Post-hoc Explainability for Deep Reinforcement Learning

Master Thesis submitted for partial fulfillment of the requirements for the award of the double degree between the Politecnico di Torino and Grenoble INP Ensimag, a Master of Data science and Computer Engineering

Submitted by

Antonin POCHÉ

Supervisors

Xabier JAUREGUIBERRY
R.D. Team Lead
Delfox

Olivier FRANÇOIS
Prof. of statistics
TIMC-IMAG

Giovanni SQUILLERO
Prof. of Computer Science
DAUIN - PoliTo

DELFOX PREDICTIVE TECHNOLOGIES
Domaine PELUS PLAZA 16 Avenue Pythagore
33700 Mérignac FRANCE

May, 2021

Contacts

Antonin Poché
Role: Student
Email: poche.antonin@gmail.com
Phone: +33 (0)6 31 57 27 34

Xabier Jaureguyberry
Role: Delfox's supervisor
Email: xabier.jaureguiberry@delfox.net
Phone: +33 (0)5 35 54 37 29

Olivier François
Role: Ensimag's supervisor
Email: olivier.francois@univ-grenoble-alpes.fr
Phone: +33 (0)4 56 52 00 25

Giovanni Squillero
Role: Polito's supervisor
Email: giovanni.squillero@polito.it
Phone: +39-011090.7186

Abstract

Artificial Intelligence (AI) has developed tremendously in recent years, notably thanks to the advances in neural networks. However, the "black box" character of the latter has slowed down the diffusion of Deep Learning (DL) in the industry. Indeed, despite the in growing efficiency, neural networks still do not have the confidence of industrials. This is why explainability is a rapidly expanding research sector.

Delfox is working on Deep Reinforcement Learning (DRL) solutions for important industrial actors working in particular with critical systems. Explainability applied to Reinforcement Learning (RL) is therefore a key issue for Delfox and thus it is the focus of this internship. Explainability is still a recent field of research and there is no industrial application of such a technology known to date. Hence the challenge of Delfox is to make that happen, as they also need to show that its AIs are reliable.

This report presents an exhaustive bibliography, a taxonomy of both the eXplainable Artificial Intelligence (XAI) methods applicable to RL and the methods from eXplainable Reinforcement Learning (XRL). From this bibliography, three methods, namely Feature Relevance (FR), Observation Clustering (OC) and Probe Sensing (PS), have been selected, applied and studied onto one of Delfox's projects, they have been studied and applied on a project. This report introduces these three methods and discusses the results obtained and how they can generate complementary explanations of the decisions and behaviors of an Artificial Intelligence (AI) of RL.

Keywords: *Explainability, Interpretability, Explainable Artificial Intelligence, Reinforcement Learning, Deep Reinforcement Learning, Explainable Reinforcement Learning*

Abstract in French

L'intelligence artificielle s'est énormement développpée ces dernières années, notamment grâce aux avancées sur les réseaux de neurones profonds. Cependant, le caractère "boîte noire" de ces derniers a freiné la diffusion du Deep Learning (DL) dans l'industrie. En effet, malgré l'efficacité grandissante des réseaux de neurones, ceux-ci n'ont toujours pas la confiance des industriels. C'est pourquoi l'explicabilité est un secteur de recherche en pleine expansion.

Delfox travaille sur le Deep Reinforcement Learning (DRL) pour d'importants acteurs industriels travaillant notamment avec des systèmes critiques. L'explicabilité appliquée au Reinforcement Learning (RL) est donc un enjeu clé pour Delfox qui a motivé le travail de ce stage et la rédaction du présent rapport. Le DRL est un domaine encore jeune et il n'y à ce jour pas d'application industrielle d'une telle tehnologie, d'où le défi de Delfox et l'importance de montrer que leurs IA sont fiables.

Ce rapport présente une bibliographie complète, une taxonomie des méthodes d'eXplainable Artificial Intelligence (XAI) applicables au RL et des méthodes du XRL. De cette bibliographie, trois méthodes appelées Feature Relevance (FR), Observation Clustering (OC) et Probe Sensing (PS), elles ont été séléctionnées, étudiées et appliquées sur l'un des projets de Delfox. Ce rapport introduit ces trois méthodes et discute des réultats obtenus et comment elles peuvent générer des explications complémentaires sur les décisions et le comportement d'une IA de RL.

