarguments for a decision are provided by the tree corresponding to an action which is different than the specific measure decided. Specifically, choosing at a time step to add d>0 minutes of delay, or choosing a level capping measure, the counterargument is provided by the action for not applying any measure, i.e. 0 for delays and *O* for choosing the original flight plan instead of applying a level capping measure.

As presented in D4.1 [18], given a state *s,* explanations provided are constructed by following any node in a path from the root of a decision tree to the leaf corresponding to that state, depending on the splitting criteria learnt and the values of state features. The form in which nodes are provided in an explanation are described in D4.1 [18].
The way explanation content is rendered and provided in a surface representation via visual means is described in D4.3 [19].

3.7.5 Experimental Results

In this section we provide experimental results regarding (a) the complexity of the explanations constructed in terms of decision trees depths, (b) the learning curves in terms of error reduction per tree and learning epoch, and (c) the fidelity of the mimic models in terms of predicting the action decided by the DRL model.

These results are provided per training epoch, where an epoch involves the provision of all training samples (i.e. regarding each simulation time step and each agent) in SGTs.

- Data on trees constructed: Figure 15 provides the depth of the trees constructed per training epoch. As it can be seen, SGTs manage to keep the trees as shallow as possible, thus, close to

EUROPEAN PARTNERSHIP
Co-funded by the European Union

our interests, reducing the complexity of the explanations. This is an inherent feature of SGTs, given that the objective while splitting a node involves not only reducing the expected loss, but also the complexity of the representation. Notable examples are trees for Q_5 and Q_{10} which expand in depth greater than 30 in epochs greater than 6. While the tree for Q_5 in epoch 6 has reached nearly its max depth, Q_{10} continue expanding even in epoch 9, showing that there are ways to further reduce loss via splitting nodes. This will be discussed further after viewing loss and accuracy results.

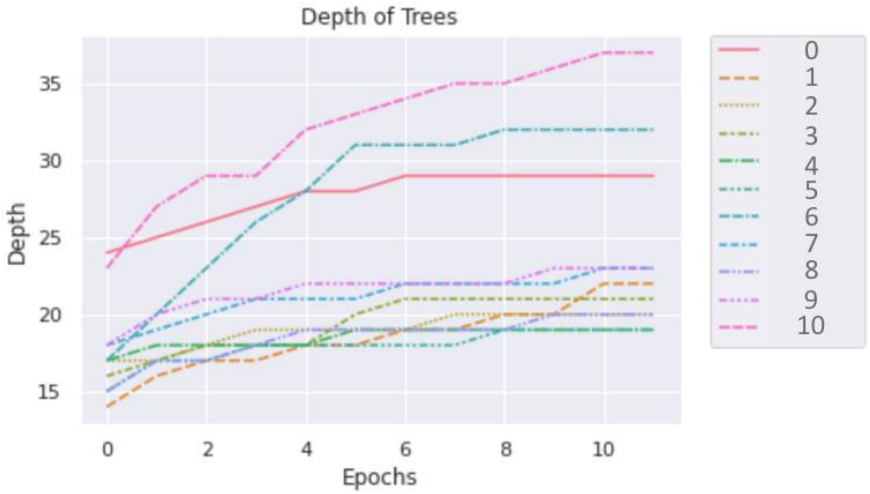


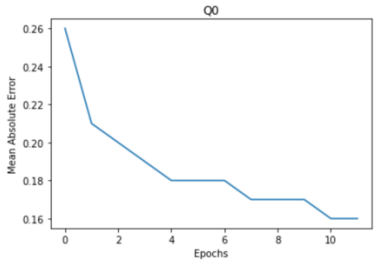
Figure 15. Depth of trees learnt by SGTs

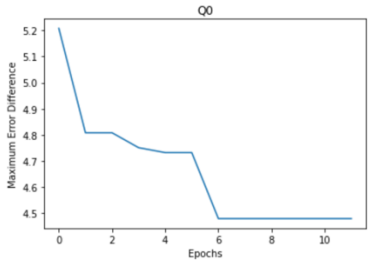
- Loss function diagrams: Figure 16(a)-(k) show how loss evolves per tree and epoch. Given a state *s*, loss is measured here in terms of the Mean Absolute Error (MAE) and the Maximum difference (error) (MER) between Q_i as predicted by the STGs and $\hat{Q}_i$ as predicted by DQN.

  As these diagrams report, MAE and MED are very low in all cases. The greatest MAE and MED reported is in predicting Q_{10}, which seems learnt models, also in accordance to the evolution of the corresponding tree depth, to have further room to reduce the expected loss either by expanding the tree (by splitting leaf nodes) or by improving predictions in the existing leaf nodes. However, the differences between the MAE / MED reported in the 1st epoch, and the MAE/MER reported in the 10th epoch are very small.

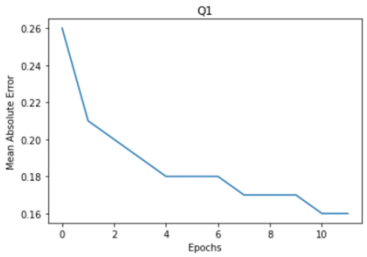
  An important observation concerns the evolution of the MAE in terms of evolution of tree depth: While most of the trees stop to expand after epoch 6, most of them continue to improve the reported error. This further provides evidence on SGTs inherent ability to keep trees' height as small as possible while improving predictions.

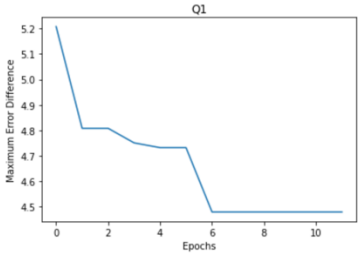
EUROPEAN PARTNERSHIP
Co-funded by the European Union



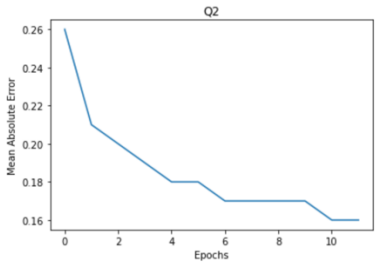


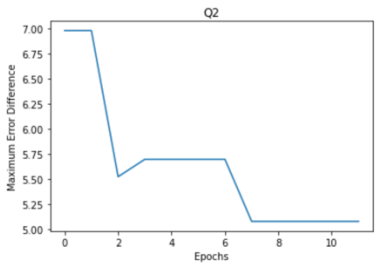
a



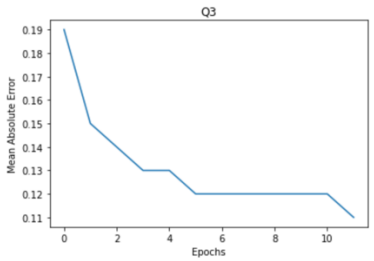


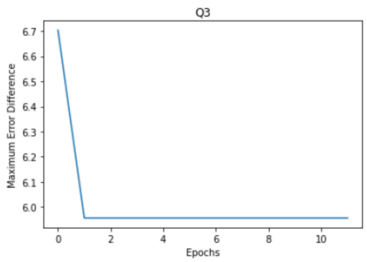
b





c

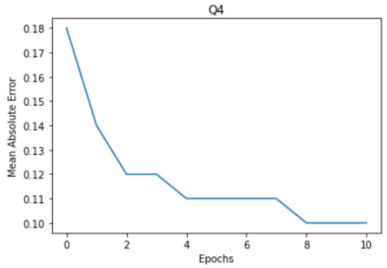


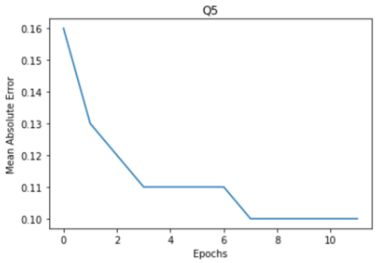


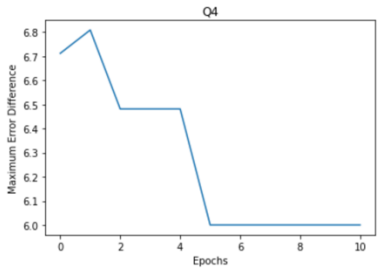
d

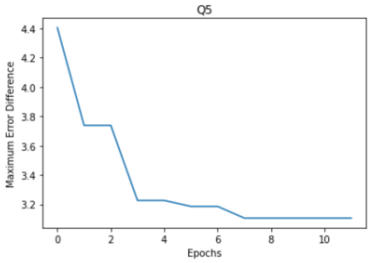
EUROPEAN PARTNERSHIP

Co-funded by the European Union



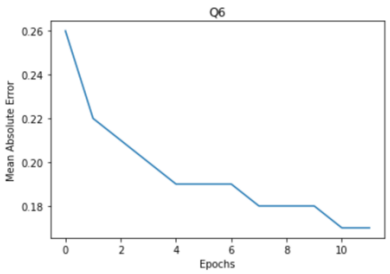


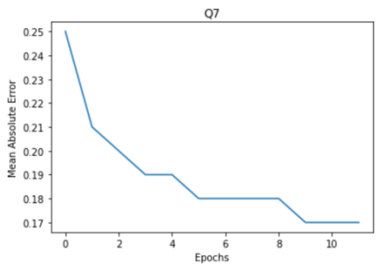


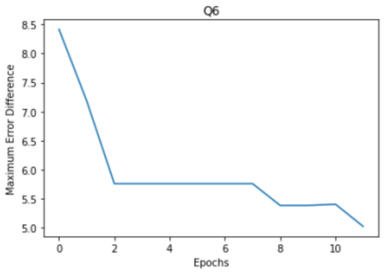


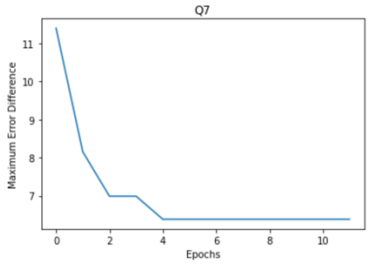
e

f









g

h

EUROPEAN PARTNERSHIP

Co-funded by the European Union

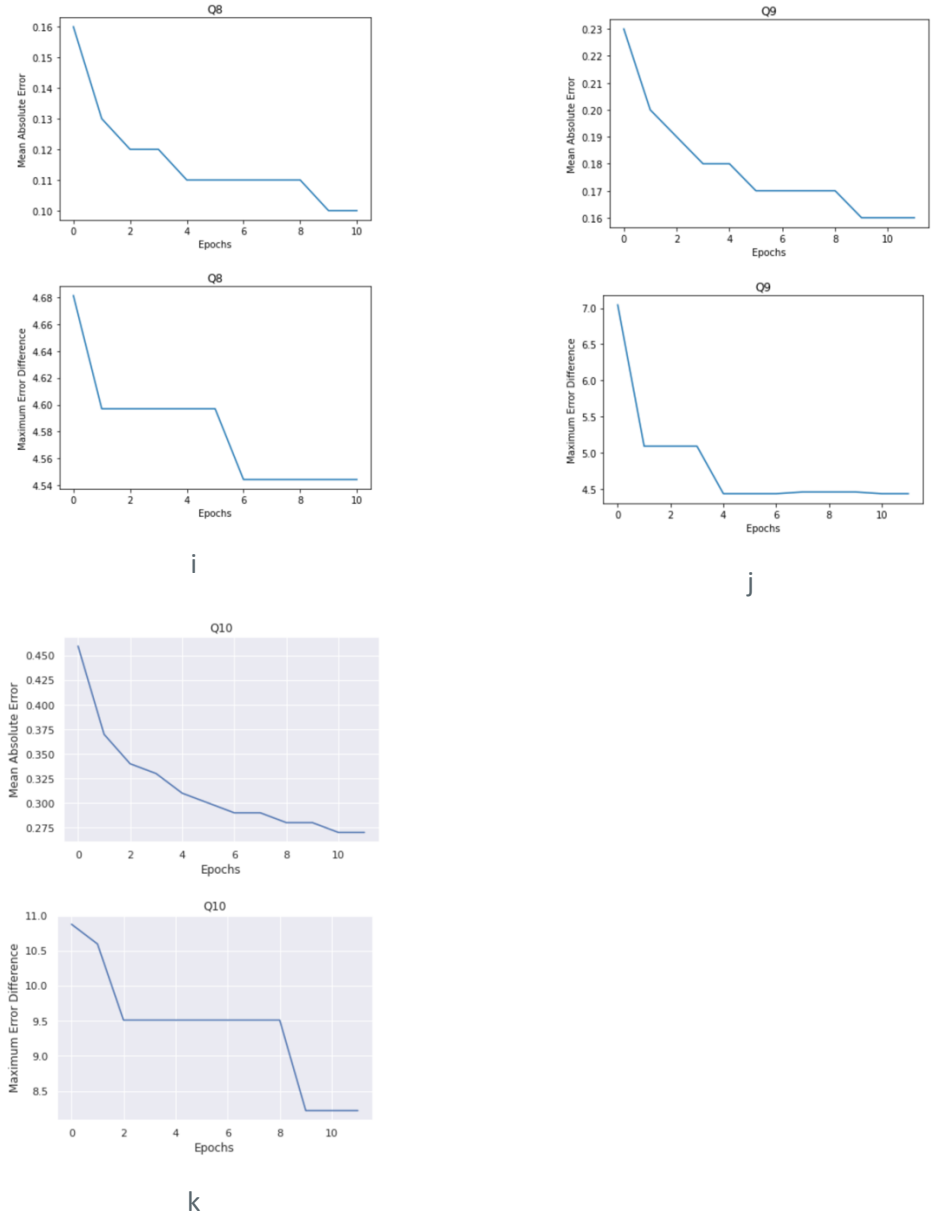


Figure 16. Error in SGTs prediction per tree and epoch

- Fidelity (accuracy of predictions): Fidelity of the mimic model concerns how well it mimics the original model. In order words, it measures whether the decision taken using the mimic model match those of the original model. This is measured by the accuracy of the predictions, considering that it must hold that $\text{argmax}\hat{Q}(s,i) = \text{argmax}Q_i(s),\ i = 0,..10$.