Acknowledgments

First of all, I am deeply grateful to **Dr. Xabier Jaureguyberry**, my supervisor at Delfox, for his insightful comments and assistance during my internship, for his availability and the time he took to guide me.

I would like to express my sincere gratitude to **Prof. Olivier François** and **Prof. Giovanni Squillero** my supervisors at the Ensimag and the Politecnico di Torino respectively. For their support, understanding, and enthusiasm about my project.

I would like to extend my sincere thanks to **Prof. Clovis Galiez** my tutor at the Ensimag for this double degree, for his teaching, his support, the inspirations he gave me, the insightful suggestions and guidance he provided me with.

I am deeply grateful to **Prof. Edgar Ernesto Sanchez Sanchez** my academic advisor at the Politecnico di Torino for this double degree, for unwavering support and patience in front of all the difficulties I brought to him.

I am deeply grateful to **Ms. Mairwen Perenon**, International Relations Officer at the Ensimag, and her predecessor **Ms. Berangere Voue**, for their unwavering support, patience, and the time they dedicated to supporting me during all the steps of my double degree.

I am deeply grateful to **Ms. Cécile Cozette**, Executive Assistant Manager at Delfox, for her accompaniment and assistance at every stage of this internship.

I would like to express my sincere gratitude to **Mr. Maxime Rey** and **Ms. Alice Memang**, CEO and COO of Delfox for their support, belief in me, and for this opportunity at Delfox that they offered me.

I would like to express my sincere gratitude to **Ms. Moufida Derbal**, internship administrator at the Ensimag for the support, patience, and the time she took to solve the problems of my double degree's internship.

I would like to extend my sincere thanks to all of the Delfox team. To **Mr. Léo Dupouy**, **Dr. Kévin Gravouil**, **Mr. Max David**, **Mr. Mathieu Prouveur**, **Mr. Clément Collgon**, **Ms. Marie Chevalier**, **Mr. Adrien Milcent**, **Mr. Alexandre Juppet** and all for the help and assistance they sincerely offer and their warm welcome at the company.

Finally, I would like to offer my deepest thank to two fellow students, **Mr. Adrien Thirion** and **Mr. Martin Herault**, for the inspirations, motivations, support, and insightful suggestions they made through my double degree project.

Contents

List of Figures

Acronyms

AI Artificial Intelligence. iii, 1, 3, 32

DL Deep Learning. iii, iv, 3

DRL Deep Reinforcement Learning. iii, iv, 1–5, 12, 15, 16

FR Feature Relevance. iii, iv, 43

GDPR General Data Protection Regulation. 4

OC Observation Clustering. iii, iv

PS Probe Sensing. iii, iv

RL Reinforcement Learning. iii, iv, 2–5, 8, 10–12, 15, 16, 43

XAI eXplainable Artificial Intelligence. iii, iv, 1, 4–8, 10–12, 15, 40, 41

XRL eXplainable Reinforcement Learning. iii, iv, 1, 4, 6, 7, 10–12, 15, 40, 41

Glossary

Explainability An explainable model is a model where feature contributions, the actions taken, and the decision process (at each step) can be understood. A model can be made explainable through other methods applied to this model. (explainability and interpretability will be used as synonyms). 2, 5

Explanation Additional meta-information, generated by an external algorithm or by the machine learning model itself, to describe the feature importance or relevance of an input instance towards a particular output classification. (from Das and Rad 2020). 9

Interpretability A desirable quality or feature of an algorithm that provides enough expressive data to understand how the algorithm works. (from Das and Rad 2020), (explainability and interpretability will be used as synonyms). 1, 5, 6, 11

Taxonomy The practice and science of classification of things or concepts, including the principles that underlie such classification. Originally used only for biological classification, taxonomy has developed to become synonym for classification. 1, 7, 8, 41

Introduction

Delfox is an AI-first startup which develops Autonomous Learning Systems based on Deep Reinforcement Learning (DRL). Its clients are prestigious actors of the aeronautics, spatial and defense industries (ASD) such as Thales, Ariane Group and Dassault Aviation. However, the infamous "black box" effect that deep neural networks (DNNs) suffer also affect DRL. Therefore, Delfox needs to prove that its AI can be trusted and are robust. One way to enhance this confidence consists in making their decisions explainable. The problematic of this thesis is thus the application of Interpretability methods to DRL in an industrial context.