As Figure 17 shows, after the 1st epoch, the accuracy is very high. This is due to the incremental learning nature of SGTs. Thus, a single pass of the samples is enough to reach high accuracy. Additional training epochs do not increase the accuracy further (neither reduce it, given the scale in y axis).

Now, it must be noted that even when MAE/MED is somehow high in the first epoch, the mimic model is able to predict accurately the actions decided by the DRL module. This is very

EUROPEAN PARTNERSHIP
Co-funded by the European Union

important given that in these cases the depth of the trees is not greater that 24, and thus the complexity of explanations is low.

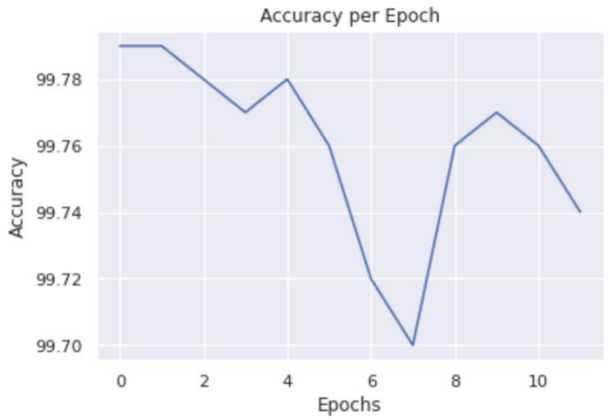


Figure 17. Fidelity: Accuracy of SGTs predictions, w.r.t. the predictions of the DRL.

As a conclusion, SGTs manage to keep the complexity of the explanations constructed in terms of decision trees depths at low levels. In fact, they manage to fit accurate models (decision trees) that have not a very large height (less than 30 in most cases) even in early learning epochs, although they can have a very large breadth. This means that explanations provided (comprising arguments constructed from decision nodes in paths from roots to leaves of SGTs) have not a very high complexity. Also, the fidelity of the mimic SGT models in terms of predicting the action decided by the DRL model is rather high: These aspects make SGTs a good fit to explain agents' decisions.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

4 CD&R Use Case

4.1 CD&R Use Case Specification

The scope of this use case concerns the conflict detection and resolution (CD&R) process in the executive controller timeframe, as part of the tactical separation assurance process.

The primary actor is the executive controller and the supporting actors are the executive controllers of the upstream and downstream sector.

It is assumed that the conflict detection and resolution during the planner timeframe for action has been performed. Separation requirements are considered to be independent from the region being analysed (area of responsibility (AoR)) and stable. Flights are able to traverse the AoR and are handed over to the downstream sector free of conflicts, w.r.t. specific exit/entry sector conditions.

The objective is to perform conflict detection and resolution during the tactical separation assurance process, identify and resolve all the potential tactical conflicts, providing transparency to Air Traffic Controllers (ATCo) of the AoR.

4.2 CD&R Problem Specification

The CD&R task involves a number of flights in a spatial area of responsibility (AoR). The task is about detecting at any time point *t* the conflicts that may occur between any pair of flights, and for any such conflict, decide whether and what resolution actions should be applied to any, or both, of the conflicting flights.

Casting this problem into a multiagent problem, we consider that each agent *i* represents one of the *N* flights co-occuring at any time point *t* in the AoR, or in any potential downstream sector. We consider the set of Relevant AoRs (RAoRs) as the union of AoR with any potential downstream sector.

Given the trajectory T_i of agent *i* within the RAoRs, we define the set of *neighbouring agents* to be the set of conflicting trajectories to T_i in RAoRs at a specific time point *t*. I.e., the set of agents interacting with *i* at time point *t*. These are denoted *Neigh*(*i,AoR,t*).

Agent *i* has to react and resolve all conflicts with *Neigh*(*i,AoR,t*), deciding whether it will apply any resolution action at *t*+1, and what this action should be.

Specifically, we consider the following potential actions:

(a) *Flight Level change*, where the agent changes its current flight level with vertical speed +17 or -17 feet/s for ascending/descending course;

(b) *Course change*, where the available changes of agent's course are 10, -10, 20, -20 degrees;

(c) *Horizontal speed change*, where the available changes of agent's horizontal speed are -3.6008 or 3.6008 m/s;

(d) *Direct to waypoint*, where the agent can choose one of the next flight plan waypoints; and

(e) *No action*, where the agent continues its current course without any change.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

The problem is formulated as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP), where at each timestep *t* each agent *i* receives a local observation o^t_i, takes an action a^t_i, and gets an individual reward r^t_i. The objective is to maximize the sum of all agents' expected returns.

A local observation of an agent is a vector comprising the following features:

- Nalt = alt/max_{alt}, where *alt* is the agent's current altitude in feet and max_{alt} is a normalization factor,

- *cos x* and *sin x*, where *x* is the bearing of the aircraft, i.e. the angle of the agent's course w.r.t North, in degrees,

- $Nh_{speed}=(h_{speed}-min_{hspeed}) / (max_{hspeed}-min_{hspeed})$, where h_{speed} is the magnitude of the agent's horizontal speed in m/s, max_{hspeed} and min_{hspeed} are mormalization factors,

- $cos(x-\psi)$ and $sin(x-\psi)$, where ψ is the relative bearing of the agent w.r.t. the AoR exit point,

- NdistExitPoint= d_{exit}/D_{exit}, where d_{exit} is the horizontal distance of the agent w.r.t. the AoR exit point in meters, and D_{exit} is a normalization factor,

- NaltDiffExitPoint=$|alt - alt_{Exit\ Point}| / |max\ |alt - alt_{Exit\ Point}|$, where $|alt - alt_{Exit\ Point}|$ is the absolute difference in feet between the agent's altitude at the exit point and the filed altitude at the exit point,

- $cos(dcourse_{wp})$ and $sin(dcourse_{wp})$ for each one of the next four waypoints *wp*=1,2,3..., where $dcourse_{wp}$ is the angle of the current agent's course w.r.t. the course that the agent must follow to reach the corresponding waypoint,

- NdistWaypoint= h_{d_wp}/HD for each one of the next four waypoints, where h_{d_wp} is the horizontal distance in meters between the current agent's position and the position of the corresponding waypoint, and *HD* is a normalization factor,

- NaltDiffWaypoint=v_{d_wp}/VD for each one of the next four waypoints, where v_{d_wp} is the vertical distance in feet between the agent's altitude at the waypoint and the filed altitude at the waypoint, and *VD* is a normalization factor.

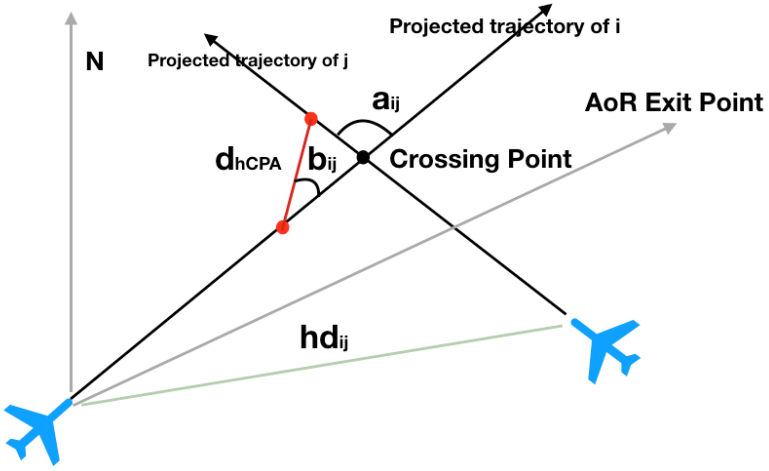


Figure 18. CPA Geometry

EUROPEAN PARTNERSHIP

Co-funded by the European Union

In addition to these observations, each agent *i* maintains a vector e_{ij} with any agent *j* in *Neigh*(*i,AoR,t*), comprising *ij*-edge features that mostly depend on the Closest Point of Approach (CPA): The point in which agents *i* and *j* are estimated to be (or have been) closer using trajectories projections for a time horizon of t_h minutes, as shown in Figure 18:

- $Nt_{CPA}=t_{CPA}/T_{CPA}$, where t_{CPA} is the time required in seconds for agent *i* to reach the CPA with agent *j* and T_{CPA} is a normalization factor,

- $N_{d\_h\_CPA}=d_{h\_CPA}/D_{h\_CPA}$, where d_{h_CPA} is the horizontal distance in meters between agents *i* and *j* at the CPA and D_{h_CPA} is a normalization factor,

- $cos\ a_{i,j}$ and $sin\ a_{i,j}$, where $a_{i,j}$ is the intersection angle in degrees between agents *i* and j,

- $cos\ b_{i,j}$ and $sin\ b_{i,j}$, where $b_{ij,}$ is the relative bearing of agent *i* w.r.t. to agent *j* at the CPA,

- $Nd_{v\_CPA}=v_{d\_CPA}/V_{d\_CPA}$, where v_{d_CPA} is the vertical distance in feet between agents *i* and *j* at the CPA, and V_{d_CPA} is a normalization factor,

- $Nd_{cp}=d_{cp}/D_{cp}$, where d_{cp} is the distance in meters between agents *i* and *j* when any of them passes the crossing point first, and D_{cp} is a normalization factor,

- $Nt_{cp}=t_{cp}/T_{cp}$, where t_{cp} is the time required in seconds for any of agents *i* and *j* to pass the crossing point first, and T_{cp} is a normalization factor,

- $Nd_{h(i,j)}=hd_{i,j}/HD$, where $h_{d\_i,j}$ is the current horizontal distance in meters between agents *i* and *j*,

- $Nd_{v(i,j)}=vd_{i,j}/VD$, where $vd_{i,j}$ is the current vertical distance in feet between agents *i* and *j.*

4.3 CD&R Functional Roadmap

Based on the TAPAS CD&R use case description, the following functions / tasks have been identified for the CD&R use case. For each function / task, the description is provided, as well as the input, output and step of the ATM Master Plan Automation Levels Task Breakdown.

1. Assessment of the planned and desired trajectory profile

This task refers to the evaluation of each aircraft desired profile (flight plan filled and radiofrequency or datalink requests) against the current trajectory to comply as much practicable as possible with the Airspace Users' preferences.

- **Input:** Flight plan, trajectory (radar track) information and AUs' preferences
- **Output:** Gap between flight desired profile and flight trajectory
- **ATM Master Plan Automation Levels Task Breakdown:** Information acquisition and information analysis

2. Identification of potential conflicts

This task is devoted to identifying all the potential conflicts between the aircraft inside the Executive Controller's Area of responsibility.

- **Input:** Flight trajectories (track) and flight plan for the aircraft within Executive Controller's AoR
- **Output:** Potential conflicts within Executive Controller's AoR

EUROPEAN PARTNERSHIP
Co-funded by the European Union

- **ATM Master Plan Automation Levels Task Breakdown:** Information acquisition and information analysis

3. Identification of conflict resolution strategies and clearances proposal

This task includes the proposal of conflict resolution strategies (e.g. flight level change, speed restriction, vectoring, direct to, etc.) for all the conflicts identified in the Executive Controller's Area of Responsibility.

The task is also devoted to the proposal of the most appropriate ATC clearance to be given to the Cabin Crew in order to comply with the proposed resolution strategies and flight sector exit conditions.

This task should consider the management of every planned constraint (flight level, coordination, speed restriction, target time over certain points, etc.).

- **Input:** Potential conflicts within Executive Controller's AoR and flight's planned constraints
- **Output:** Conflict resolution strategies and ATC clearances
- **ATM Master Plan Automation Levels Task Breakdown:** Information analysis, decision and action selection

4. Clearances implementation

This task refers to the implementation of the selected ATC clearances in the corresponding ATC control system.

- **Input:** Conflict resolution strategies and ATC clearances
- **Output:** De-conflicted trajectories for the aircraft within the Executive Controller's AoR
- **ATM Master Plan Automation Levels Task Breakdown:** Action implementation

5. Conformance Monitoring

This function allows the identification of any possible deviation of each aircraft trajectory with regards to a given ATC clearance.

- **Input:** Flight trajectories (radar track) and ATC clearances
- **Output:** Conformance monitoring alerts
- **ATM Master Plan Automation Levels Task Breakdown:** Information acquisition and information analysis

Taking into account the functions identified the following table depicts the allocation of tasks between the human and the machine for the CD&R (Executive Controller) Use Case.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

CDR (Executive) Functions / Tasks	Automation Level 1	Automation Level 2	Automation Level 3
Assessment of the planned and desired trajectory profile	Human	Machine	Machine
Identification of potential conflicts	Machine	Machine	Machine
Identification of conflict resolution strategies and clearances proposal	Human	Machine	Machine
Clearances implementation	Human	Human	Machine
Conformance Monitoring	Machine	Machine	Machine

Table 8. TAPAS CD&R (Executive) Initial Functional Roadmap

Overall, the functional requirements identified for the CD&R use case are as follows:

- The executive controller shall be provided with potential encounters between the aircraft within his area of responsibility for a 7-10-minute look-ahead timeframe.
- The executive controller shall be provided with conflict resolution strategies for the encounters identified between the aircraft within his area of responsibility.
- The executive controller shall be provided with ATC clearances implementation options to solve the identified conflicts.
- The executive controller shall be provided with the ATC clearances required to comply with the agreed exit sector conditions.
- The executive controller shall be able to implement the ATC clearances given to the flight cabin crew.
- The executive controller shall be provided with conformance monitoring alerts, indicating deviations of the flight trajectory compared to the given ATC clearances.
- The executive controller shall be provided with information about the flight desired profile (including radiofrequency and datalink requests), flight constraints and flight actual trajectory.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

4.4 Data Sets

This section specifies the data sets used for training and testing the AI/ML methods for the CD&R case. In the next subsections, we briefly describe the data sets: A full and detailed description of each dataset is provided in deliverable D4.1 [1].

4.4.1 Flight Plans

Temporal coverage: entire 2019
Spatial coverage: Iberian Peninsula
Size on disk: 33.4GB
Distinct FPKeys: 1328640
Distinct RTKeys: 1328848
minAlt: 100.0
maxAlt: 55000.0

This data set reports the flight plans submitted for the region over Iberian Peninsula during 2019.

4.4.2 Radar Tracks

Temporal coverage: entire 2019
Spatial coverage: Iberian Peninsula and Canaria Islands (spatially covers the entire Flight Plans data set)
Size on disk: 183.2GB
Distinct FPKeys: 892027
Distinct RTKeys: 892323

This data set provides information about flights during 2019 over the Iberian Peninsula.

4.4.3 ATCo Events

Temporal coverage: entire 2019
Spatial coverage: Iberian Peninsula
Size on disk: 343 MB

The data is provided in a CSV format, comprising information related to ATCo events.

4.5 Overall CD&R Prototype

Figure 19 provides the overall CD&R prototype, as it is also specified in deliverable D4.1. [1]

EUROPEAN PARTNERSHIP
Co-funded by the European Union

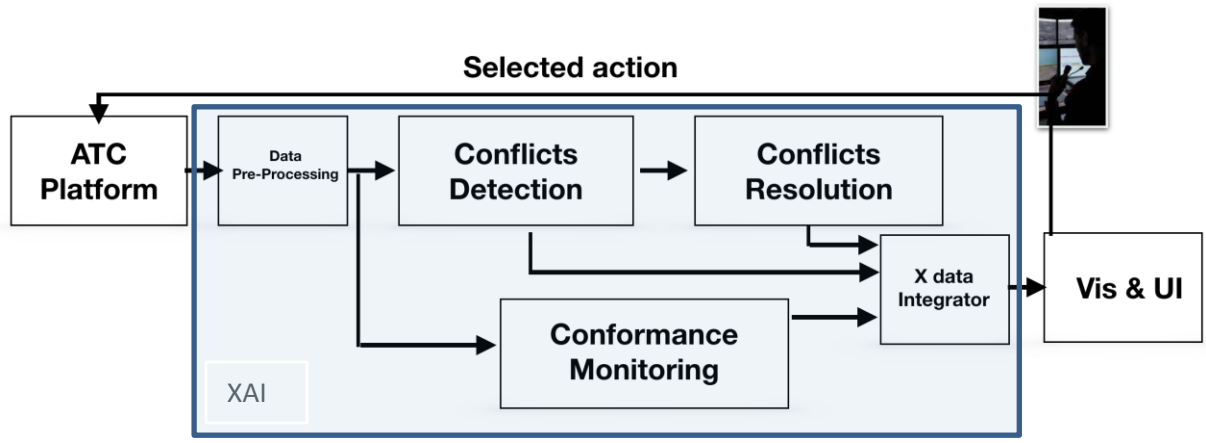


Figure 19. Overall CD&R Prototype System

The prototype system comprises the operational ATC platform SACTA and the XAI system, together with the visualization and user interface (Vis&UI) component, all integrated into a whole.

Specifically, the ATC platform provides updates of radar tracks and flight plans for flights within any specific area of responsibility. These data updates are provided at every P seconds (currently set to P=30sec) which is a system configuration parameter, together with the area of responsibility and all related downstream sectors. The platform receives as input the air-traffic controllers' instructions for the resolution of conflicts.

Data provided (radar tracks and flight plans) are according to the formats specified for the data sources in section 3.4. This data is processed at the data pre-processing component and are forwarded to the conflicts detection and conformance monitoring component. These two last components are jointly referred as traffic simulation component. Data describing the conflicts detected and non-conformance events, are provided in the data integrator component, together with the output of the conflicts resolution component. This last component is the XAI/ML (conflicts resolution) component that decides the instructions to flights for the resolution of conflicts detected. These instructions, together with their effects, as assessed by the system, are provided to the data integrator. The data integrator gathers output data from these three components and provides this to the Vis & UI component.

It must be noted that in Automation level 2 the system provides per conflict detected all potential instructions per flight (including the "no action") ranked according to their potential (as assessed by the system) to resolve the conflict. At this level of automation, the ATCo may choose one of these actions to instruct a flight. At automation level 3, the system instructs the flights to execute actions with no human intervention, on its own initiative.

The CD&R prototype does not have any distinct explainability component for providing explanations of instructions decided by the system. Thus, we do follow a different paradigm from the one followed in the ATFCM case: We provide all relevant parameters that drive system's decisions, offering more transparency on decision making (i.e. making transparent the situations that the system considers and foresees), **focusing on operational concerns**, rather than explainability on **how** decisions are taken from a deep learning model. This is purely an operational transparency perspective regarding AI/ML explainability, where the emphasis is on pragmatic constraints of ATCo.

Data gathered by the data integrator component and provided to the Vis&UI component. This data is shown in Figure 20 and described in full detail in D4.1.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

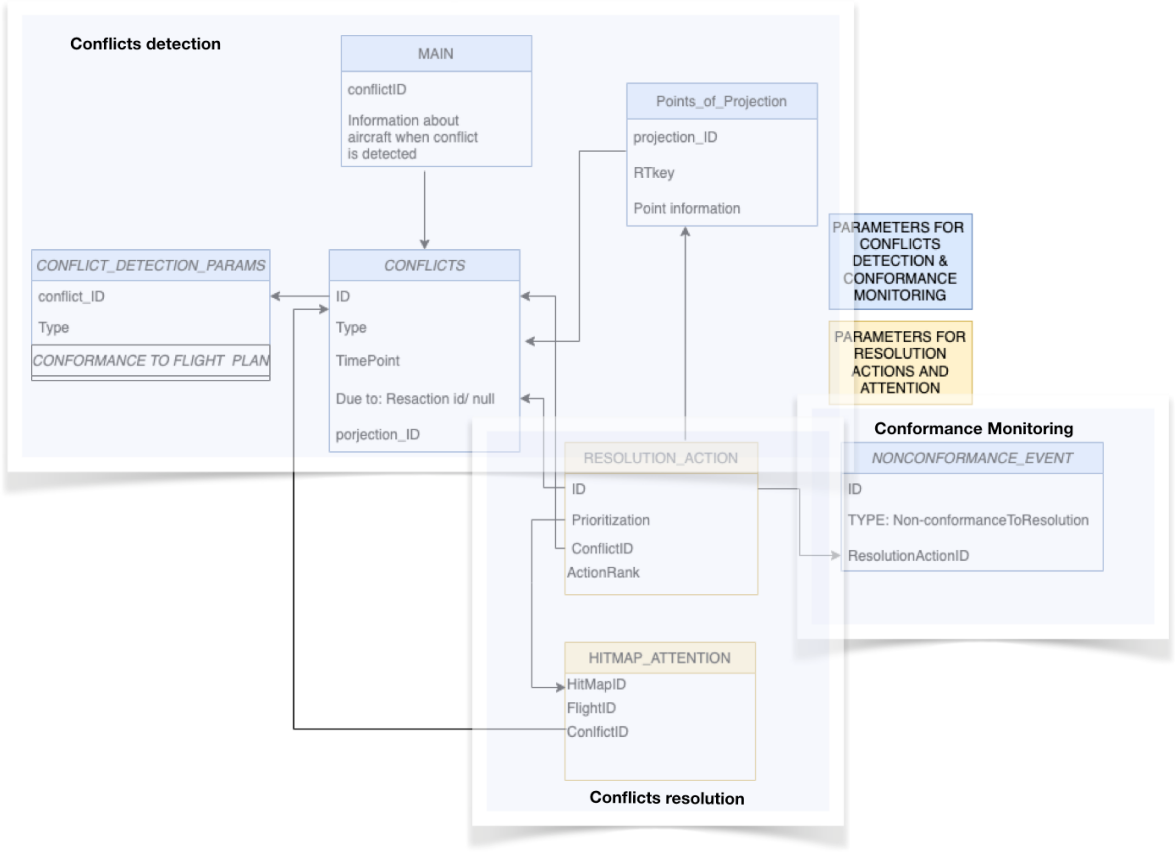


Figure 20. Data provided by the XAI

Subsequent paragraphs provide an overview of the data specified in Figure 21, comprising AI/ML CD&R transparency data.

Overall, the main entity is the conflict (specified in CONFLICTS). This is related to the specific aircraft involved in that conflict (as specified in MAIN), as well as to (a) the parameters assessed for the detection of that conflict (specified in CONFLICT_DETECTION_PARAMS) and to (b) the specific future projections of flight trajectories that drive the detection of the conflict (specified in Points_of_Projection). These are provided by the conflicts detection component.

In case a conflict is assessed to be a side-effect of a potential resolution action (not yet instructed to any of the flights), this conflict is also related to the resolution action that may cause it.

The specific instructions assessed by the conflicts resolution module and all specific parameters (specified in RESOLUTION_ACTION) are related to the conflict to be resolved, as well as future projections of flight trajectories (specified in Points_of_Projection), assessed for all flights following any potential resolution action (i.e. assessing how the trajectory will evolve due to a specific resolution action). Any such resolution action is related to a hitmap that is related to any of the involved flights and specifies the attention of the corresponding agent(flight) to all other agents in its neighbourhood (i.e. the flights with which is in conflict). This data is provided by the conflicts resolution component.

Finally, given that a proposed resolution action is instructed to a flight, the system provides an alert for any non-conformance event (non-conformance to the resolution action) for the flight instructed (this is specified in the NONCONFORMACE_EVENT). Non-conformance events are provided by the conformance monitoring component.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

Given that the data pre-processing component has been described in detail in deliverable D4.1 [20], we will not repeat its description here, where the emphasis is on the AI/ML methods for the resolution of conflicts and their transparency.

However, before delving into the details of the conflicts resolution components, the following two sections describe the functionality of the conflicts detection and the conformance monitoring components.

4.6 Conflicts detection component

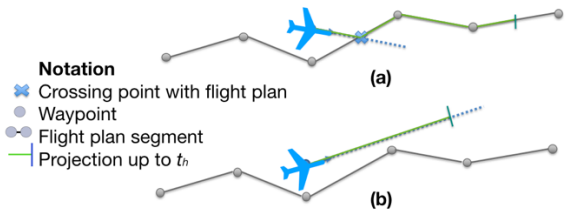


Figure 21. Cases for projecting flights' trajectories up to t_h minutes.

To detect conflicts, a trajectory is projected into the future for a specific time horizon t_h following a nominal approach. As shown in Figure 21, to do so, we need to decide whether a) the flight follows its flight plan and so does the projection, or b) the flight deviates from the flight plan and the projection is estimated according to the flight's course.

To distinguish between these cases, the conflicts detection component considers that a flight follows its flight plan if, either (a) the shortest distance between the aircraft's 2D position and the flight plan horizontal profile is less than a distance threshold d_h, and the difference between the aircraft's and the flight plan's course -at the closest point to the aircraft's position- is less than c_h ; or (b) the line segment defined by projecting the aircraft trajectory for t_h minutes, intersects with the flight plan's horizontal profile. In any other case it is considered that the flight deviates from its flight plan. In any case it is assumed that the flight will retain its current vertical and horizontal speed for t_h. As shown in Figure 21 this procedure results in projections represented by line segments.

To detect any conflict, the following two cases are considered: (a) Both flights have zero vertical speed, (b) one of the flights has a vertical speed greater than zero. In case (a) it is first checked whether the vertical separation minimum is violated. If so, the flights' projection segments that intersect in the temporal dimension is detected and the horizontal Closest Point of Approach (CPA) between these segments is computed using the methodology presented in [20]. If the horizontal separation minimum is violated in any of the computed CPAs then a conflict is detected. In case (b), the CPA in the vertical axis for the projection horizon t_h is computed. If the vertical separation minimum is violated at that point, the process starts from that point and searches in both directions until the horizontal separation minimum gets violated: In such a case occurs at a specific point, a conflict is detected at that point.

The values of the thresholds used are as follows: t_h is equal to the time needed to reach the next flight level, if vertical speed is greater than 0, or equal to 10min, otherwise; d_h = 2 km and c_h = 20 degrees.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

4.7 Conformance monitoring component

The conformance monitoring component monitors whether an aircraft conforms with the conflict resolution actions prescribed. Conformance monitoring is applied when a resolution action is prescribed and ends when the manoeuvre corresponding to the prescribed action completes.

The conformance monitoring component monitors the course of the aircraft and the vertical and horizontal speeds and logs a conformance monitoring event when the "actual" values of these features deviate from the "desired" ones. As "actual" we consider the values of these features according to the radar track information provided by the operational platform. With the term "desired" we refer to the values of the course and horizontal and vertical speed that the aircraft should have according to the prescribed resolution action.

To avoid false positive alarms caused by small deviations between the actual and desired values we do not check for equality between them. Instead, we consider an interval around the desired values and check if the actual values fall in that interval. If so, the aircraft conforms to the prescribed action. More formally the aircraft conforms to the prescribed action if $|d_i - a_i| \leq l_i$ where d denotes the desired value, l the length of the interval divided by 2, a the actual value and $i \in \{course, horizontal\ speed, vertical\ speed\}$ denoting the considered feature.

In practice we use the following values for l_i: $l_{course} = 1degree$, $l_{horizontal\ speed} = 1m/s$, $l_{vertical\ speed} = 2feet/s$.

In case the aircraft does not conform to the prescribed resolution action the component logs a conformance monitoring event that contains information about the feature whose value deviates from the desired one and the actual and desired values of that feature.

4.8 AI/ML component

Considering that cooperation is crucial among flights involved in conflicts, as safety is the top priority of all stakeholders in ATM, followed by flight efficiency, the method adopted for conflicts resolution among agents representing flights is the graph convolutional cooperative reinforcement learning method (DGN [21]). According to the graph convolutional reinforcement learning method, agents (flights) are able to learn cooperative policies by focusing on their neighbors (i.e. those with whom they do interact) instead of taking into account all the existing agents in their environment.

DGN is based on deep Q network (presented in Section 3.6.1) and trained end-to-end. DGN shares weights among all agents involved in a situation - making it easy to scale. It abstracts the mutual inter-play between agents by relation kernels, and extracts latent features by convolution. Modelling the multi-agent interaction setting as a graph, DGN considers that this graph can dynamically change. This is of crucial importance to the CD&R task, as the neighbor agents to an agent *i* (i.e. those representing flights in conflict with a specific flight) may change.

Figure 22 illustrates the architecture of DGN from the view of all agents. DGN comprises of three main modules:

- An encoder for encoding agent's *i* observations at any time step *t*, o_i^t, projecting these observations in a higher dimensional space,

EUROPEAN PARTNERSHIP

Co-funded by the European Union

- Stacked convolutional layers for implicit communication among the neighboring agents, expanding their receptive field,
- and the Q-network which takes as input all features from the encoded observations and the preceding convolutional layers.

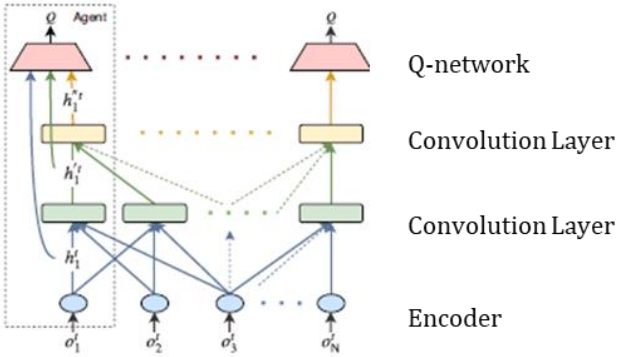


Figure 22. DGN architecture [21].

Describing briefly the operation of DGN, at first, each agent *i* constructs the adjacency matrix C_i^t of size $(|\mathbb{B}| + 1) \times N$, where $|\mathbb{B}|$ is the number of *i*'s neighbors at time point *t*. Each row represents either *i* itself, or $j \in \mathbb{B}$. The neighbors are sorted based on their current distance from *i*. Then, *i*'s observations are encoded (h_i^t) and are fed in the first convolution layer along with its neighbors' encoded observations.

The next steps of the algorithm comprise the computations in the first convolution layer. The encoded observations are projected into query, key, and value ($W_Q h_i$, $W_K h_i$, and $W_V h_i$, respectively) through corresponding MLPs, each one of which is divided into *M* heads. The convolution kernel is the multi-head dot-product attention kernel, which plays a salient role in DGN superior performance. The attention value of head *m* for agent i and its neighbor $j \in \mathbb{B}_{+i}$ is computed as follows:

$$a_{ij}^m = \frac{exp\left(\tau \cdot W_Q^m h_i \cdot (W_K^m h_i)^{\mathrm{T}}\right)}{\sum_{k \in \mathbb{B}_{+i}} exp\left(\tau \cdot W_Q^m h_i \cdot (W_K^m h_k)^{\mathrm{T}}\right)}$$

where τ is a scaling factor. The attention values a_{ij}^m for all $m \in M$ are utilized to acquire the significance of the neighbor-agent *j* for agent *i*, which is reflected in the outcome of the convolution layer ($h_i'^t$). This latent vector is inserted into the second convolution layer along with the corresponding vectors of *i*'s neighbors, where the same operations occur outputting the latent vector $h_i''^t$

The final step concatenates the outputs of the two convolution layers ($h_i'^t$ and $h_i''^t$) and the encoded observations (h_i^t), and feeds them into the Q-network, in order to obtain the Q-values of all potential actions of agent *i*. Q-values, rank all potential actions for agent *i.* These operations are performed for all agents, simultaneously.

Concerning the training process, given that there are *N* agents, at each timestep the tuple (O, A, O', R, C) is stored in a replay buffer, where:

- $\mathcal{O} = \{o_1^t, o_2^t, \dots, o_N^t\}$ is the set of observations of all agents,

EUROPEAN PARTNERSHIP
Co-funded by the European Union

- $\mathcal{A} = \{a_1^t, a_2^t, \dots, a_N^t\}$ is the set of their actions,
- $\mathcal{O}' = \{o_1^{t+1}, o_2^{t+1}, \dots, o_N^{t+1}\}$ is the set of their next observations,
- $\mathcal{R} = \{r_1^{t+1}, r_2^{t+1}, \dots, r_N^{t+1}\}$ is the set of rewards received for the transition of the environment from state *s* to state *s'* due to execution of agents' joint action,
- $\mathcal{C} = \{C_1^t, C_2^t, \dots, C_N^t\}$ is the set of agents' adjacency matrices.

Sampling randomly a minibatch of size *S* from the replay buffer, the loss minimized is the following:

$$\mathcal{L}(\theta) = \frac{1}{S}\sum_{S}\frac{1}{N}\sum_{i=1}^{N}\left(y_i - Q\left(O_{i,\mathcal{C},}^t, a_i^t; \theta\right)\right)^2$$

where $y_i = r_i^{t+1} + \gamma max_{a_i^{t+1}}\left(Q\left(o_i^{t+1}, a_i^{t+1}; \theta'\right)\right)$, $O_{i,C,}^t \subset \mathcal{O}$ denotes the set of observations of the agents in *i*'s receptive field determined by $\mathcal{C}$, γ is the discount factor, and the Q function is parameterized by θ and outputs the value per action a_i^t for agent *i*. An action may change the graph at the next timestep. As suggested in [21], we keep $\mathcal{C}$ unchanged in two successive timesteps when computing the Q-loss in training, to ease learning. Following the paradigm CTDE (as it has been described in Section 3.6.2.3), the gradients of Q-loss of all agents are accumulated to update the set of parameters θ. The target network parameters θ' are updated as follows:

$$\theta' = \beta\theta + (1-\beta)\theta'$$

4.8.1 Enhancing DGN with edges

An essential shortcoming of DGN, according to our needs, is the lack of using edges properties. In our formulation for the ATC problem, edges connecting neighboring agents (flights) are immensely important as the agents would be unaware of their neighbors' relative positions and properties of CPA geometries. For instance, considering observations only, a neighbor *j* of a flight-agent *i* with a specific altitude, speed and bearing could be anywhere in the airspace and therefore *i* could not infer its relative position (relevant to a detected conflict) w.r.t *j* only on the basis of these features.

For this reason, a modified version of DGN was implemented which exploits the information obtained by edges' features. In brief, we use two different MLP to encode observations and edges features, MLP_{obsEnc} and MLP_{EdgEnc}, respectively. We concatenate the encoded vectors (both of observations and edges features) of each $j \in (Neigh(i, AoR, t) + i)$, and then we feed them into the first convolution layer. Recall that $(Neigh(i, AoR, t) + i)$ symbolizes the set of *i*'s neighbors at timepoint *t,* i.e. the set of *conflicting trajectories to $\mathcal{T}_i$ at a specific timepoint t.*

The output of the first convolution layer is concatenated with the encoded edges again and are fed into the second convolution layer. Afterwards, the operations of Q-network are performed as in the original DGN version.

Figure 23 depicts the architecture of the enhanced DGN version used for CD&R from the perspective of each agent, in comparison to the original DGN version.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

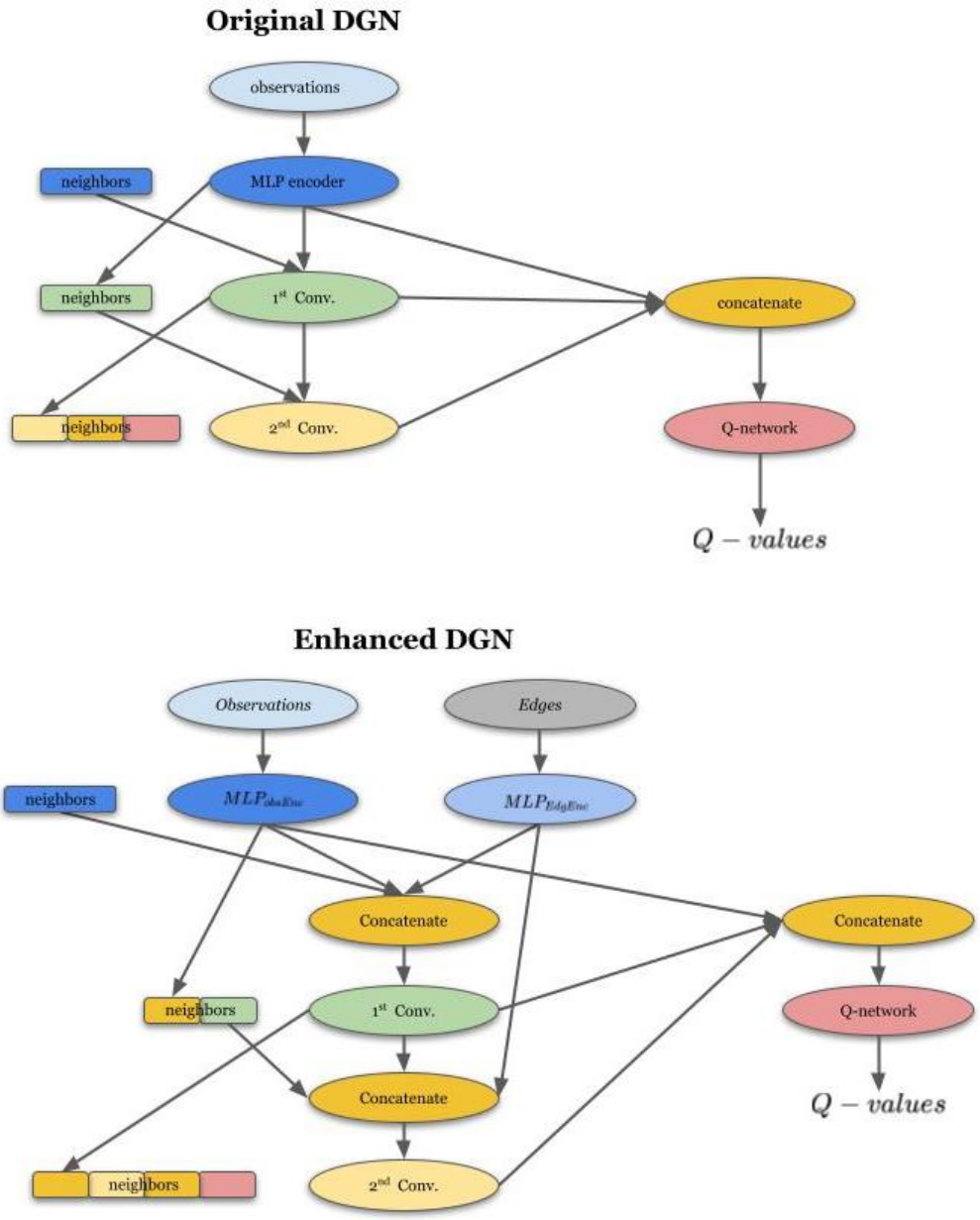


Figure 23. DGN architecture: (Up): Original, illustrating the perspective of each agent, (Down) Enhanced, illustrating the perspective of each agent.

Moreover, we define the way that the neighbors of agent-flight *i* are ordered in the corresponding adjacency matrix as follows:

- We rank first those that are in loss of separation with *i*, which are further ranked by their Euclidean distance to *i*,
- These are followed by those causing an alert with *i* (i.e., the time to the CPA with *i* is below 10 minutes) and are further ranked based on the time they need to reach the CPA, t_{CPA},

EUROPEAN PARTNERSHIP
Co-funded by the European Union

- Those in conflict with *i* follow, ranked based on t_{CPA}, too.
- Finally, those that are in conflict with *i* but with a negative t_{CPA} have the lowest priority.

Ultimately, we use the same loss function as in the original DGN version, but we store the sample of each timepoint in a *prioritized* replay buffer (PER). Each sample-tuple consists of the following elements:

- $\mathcal{O} = \{o_1^t, o_2^t, \dots, o_N^t\}$ is the set of observations features,
- $\mathcal{E} = \{e_1^t, e_2^t, \dots, e_N^t\}$ is the set of edges features,
- $\mathcal{A} = \{a_1^t, a_2^t, \dots, a_N^t\}$ is the set of agents' actions,
- $\mathcal{O}' = \{o_1^{t+1}, o_2^{t+1}, \dots, o_N^{t+1}\}$ is the set of their next observations features,
- $\mathcal{E}' = \{e_1^{t+1}, e_2^{t+1}, \dots, e_N^{t+1}\}$ is the set of edges features,
- $D\mathcal{R} = \{dr_1^{t+1}, dr_2^{t+1}, \dots, dr_N^{t+1}\}$ is the set of the discounted rewards calculated by the history of rewards of each agent; note that each agent obtains a history of rewards as its actions have duration, and thus more than one reward can be received (for each action); the individual rewards (parts of the rewards history) are computed based on the reward function defined in the next section,
- $\mathcal{C} = \{C_1^t, C_2^t, \dots, C_N^t\}$ is the set of agents' adjacency matrices sorting the neighbors according to the aforementioned way.

4.8.2 Conflict Resolution Actions

Here we define a discretized action space. However, the action space consists of the entire palette of actions employed by an ATCo.

The resolution action types include: 1) *A1-flight level change,* 2) *S2-course change*, 3) *A2-speed change*, 4) *A3-direct to waypoint*. Each one of these has more than one candidate values. More specifically, the available values are:

- *A1*: 17, -17 feet/s for ascending or descending course, respectively, until the next flight level is reached,
- *S2*: 10, -10, 20, -20 degrees,
- *A2*: 3.6008, -3.6008 m/s for acceleration or deceleration, respectively,
- *A3*: 1,2,3,4 denoting the next four waypoints.

In addition to the above, an extra "action" implemented, is the *A4-continue course*, which guides the corresponding flight-agent to maintain its course. After experimenting with all these actions, deciding a resolution action every 30 seconds (the data update period from the external operational platform) for each conflicting flight-agent, we concluded that this approach does not properly simulate the CD&R task.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Actually, an ATCo can determine the exact *duration* of a resolution action which means that if the decided duration is *n* seconds at timepoint *t*, then the corresponding flight should execute the instructed resolution action until the timepoint $t + \left(\frac{n}{d}\right)$. If the conflict persists, the ATCO can provide another resolution action (but if the situation worsens, an intervention is possible). Hence, actions' durations are added as options, expanding further the action space.

According to the domain experts, the accepted range of the duration values is 1-3 minutes. Based upon this range, we provide to the flights-agents the opportunity to choose the duration of each action among four values: 30, 60, 120 and 180 seconds. As *A4* resolution action does not match to the case of an action with duration, the only available duration is 30 seconds (which is equal to the time period between two successive timepoints). Unfortunately, as a consequence of limitations in the environment's implementation, *A1* has no duration, or in other words, its duration is equal to the time required for the desired next flight level to be reached. Also, *A3* has a predefined duration equal to the time needed for the flight-agent to reach the corresponding waypoint. Therefore, the actions which have multiple duration values available are *S2* and *A2,* signifying that each conflicting flight-agent at each timestep has to decide the value of its action (e.g., 3.6008 m/s) and its duration (e.g., 60 seconds).

Before concluding about the final action space, it should be pointed out that there is an extra action, *CA* (Continue Action), which is deterministically executed when an *A1* or *A3* action is in progress. For instance, at the timepoint t, a conflicting flight-agent decides to execute such an action, e.g., *A3,* and at the next timepoint $t + 1$, the corresponding waypoint is not reached, then the *CA* action is executed. This action is executed for each discrete timepoint, $until$ the desired next waypoint or the targeted next flight level (if the action in progress is the A1) is reached. Accordingly, when an action of type *S2* or *A2* is in progress, the action *A4 continue course* is executed because it is assumed that the desired change in speed or in course can be applied within the next $d = 30$ seconds after its selection, and thus the flight-agent should continue its course as is until the timepoint $t + (n/d)$ (n is the selected duration).

Ultimately, the action space is composed of 32 individual actions listed below:

1) (*A1*), value: 17 feet/s, duration: predefined,
2) (*A1*), value: -17 feet/s, duration: predefined,
3) (*S2*), value: 10 degrees, duration: 30 seconds,
4) (*S2*), value: 10 degrees, duration: 60 seconds,
5) (*S2*), value: 10 degrees, duration: 120 seconds,
6) (*S2*), value: 10 degrees, duration: 180 seconds,
7) (*S2*), value: -10 degrees, duration: 30 seconds,
8) (*S2*), value: -10 degrees, duration: 60 seconds,
9) (*S2*), value: -10 degrees, duration: 120 seconds,
10) (*S2*), value: -10 degrees, duration: 180 seconds,
11) (*S2*), value: 20 degrees, duration: 30 seconds,
12) (*S2*), value: 20 degrees, duration: 60 seconds,
13) (*S2*), value: 20 degrees, duration: 120 seconds,

EUROPEAN PARTNERSHIP

Co-funded by the European Union

14) (*S2*), value: 20 degrees, duration: 180 seconds,
15) (*S2*), value: -20 degrees, duration: 30 seconds,
16) (*S2*), value: -20 degrees, duration: 60 seconds,
17) (*S2*), value: -20 degrees, duration: 120 seconds,
18) (*S2*), value: -20 degrees, duration: 180 seconds,
19) (*A2*), value: 3.6008 m/s, duration: 30 seconds,
20) (*A2*), value: 3.6008 m/s, duration: 60 seconds,
21) (*A2*), value: 3.6008 m/s, duration: 120 seconds,
22) (*A2*), value: 3.6008 m/s, duration: 180 seconds,
23) (*A2*), value: -3.6008 m/s, duration: 30 seconds,
24) (*A2*), value: -3.6008 m/s, duration: 60 seconds,
25) (*A2*), value: -3.6008 m/s, duration: 120 seconds,
26) (*A2*), value: -3.6008 m/s, duration: 180 seconds,
27) (*A3*), value: 1st waypoint, duration: predefined,
28) (*A3*), value: 2nd waypoint, duration: predefined,
29) (*A3*), value: 3rd waypoint, duration: predefined,
30) (*A3*), value: 4th waypoint, duration: predefined,
31) *(A4)*, value: 0, duration: 30 seconds,
32) *(CA)*, value: 0, duration: 30 seconds.

Another point that should be clarified is the eligibility of resolution actions. Apparently, an aircraft could not increase or decrease its speed indefinitely. Additionally, the four next waypoints do not always exist. In consequence, at each timestep, actions are filtered out in accordance with the flights-agents' state, which means that many flights-agents could have different sets of available actions at the same timestep. Denoting the selected horizontal and vertical speed change value of an agent *i* at a specific timepoint t as $HSpeedCh_i^t$ and $VSpeedCh_i^t$, respectively, we can define the following conditions of filtering:

- If $178.67\frac{m}{s} \leq {h_{speed}}_i^t + HSpeedCh_i^t \leq 261\frac{m}{s}$, then *A2* action with value $HSpeedCh_i^t$ is available, otherwise is filtered out,
- If $-80\frac{feet}{s} \leq {v_{speed}}_i^t + VSpeedCh_i^t \leq 60\frac{feet}{s}$, then *A1* action with value $VSpeedCh_i^t$ is available, otherwise is filtered out,
- If the waypoint wp of agent *i* at the timepoint *t* is available, then *A3* with value wp is available, otherwise is filtered out,
- If $Fphase_i^t = climbing\ or\ descending$, then no action is available.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

According to the he last condition when the flight phase is climbing or descending, then the corresponding flight-agent is prohibited from performing any conflict resolution action. This is an extremely critical safety point as take-off and landing should be realized without any alteration of the flight plan.

4.8.3 Reward Function

The reward function which generates the individual reward r_i^{t+1} of each flight-agent *i* at a timepoint *t*, given the transition of the environment from the joint state $s \in \mathcal{S}$ to $s' \in \mathcal{S}$ by executing the joint action a, is comprised of eight different terms:

$$\begin{aligned} \mathrm{R}_i(s,a,s') = & -w_{\chi\psi} * TdriftFromExitPoint_i^t \\ & -w_{\chi} * TtrackChange_i^t \\ & -w_{alt} * TaltDiffExitPoint_i^t \\ & -w_{v_{speed}} * TvertSpeedChange_i^t \\ & -w_{h_{speed}} * ThorSpeedChange_i^t \\ & -w_{roc} * Troc \\ & -w_a * TnumberOfAlerts_i^t \\ & -w_l * TnumberOfLosses_i^t \end{aligned} \quad (1)$$

where $w_{\chi\psi}$, w_{χ}, w_{alt}, $w_{v_{speed}}$, $w_{h_{speed}}$, w_{roc}, w_a and w_l are weights which determine the importance of each term. By setting the desired values, we can declare our preference or prioritization to the corresponding quantity, managing the flight-agent's behavior. The selected weight values are shown in Table 9. Below, the terms of the reward function are analyzed:

- $TdriftFromExitPoint_i^t = \frac{|\chi_i^t - \psi_i^t|}{X\Psi}$, where χ_i^t and ψ_i^t are included to *Observations* features, and $X\Psi$ is a normalization factor (all factors are also presented in Table 9); this term quantifies the drift from AoR Exit point associated with the corresponding Downstream Sector (AREDS) of flight-agent *i* at timepoint *t*, and it is the angle between *i*'s current course and the straight line between *i*'s current position and the AREDS.
- $TtrackChange_i^t = \frac{|\Delta\chi_i^t|}{\Delta X}$, where $|\Delta\chi_i^t| = |\chi_i^t - \chi_i^{t-1}|$ describes the absolute change in *i*'s track (in degrees) as a result of its individual action a_i^{t-1}, and ΔX is a normalization factor,

EUROPEAN PARTNERSHIP

Co-funded by the European Union

- $TaltDiffExitPoint_i^t = \frac{|alt - alt_{ExitPoint}|_i^t}{AEPDiff}$, where $|alt - alt_{ExitPoint}|_i^t$ is an element of *Observations* features and $AEPDiff$ is a normalization factor; this term penalizes (due to the negative sign in Equation (1)) the difference between *i*'s current altitude and the altitude at the AREDS,
- $TvertSpeedChange_i^t = \frac{\left|\Delta v_{speed}{}_i^t\right|}{V}$, where $\left|\Delta v_{speed}{}_i^t\right| = \left|v_{speed}{}_i^t - v_{speed}{}_i^{t-1}\right|$ is the absolute difference in *i*'s vertical speed resulting from its individual action a_i^{t-1},
- $ThorSpeedChange_i^t = \mathbb{1}_{\Delta h_{speed}{}_i^t != 0}$, where $\mathbb{1}_{\Delta h_{speed}{}_i^t != 0}$ is equal to 0 when *i*'s action does not alter the magnitude of its horizontal speed (i.e., $\left|\Delta h_{speed}{}_i^t\right| = \left|h_{speed}{}_i^t - h_{speed}{}_i^{t-1}\right|$), otherwise is 1,
- $Troc = \sum_{j \in Neigh(i,AoR,t)} \frac{RoC_{i,j}}{\max RoC}$, where $RoC_{i,j}$ is the calculated Rate of Closure w.r.t conflicting flights *i* and *j*, and $\max RoC$ is a normalization factor,
- $TnumberOfAlerts_i^t$ is the number of *i*'s alerts w.r.t to its neighbors $Neigh(i, AoR, t)$,
- $TnumberOfLosses_i^t$ is the number of *i*'s losses of separation w.r.t its neighbors $Neigh(i, AoR, t)$.

Weight	Value	Normalization Factor	Value
$w_{\chi\psi}$	0.5	$X\Psi$	π
w_{χ}	1	ΔX	20
w_{alt}	0.5	$AEPDiff$	6150
$w_{v_{speed}}$	1	V	17
$w_{h_{speed}}$	1	$Rnorm$	20
w_{roc}	1	$\max RoC$	5
w_a	5		
w_l	10		

Table 9. Values of the reward weights and normalization factors.

Concerning the weights of the reward specified in Table 9, we can infer that the priority is in resolving the losses of separation as fast as possible by assigning 10 in w_l weight (the higher the weight value, the higher the priority level of the corresponding term). Next is the resolution of alerts with a weight value equal to 5. The terms correlated to changes in track, horizontal and vertical speed follow, with each one having a weight equal to 1, in order to explicitly penalize the ineffectiveness of the actions equally. The penalization of conflicts through the RoC metric has the same priority as the previous terms, assigning 1 to w_{roc} weight. Finally, the terms which implicitly penalize the ineffectiveness of

EUROPEAN PARTNERSHIP

Co-funded by the European Union

resolution actions have the lowest priority level with 0.5 weight values: The drift from AREDS and the altitude difference between the AREDS and that of the flight-agent *i*, imply how far *i* is from its destination and thus how much will cost to fix its course.

The normalization factor $Rnorm$ in Table 9 is utilized to normalize each reward r_i^t received to keep it as much as possible in the range $[-1,0]$.

4.9 Experimental Setting & Results

4.9.1 Experimental Setting

First we present experimental results with 6 scenarios. Table 10 presents details of these scenarios, where each scenario has an ID in the form "*StartScenarioTimeUnix-AoRsector*": a) the date in unix-time format in which the scenario took place, and b) the sector name, which is considered to be the AoR. Note that "#" stands for "*number of*", and "*LoSs*" stands for "*Losses of Separation*". The number of conflicts include alerts but not the pairs of flights being in loss of separation. We can conclude about the complexity of each scenario not only from the number of conflicts/alerts/LoSs but also from the number of flights and from its duration.

Scenario ID	# Flights	# Conflicts	# LoSs	# Alerts	Duration
1564760140- LECBLVU	32	10	1	3	1620 secs
1564759880- LECBLVU	31	10	1	4	1590 secs
1564741571- LECBVNI	27	10	6	9	1980 secs
1564750070- LECBLVU	38	4	0	0	1530 secs
1564736806- LECBVNI	30	9	1	4	1920 secs
1564751917- LECBP2R	19	1	0	0	660 secs

Table 10. Details of scenarios utilized for training/testing.

Given these 6 scenarios, the scenarios used for training the AI/ML model are the 3 first scenarios in Table 10 and those used for its evaluation are the 3 last scenarios (it must be noticed that validation scenarios, i.e. scenarios used for validating the system, were not provided to WP4). We have selected the most difficult scenarios to train the enhanced DGN version, among 547 different scenarios in terms of number of losses of separation and alerts. In addition, we tested our models (with the same metrics) in both difficult and easy scenarios, to examine its effectiveness and any potential bias (e.g., bias in difficult scenarios similar to those of training). Both, training and testing sets of scenarios, contain scenarios from various sectors

Actually, we have trained two models in order to examine the efficacy and efficiency of two training patterns. These models are denoted by All3 and 3Seq. All3 has been trained with the three first scenarios of Table 10 in a single training process (200 warm up episodes – only collecting samples, 1021 exploration episodes, 979 exploitation episodes, 256 batch size). On the other hand, 3Seq has been trained with the first three scenarios sequentially in three distinct training process: a) at first, we trained the model using the first scenario (1564760140- LECBLVU) and a full training process (the same episodes as mentioned before, but with 10 batch size) obtaining the model 1Seq, b) then we retrained 1Seq with the second scenario (1564759880- LECBLVU) again with a full training process, resulting in 2Seq model, and c) we retrained 2Seq acquiring the model 3Seq.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Scenario ID	# Flights	# Conflicts	# LoSs	# Alerts	Duration
1564760140- LECBLVU	32	9	1	3	1620 secs
1564759880- LECBLVU	31	10	1	4	1590 secs
1564741571- LECBVNI	27	4	6	9	1980 secs
1564750070- LECBLVU	38	2	0	0	1530 secs
1564736806- LECBVNI	30	8	1	4	1920 secs
1564751917- LECBP2R	19	1	0	0	660 secs
1564760500-LECBLVU	31	4	1	2	1470 secs
1564759733-LECBBAS	38	2	1	2	1710 secs
1564759684-LECBBAS	36	2	1	2	1620 secs
1564752230-LECBP2R	27	8	1	4	1620 secs
1564765001-LECBVNI	31	7	0	0	1560 secs
1564731484-LECBP2R	34	0	0	0	1320 secs
1564741469-LECBVNI	29	5	6	9	1860 secs
1564773821-LECBCCC	31	1	2	2	1590 secs
1564730565-LECBP2R	34	0	0	0	1320 secs
1564751659-LECBCCC	29	1	1	2	1530 secs
1564753001-LECBP2R	27	8	1	4	1620 secs
1564759004-LECBBAS	39	2	1	2	2760 secs
1564765000-LECBVN	31	7	0	0	1560 secs
1564745384-LECBP2R	26	1	1	1	870 secs
1564777409-LECBP2R	23	9	0	0	750 secs
1564745315-LECBP2R	26	1	1	1	870 secs
1564734831-LECBBAS	43	5	1	5	1800 secs
1564746345-LECBP2R	43	1	1	2	1920 secs
1564772971-LECBCCC	31	1	1	1	1530 secs
1564745170-LECBP2R	39	1	1	2	1650 secs
1564740976-LECBVNI	29	0	0	0	1800 secs
1564730330-LECBP2R	32	0	0	0	1260 secs
1564731530-LECBP2R	40	0	0	0	1740 secs
1564763981-LECBVNI	54	13	0	4	2040 secs
1564765959-LECBBAS	54	2	0	2	2070 secs
1564763894-LECBP2R	28	4	0	2	1350 secs
1564736579-LECBGO3	34	11	0	0	1230 secs
1564741854-LECBVNI	29	5	6	9	1860 secs
1564745446-LECBP2R	26	1	1	1	870 secs

EUROPEAN PARTNERSHIP

Co-funded by the European Union

1564771680-LECBLVU	29	0	1	1	1920 secs
1564735391-LECBBAS	36	1	1	1	1560 secs
1564740452-LECBGO3	35	3	0	2	1230 secs
1564757225-LECBBAS	31	1	0	1	1530 secs
1564756177-LECBLVU	19	2	0	1	690 secs

Table 11. Details of scenarios utilized for training/testing based on the improved conflict detection method.

The enhanced DGN method hyperparameters values used in all experiments are listed in Table 12.

The final model utilized in TAPAS second (i.e. that corresponding to the CD&R use case) validation exercise has been trained with 36 scenarios (the first 34 of Table 11 –2 of them were reused). The model has been evaluated with the 6 last scenarios in the same table. We declare this model as 6Seq6 because we have trained it sequentially in 6 training process, feeding it with samples (in each training process) from 6 scenarios (different 6 scenario for each training process). Each training process consists of 200 warm up episodes, 3000 exploration episodes and 1000 exploitation episodes, using 256 batch size.

It must be noted that moving from the 3Seq to 6Seq6 models DGN we added the RoC term in the reward function and the conflict detection method was improved in some aspects: This is the reason for some discrepancies in the scenarios that appear in both Table 10 and Table 11 (the 6 first scenarios of Table 11 are identical to those of Table 10).

Hyperparameter	Enhanced DGN
γ	0.96
batch size (S)	$10/256$
buffer capacity	2×10^5
β	0.01
$\varepsilon/\ min\ \varepsilon/\ \varepsilon\ decay$	$0.6/0.001/0.996$
# exploration episodes	1021/3000
# exploitation episodes	979/1000
# warmup episodes-filling PER	200
optimizer	Adam
a (learning rate)	10^{-5}
# training sub-steps per episode	80
# neighbors	3
M (#attention heads)	8
H (length of the vector of each head)	16
# $MLP_{ObsEnc}/MLP_{EdgEnc}$ layers	$2/2$
# $MLP_{ObsEnc}/MLP_{EdgEnc}$ neurons	$(512,128)/(512,128)$
$MLP_{ObsEnc}/MLP_{EdgEnc}$ activation	ReLU/ReLU
Q-network	affine transformation
weights initializer	random normal
PER $betta$ initial value	0.4

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Hyperparameter	Enhanced DGN
PER $max\ betta$	1
PER $betta\ decay\ (step)$	0.0025
PER $epsilon$	0.05
PER $alpha$	0.6

Table 12. Hyperparameters of DGN.

4.9.2 Experimental Results

Proceeding to the presentation and analysis of training and testing results, we start by comparing *All3* with the sequential model *3Seq*, and then we present the results of *6Seq6*. All outcomes are demonstrated both in figures and tables. It should be noticed that the curves in figures have been smoothed out by calculating the moving average in a window of size 100. The shadowed area is enclosed by the minimum and maximum of each window.

Figure 24 shows the curves corresponding to training the *All3* and *3Seq* models. In order to fairly compare them, we should note that the curves of alerts, LoSs and ATCo instructions of *All3* illustrate the sum of all timepoints of all scenarios in each episode. For this reason, we also summed up the corresponding results of *1Seq, 2Seq, 3Seq* which are depicted by the orange curves. This comparison might lack of fairness, to some extent, because each sequential model has been trained with 2000 episodes (but with smaller batch size than *All3*). However, we can get a comparison that can be combined with the results of Table 13, to draw conclusions about their performance in scenarios in which they have been trained, as well as to unseen scenarios.

It is worth noting that we provide the number of ATCo instructions (both in figures and table results) in order to assess the ability of our models to contribute to the automation of CD&R by offering few and efficient conflict resolution instructions to involved flights. We do not count any A4 or CA action as an ATCo instruction due to the nature of such actions. We also provide the additional nautical miles (NMs) resulting from the instructions applied, in comparison with the (conflicting) flight plan. In this way, we can quantify the cost of lateral only maneuvers (as a result of *S2* or *A3* action types) which is critical to evaluate the efficiency of resolution actions.

Regarding the mean reward curve, for *All3* we have computed the average over all agents and all timepoints for all scenarios of each episode, while for *3Seq* we have computed the average over all agents and all timepoints only for the scenario 1564741571- LECBVNI (the last scenario in sequence in which *3Seq* has been trained). Relying on Figure 24, *3Seq* outperforms *All3* in terms of stable curves of alerts and LoSs, which are of our greatest interest due to the safety criticality of CD&R process. Note that a slight variance at the end of training is a reasonable outcome due to the randomness of selecting actions, which benefits the agents' training.

Based on Table 13, both *3Seq* and *All3* models manage to resolve all LoSs that appear in the original scenarios with which the models have been trained. However, *All3* seems to suffer from overfitting as it increases the LoSs of the unseen scenarios 300%, compared to those in the original scenarios, the alerts 100%, and the conflicts 57%. However, it manages to resolve conflicts in 66% of the unseen scenarios. Note that we consider a scenario to be resolved when $\#\ LoSs = 0$ is true. On the other hand, *3Seq* increases the LoSs of the unseen scenarios 100% (3 times lower than *All3*), the alerts 30% (almost 3 times lower than *All3*), while it reduces conflicts by -21% (*All3* increases them). It resolves

EUROPEAN PARTNERSHIP

Co-funded by the European Union

only 33% of the unseen scenarios, but this is due to its inability to resolve the single LoSs of scenario 1564751917- LECBP2R.

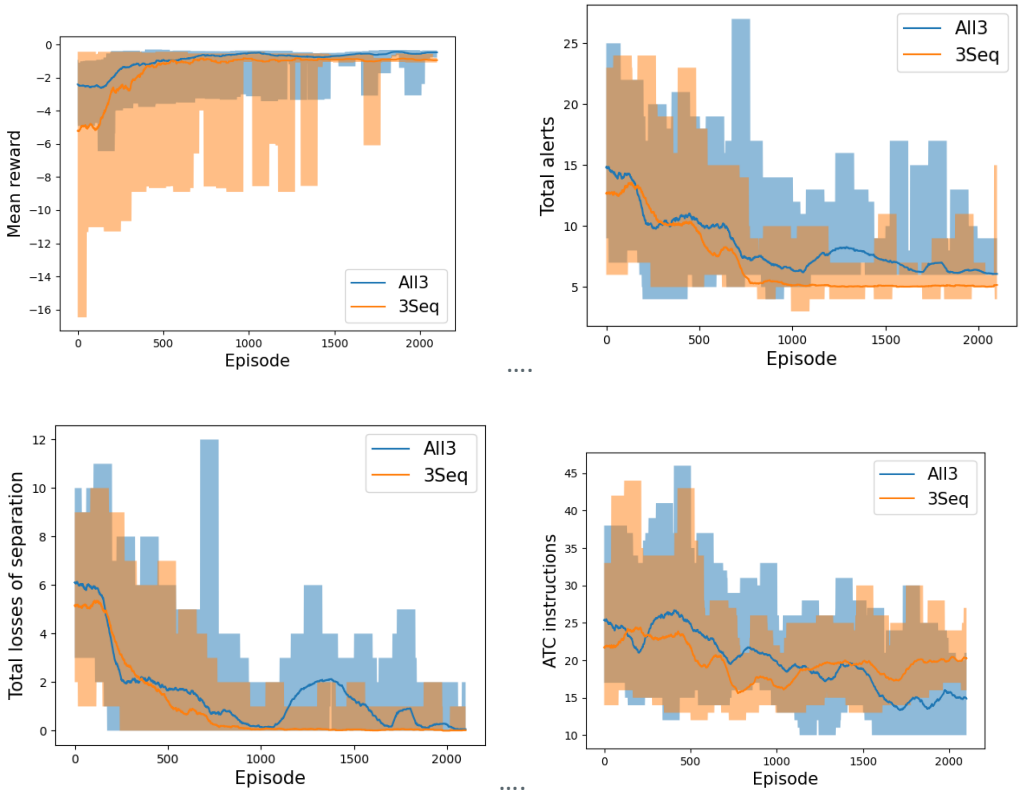


Figure 24. Comparative results of All3 and 3Seq training. (Up-Left): Mean reward over all agents and all timepoints of all scenarios in each episode, (Up-Right):Total alerts of all timepoints of all scenarios in each episode, (Bottom-Left): Total LoSs of all timepoints of all scenarios in each episode, (Bottom-Right): Total ATC instructions of all timepoints of all scenarios in each episode.

Examining the learning process of the sequential models, we can infer that the agents do not forget the knowledge accumulated during previous training processes. This means that their ability to generalize to unseen scenarios is progressively expanded along with the training. Specifically, 1Seq cannot deal with the unseen scenarios, increasing conflicts, alerts and LoSs by 121%, 275% and 300%, respectively. 2Seq reduces the percentage of conflicts increment (of unseen scenarios) to 50% (thus there is a reduction from 121% to 8%) while it manages to reduce the percentages of alerts and LoSs increment (from 275% to 75% and from 300% to 200%, respectively). Ultimately, after three training process, 3Seq is able enough to reduce conflicts by −21%, and further reduce the percentages of alerts and LoSs increment (from 75% to 30% and from 200% to 100%, respectively). Due to 3Seq promising results, we decided to train 6Seq6 in a sequential way, but using batches of 6 scenarios (instead of only 1) in order models trained to generalize effectively (as done by All3) and to speed up the training process.

EUROPEAN PARTNERSHIP
Co-funded by the European Union

Measure / Model	Increase/Decrease conflicts[1] %		Increase/Decrease LoSs %		Increase/Decrease. alerts %		Scenario Resolved[2] %		Average number of ATCo instructions		Average additional NMs	
	Training[3]	Testing[4]	Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing
All3	−56%	57%	−100%	300%	−63%	100%	100%	66%	7	7	2	2
3Seq	−76%	−21%	−100%	100%	−63%	30%	100%	33%	4	6	1	−2
2Seq	−65%	50%	−100%	200%	−43%	75%	100%	0%	6	7	0	17
1Seq	−30%	121%	−100%	300%	−33%	275%	100%	0%	7	9	0	10

Table 13. Rounded statistical measures of training/testing results for All3, 3Seq, 2Seq, and 1Seq models.

Figure 25 shows the curves regarding the training of 6Seq6 model (only for the final, 6th training process). Apparently, LoSs and alerts decrease as the training progresses, whereas mean reward increases, which is what we expect.

Given also the detailed results shown in

Table 14, we can conclude that *6Seq6* generalizes well, yielding comparable results between training and testing scenarios. It solveσ the 50% of the unseen scenarios (the 6 last of Table 15) with only 3 resolution actions on average and 0.0 additional nautical miles on average.

It must be noted that we do not compare *6Seq6* results with those of *All3* and *3Seq*, as *6Seq6* has been trained with many more scenarios and with an improved model and conflicts detection method. In addition, the different way that the conflicts are detected (in *6Seq6* version) does not allow us to compare them fairly.

Concluding, experimental results show that the sequential training pattern manages to fit models that are more robust and generalizable to unseen scenarios, while the training process is very efficient. Therefore, the delivered model (*6Seq6*) has been trained in 6 batches of 6 scenarios (in total 36 scenarios) in sectors above the Barcelona Airport, managing to solve a very high percentage of conflicts in test scenarios (on average 90% for test scenarios) with few resolution actions (~2.5 on average for test scenarios) and with few additional NMs due to resolution actions (~0.35 additional NMs on average for test scenarios). It must be emphasized that this *6Seq6* model, has been validated during the TAPAS validation exercises in sectors above the Madrid Airport, i.e., in sectors and scenarios it has not met during training and testing.

[1] Percentage w.r.t. the corresponding measure in the original scenario

[2] *Scenarios Resolved* stands for the number of scenarios for which # $LoSs = 0$ is true.

[3] *R*efers to the scenarios with which the corresponding model has been trained.

[4] *R*efers to the scenarios with which the corresponding model has been tested.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

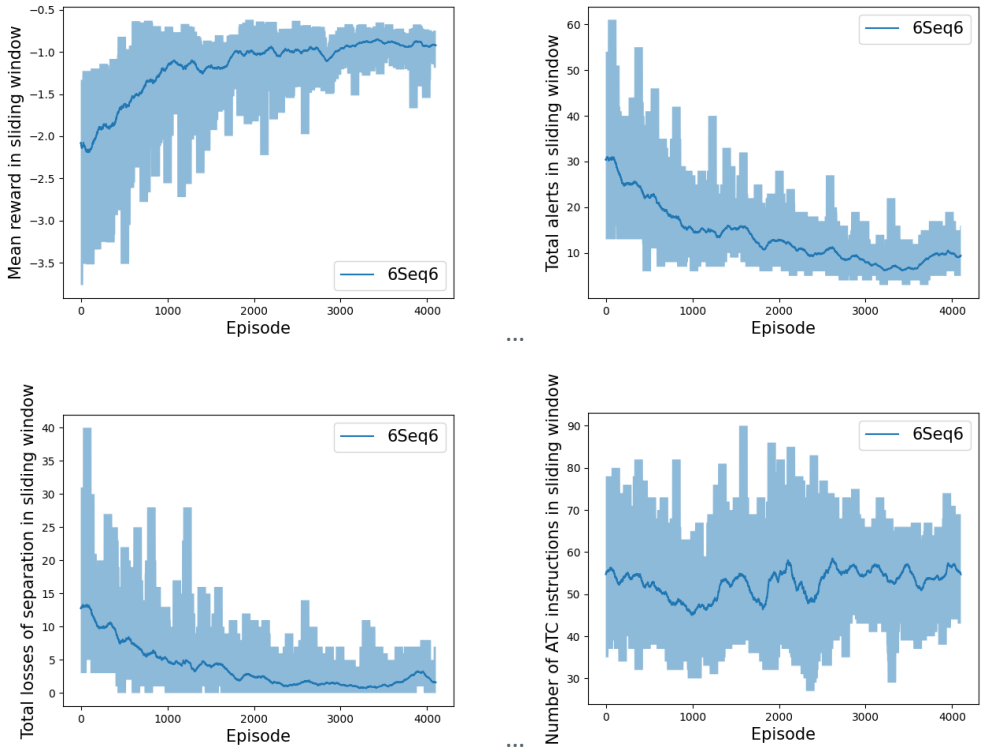


Figure 25. Results of 6Seq6 training. (Up-Left): Mean reward over all agents and all timepoints of all scenarios in each episode, (Up-Right):Total alerts of all timepoints of all scenarios in each episode, (Bottom-Left): Total LoSs of all timepoints of all scenarios in each episode, (Bottom-Right): Total ATC instructions of all timepoints of all scenarios in each episode

Measure	Increase/Decrease conflicts[5] %		Increase/Decrease LoSs %		Increase/Decrease. alerts %		Scenario Resolved[6] %		Average number of ATCo instructions		Average additional NMs	
Model	Training[7]	Testing[8]	Training	Testing	Training	Testing	Training	Testing	Training	Testing	Training	Testing
6Seq6	−31%	−33%	−38%	−42%	−40%	−43%	65%	50%	4	3	2	0

Table 14. Rounded statistical measures of training/testing results for 6Seq6 model

[5] Percentage w.r.t. the corresponding measure in the original scenario

[6] *Scenarios Resolved* stands for the number of scenarios for which # $LoSs = 0$ is true.

[7] *R*efers to the scenarios with which the corresponding model has been trained.

[8] *R*efers to the scenarios with which the corresponding model has been tested.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

Model-Scenario ID[9]	# Conflicts	# LoSs	# Alerts	# ATCo instructions	Additional NMs	Train[10]
6Seq6-1564760140-LECBLVU	7	2	4	9	10.14	True
All3-1564760140-LECBLVU	5	0	2	8	2.74	True
3Seq-1564760140-LECBLVU	3	0	2	6	0.35	True
2Seq-1564760140-LECBLVU	3	0	2	6	0.05	True
1Seq-1564760140-LECBLVU	7	0	2	7	0.0	True
6Seq6-1564759880- LECBLVU	8	3	6	13	10.14	True
All3-1564759880- LECBLVU	7	0	3	10	2.18	True
3Seq-1564759880- LECBLVU	3	0	2	5	3.20	True
2Seq-1564759880- LECBLVU	4	0	2	6	0.70	True
1Seq-1564759880- LECBLVU	9	2	5	7	0.0	False
6Seq6-1564741571- LECBVNI	1	0	1	2	0.0	True
All3-1564741571- LECBVNI	1	0	1	2	0.04	True
3Seq-1564741571- LECBVNI	1	0	1	2	0.0	True
2Seq-1564741571- LECBVNI	2	1	2	2	0.05	False
1Seq-1564741571- LECBVNI	3	2	3	3	0.13	False
6Seq6-1564750070- LECBLVU	10	0	2	10	2.25	True
All3-1564750070- LECBLVU	19	4	7	18	4.4	False
3Seq-1564750070- LECBLVU	8	1	4	12	−8.13	False
2Seq-1564750070- LECBLVU	18	1	6	13	49.8	False
1Seq-1564750070- LECBLVU	21	1	8	20	−4.17	False
6Seq6-1564736806- LECBVNI	1	0	0	2	0.0	True
All3-1564736806- LECBVNI	2	0	1	2	2.74	False
3Seq-1564736806- LECBVNI	1	0	0	1	0	False
2Seq-1564736806- LECBVNI	1	1	2	4	2.01	False
1Seq-1564736806- LECBVNI	3	1	2	4	34.60	False
6Seq6-1564751917- LECBP2R	1	2	2	4	−0.55	True
All3-1564751917- LECBP2R	1	0	0	2	0.28	False
3Seq-1564751917- LECBP2R	1	1	2	4	1.30	False
2Seq-1564751917- LECBP2R	2	1	1	3	−0.06	False
1Seq-1564751917- LECBP2R	7	2	5	2	0	False
6Seq6-1564760500-LECBLVU	7	1	2	5	4.94	True
6Seq6-1564759733-LECBBAS	2	0	1	2	0.25	True
6Seq6-1564759684-LECBBAS	2	0	1	2	0.25	True
6Seq6-1564752230-LECBP2R	3	0	1	6	6.65	True
6Seq6-1564765001-LECBVNI	2	0	0	4	1.48	True
6Seq6-1564731484-LECBP2R	0	0	0	0	0.0	True
6Seq6-1564741469-LECBVNI	2	0	1	4	0.0	True
6Seq6-1564773821-LECBCCC	1	3	3	5	−25.36	True
6Seq6-1564730565-LECBP2R	0	0	0	0	0.0	True

[9] *Model-Scenario ID* indicates the model used and the scenario to which the results refer.

[10] *Train* denotes whether the corresponding model has been trained with the specific scenario or not.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

6Seq6-1564751659-LECBCCC	1	1	2	1	0.0	True
6Seq6-1564753001-LECBP2R	3	0	1	6	6.65	True
6Seq6-1564759004-LECBBAS	2	0	1	2	0.25	True
6Seq6-1564765000-LECBVN	2	0	0	4	1.48	True
6Seq6-1564745384-LECBP2R	0	1	1	1	0.0	True
6Seq6-1564777409-LECBP2R	11	2	3	10	−1.15	True
6Seq6-1564745315-LECBP2R	0	1	1	1	0.0	True
6Seq6-1564734831-LECBBAS	2	0	1	3	0.28	True
6Seq6-1564746345-LECBP2R	1	2	3	2	0.0	True
6Seq6-1564772971-LECBCCC	1	1	1	2	0.0	True
6Seq6-1564745170-LECBP2R	1	2	3	2	0.0	True
6Seq6-1564740976-LECBVNI	0	0	0	0	0.0	True
6Seq6-1564730330-LECBP2R	0	0	0	0	0.0	True
6Seq6-1564731530-LECBP2R	0	0	0	0	0.0	True
6Seq6-1564763981-LECBVNI	12	0	3	17	−26.8	True
6Seq6-1564765959-LECBBAS	1	0	1	1	0.0	True
6Seq6-1564763894-LECBP2R	3	0	1	6	2.23	True
6Seq6-1564736579-LECBGO3	4	0	0	6	6.38	True
6Seq6-1564741854-LECBVNI	2	0	1	4	0.0	True
6Seq6-1564745446-LECBP2R	0	1	1	1	0.0	False
6Seq6-1564771680-LECBLVU	0	1	1	2	0.0	False
6Seq6-1564735391-LECBBAS	1	0	0	2	0.0	False
6Seq6-1564740452-LECBGO3	6	0	0	6	0.0	False
6Seq6-1564757225-LECBBAS	1	1	1	1	0.23	False
6Seq6-1564756177-LECBLVU	2	0	1	3	1.85	False

Table 15. Results of testing the models in various scenarios.

4.10 Transparency

Data provision for transparency is triggered by any conflict. Transparency requirements gathered from ATCo specify that for any specific conflict, they do need to be provided with (a) data regarding the conflict detected and how it has been assessed, (b) data regarding resolution action per aircraft and major factors driving decisions w.r.t. the interests of ATCo and AU, as well as (c) deviations from the flight plan and violation of AoR exit point constraints.

This supports ATCo to assess a situation and decide on the appropriate action w.r.t. operational constraints: Specifically, they do need information that reflects the system's view of the conflicts, of the foreseen effects of actions, as these are estimated by the system, and a ranking of the alternative actions per conflict instance, while detecting conflicts and operating on their AoR. The effects of a resolution action concern foreseen conflicts caused due to the resolution action, nautical miles added to the flight trajectory and additional flight time, deviation from course, foreseen alerts or losses of separation. These should be provided with respect to operational desiderata (specified below), and at the appropriate level of detail.

Specifically, given that there is little or no time for them to exploit and explore detailed explanations while monitoring and resolving safety critical situations, transparency must satisfy Grice's maxims: Be informative and give the quantity of information needed and no more, provide adequate information that describes adequately the situation assessed by the system, be relevant, clear and brief. However,

EUROPEAN PARTNERSHIP

Co-funded by the European Union

the decision to ask for further information about the proposals/decisions of the system should remain with human operators: If detailed transparency data is provided all the time, the increase in the level of workload of the operators might not compensate the benefit provided by the transparency itself.

Regarding any conflict detected at time point, the following information is provided for situation awareness: (a) Information about the flights, including the callsign and the origin and destination airports; (b) Information about the current state of the aircraft, this includes the current position of the aircraft, the current course and the current horizontal and vertical speeds, (c) The discrepancy between the actual track of each of the aircraft and of the corresponding flight plan in 3D (horizontal and vertical dimensions); this includes the course, the altitude and the horizontal and vertical speeds, at the first point of plan/track intersection or at the closest point between the aircraft course, and the latest flight plan; (d) Whether the conflict has been detected using a projection of the actual trajectory or using the flight plan; (e) The spatiotemporal points of the projection (d) The features regarding the geometry of the CPA detected; (f) The first and the final points of conflict detected; (g) Foreseen horizontal, vertical and temporal distance to the CPA, to the first and last points of conflict and to the intersection point for each of the aircraft; and (h) Whether each aircraft is in the climb or descend phase.

Besides the proposed resolution action and their duration, the XAI component provides the following information for each conflicting flight, as part of the transparency requirements:

a) Nautical miles that may be added to the trajectory due to any change caused by the corresponding resolution action,
b) Additional duration imposed on the intended flight time caused by the respective action,
c) The deviation from the trajectory course caused by the action,
d) The deviation from the AoR exit point in 3D (horizontal and vertical shift),
e) The change in the horizontal and vertical speed in accordance with the proposed resolution action,
f) The bearing of the flight (i.e., the angle of the flight's course w.r.t North) after applying the proposed action,
g) For all neighboring flights of a flight-agent *i*, the system provides an attention heatmap, showing the neighboring flights at which *i* pays attention (and how much attention it pays to each one of these), prioritizing the resolution of conflicts.
h) Additional conflicts that may be caused by the resolution action: For each of these conflicts, conflict-relevant information is provided;
i) Losses of separations with other flights, or alerts foreseen, due to the resolution action

The attention heatmap holds a single value (attention value) for each neighbor of the corresponding flight *i*. Note that each neighbor is a flight being in conflict with flight *i*. To obtain a single attention value for each neighbor $j \in Neigh(i, AoR, t)$, we keep the maximum of all heads' values which are referred to *j*. It should be pointed out that these values come from the second convolution layer.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

As shown in Figure 26, for each neighbor-agent there are multiple attention values (one for each head). Relying on [22], we have decided to keep the maximum value of all heads (for each neighbor) as it is the most representative. Regarding the appropriate convolution layer from which we should capture the maximum attention values, we preferred to use the second one because it is the last round of "communication" among agents, and thus accumulates the knowledge for the agents' expanded receptive fields. In short, the obtained attention value indicates the significance of each *i*'s neighbor (and the significance of itself) in *i*'s decision to propose the top-1 resolution action.

Furthermore, we should highlight the fact that we calculate the additional nautical miles only for action types *S2* and *A3* (that is course change and direct to waypoint, respectively), because we assume that action types *A1* and *A2* (flight level change and speed change) cannot impose an increase on nautical miles. Additionally, we assume that:

- Vertical speed can change only when an *A1* action is executed, and the change will be equal to value of the proposed resolution action (+/- 17 feet/s),
- Horizontal speed can change only when an *A2* action is executed, and the change will be aligned with the proposed resolution action (+/- 7 knots),
- Course can change only when an *A3* or *S2* action is executed, and the change (in case of *A3*) is calculated based on the position of the corresponding waypoint, or (in case of *S2*) will be equal to the value of the proposed resolution action (10/-10/20/-20 degrees).

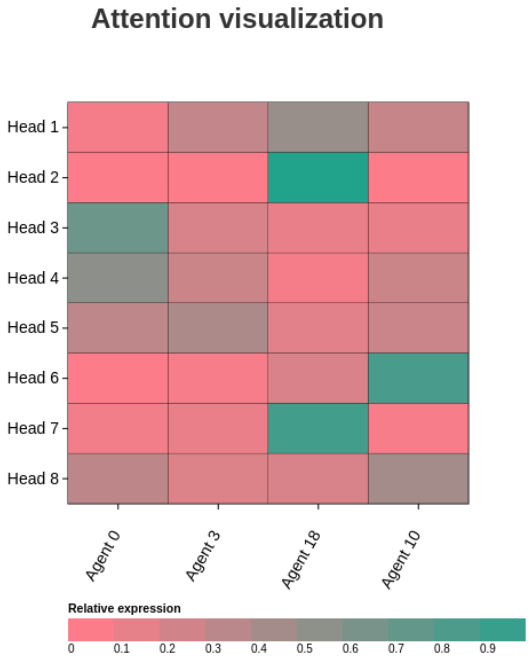


Figure 26. An example of agents' attention extracted from the second convolution layer of Agent 0.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

5 Conclusions

This report serves as a reference for the AI/ML and XAI methods implemented, addressing the requirements towards explainable Artificial Intelligence methods for the ATFCM and CD&R use cases in the context of the TAPAS project. The document, for reasons of conciseness succinctly describes the ATFCM and CD&R use cases and the roadmap of the functionalities decided for each use case, up to realizing the needs for implementing automation level 2.

It specifies the data sources being used per use case, as well as the pre-processing of the data to train and test the AI/ML module. It describes in detail the AI/ML method implemented, and the types of solutions provided (a) for the ATFCM use case, depending on the DCB measures to be applied and (b) for the CD&R use case, depending on the conflict resolution actions that are included in the repertoire of actions.

The document describes the XAI paradigm followed per use case towards providing explanation content regarding agents' (representing individual flights) individual decisions on the DCB measures / resolution actions applied, in detail.

Specifically, for the ATFCM use case, where decisions are taken at the pre-tactical phase of operations, the explainability paradigm followed is mimicking the AI/ML model, providing explanations for each individual decision, at a fine level of granularity. However, for the CD&R use case, were decisions are taken at operational time, and considering the safety criticality of the cases, major emphasis has been given on operational transparency w.r.t. operational constraints and requirements as expressed by ATCo.

Experimental results from the AI/ML and XAI methods show the quality of the solutions computed per use-case, as well as the sample-efficiency for training the XAI models, and their generalization abilities.

EUROPEAN PARTNERSHIP

Co-funded by the European Union

6 References

[1] TAPAS, “D2.1 TAPAS Use Cases Description,” 2020.

[2] TAPAS, “D2.2 Consolidated Requirements and Functional Roadmap,” 2020.

[3] TAPAS, “D3.1 Use Cases Transparency Requirements” 2021.

[4] R. Barto and A. Sutton, Reinforcement learning: An introduction, MIT Press, 2019.

[5] V. Mnih and e. al., “Human-level control through deep reinforcement learning,” Nature 518(7540), 2015.

[6] J. Tsitsiklis and B. VanRoy, “An Analysis ofTemporal-Difference Learning with Function Approximation,” IEEE Trans. on Automatic Control, 1997.

[7] H. VanHasselt, A. Guez and D. Silver, “Deep reinforcement learning with double q-learning,” AAAI, 2016.

[8] H. Hasselt, “Double Q-learning.,” Advances in neural information processing systems 23, 2010.

[9] T. Schaul, J. Quan, I. Antonoglou and D. Silver, “Prioritized experience replay. ,” arXiv preprint arXiv:1511.05952., 2015.

[10] V. Behzadan and W. Hsu, “Analysis and improvement of adversarial training in dqn agents with adversarially-guided exploration (age),” arXiv preprint arXiv:1906.01119, p. 2019.

[11] V. Behzadan and A. Munir, “Whatever does not kill deep reinforcement learning, makes it stronger.,” arXiv preprint arXiv:1712.09344, 2017.

[12] L. Pinto, J. Davidson, R. Sukthankar and A. Gupta, “Robust adversarial reinforcement learning,” ICML - PLMR, 2017.

[13] J. K. Gupta, M. Egorov and M. Kochenderfer, “Cooperative multi-agent control using deep reinforcement learning.,” AAMAS 2017, 2017.

[14] G. Hinton, O. Vinyals and J. Dean, “Distilling the Knowledge in a Neural Network. Deep Learning andRepresentation Learning Workshop,” in NIPS.

[15] A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu and R. Hadsell, “Policy Distillation,” arXiv:1511.06295v2 [cs.LG], p. 2015.

[16] G. Liu, O. Schulte, W. Zhu and Q. Li, “Towards interpretable deep reinforcem,ent learning with linear model u- trees,” ECML/PKDD, 2018.

[17] H. Gouk, B. Pfahringer and E. Frank, “Stochastic Gradient Trees,” ACML 2019, also in arXiv:1901.07777, 2019.

[18] TAPAS, “D4.1 TAPAS Integrated Prototype” 2022.

[19] TAPAS, “D4.3 Visualizations and Visual Analytics Methods,” 2022.

[20] N. P. T. S. A. V. D. a. D. D. Duc-Thinh Pham, “A machine learning approach for conflict resolution in dense traffic scenarios with uncertainties,” in ATM Seminar , 2019.

[21] J. Jiang, C. Dun and L. Zongqing , “Graph Convolutional Reinforcement Learning for Multi-Agent Cooperation,” ArXiv, 2018.

[22] G. Tang, R. Sennrich and J. Nivre, "An Analysis of Attention Mechanisms: The Case of Word Sense Disambiguation in Neural Machine Translation," in Proceedings of the Third Conference on Machine Translation: Research Papers, Brussels, 2018.

[23] C. D. a. L. Z. J. Jiang, “Graph Convolutional Reinforcement Learning for Multi-Agent Cooperation,” arXiv, 2018.

[24] R. S. a. J. N. G. Tang, “An Analysis of Attention Mechanisms: The Case of Word Sense Disambiguation in Neural Machine Translation,” in Third Conference on Machine Translation: Research Papers, Brussels, 2018.

EUROPEAN PARTNERSHIP

Co-funded by the European Union