The objectives of this thesis are therefore to justify and explain the decisions of the Artificial Intelligence (AI) developed by Delfox. This means: to explore the eXplainable Reinforcement Learning (XRL) literature, to try different methods and provides the means to easily use the relevant methods for Delfox. The idea is to propose a framework to automatically apply the methods to a trained AI from Delfox.

This thesis will present how were fulfilled those objectives. To provide the necessary elements to understand the work that was effectuated, the thesis will follow the following plan:

- Presentation of the context of this internship: the company, the scientific aspect and the relevance of this research.
- Description of the state of the art in eXplainable Artificial Intelligence (XAI) and XRL through the developed Taxonomy of those domains.
- Description of three methods that were applied to one of the project of Delfox. Each method will be explained extensively, then some graphics obtained with such method will be presented and analyzed.
- Conclusion of the thesis through the obtained results, the contributions of this internship and the perspectives.

1. Context

First of all, to understand what was done during this master thesis and why it was done, the context will be presented. The first point will be a presentation of the company, Delfox. In a second and third phase, the two scientific directions of the subject will be presented, (Reinforcement Learning (RL) and Explainability). Finally, a description of the objectives of this internship will conclude the context part.

1.1 Delfox

Delfox is an AI-first startup that develops Autonomous Learning Systems based on Deep Reinforcement Learning (DRL). Its unique technology, built upon state-of-the-art techniques (Schulman et al. 2017) and (Lowe et al. 2020), lead to great successes with prestigious actors of the aeronautics, spatial and defense industries (ASD) such as Thales, Ariane Group and Dassault Aviation. It was founded in 2018 and already counts fourteen persons.

1.1.1 Their Mission

Till now, Delfox was a company of service but it is now developing a product to expand its horizons. Traditionally, Delfox was responding to request for proposals on projects by big French companies. Those companies, which are interested in RL applications but do not have access to the related expertise internally, rely on such projects to study the benefit of RL in their own fields of application. Delfox first settled a strong partnership with Ariane for Space Situational Awareness and developed its RL expertise with Thales and Dassault Aviation on various other topics.

Today, while keeping its historical partnerships, Delfox is developing his first product a a aiming at allowing non-RL-expert to develop AI systems based on DRL.

1.1.2 The R&D Team

The R&D team is composed of nine persons, two machine learning doctors (including my tutor, the R&D team lead), five machine learning engineers and two machine learning interns (including myself). The whole team is specialized in Deep Reinforcement Learning (DRL), mostly thanks to their work at Delfox. Apart from my tutor who supervises all projects, team members usually work on one or two projects, including the product development.

1.2 Reinforcement Learning (RL)

Reinforcement Learning is an area of machine learning, at the same level as supervised and unsupervised learning. The RL is a change of paradigm compare to Deep Learning, here, the Artificial Intelligence (AI) learns by trial and error. There are several key elements in RL as depicted in figure 1.1. The elements are :

- **Agent**: The entity that represent the AI.
- **Environment**: The space or world within which the agent evolves or moves. The environment is often a simulation of the reality with several entities and a physic. But it can also be a board of chess.
- **Observations**: What the agent see of the environment, it could be an image or a list of positions. This is the input of the AI.
- **Actions**: How the agent interact on the environment, his decisions, it could to move forward or turn for example. Those actions are the output of the AI.
- **Reward**: How the environment evaluate each action, the way for the agent to learn if his decisions were good or bad. This is the process as in dog training were they receive a reward when they perform well or progress. This can be seen as the objective of the AI.
- **Policy**: A part of the agent, the decision process of the agent. This is a function that take observations as input and outputs actions. This is the part of RL that becomes a neural networks in DRL. This term will also be used to refer to the general behavior of the agent.
- **Reinforcement Learning Algorithm**: A part of the agent, the algorithm that update the policy based on the reward. This algorithm try to maximize the cumulative reward. There exist many different algorithms.

Another particularity of RL is the time. An agent is trained on episodes, an episode refer to the simulation of the environment during a given amount of steps or till the agent complete his objective. A step represent an iteration of the cycle represented in figure 1.1 (observations, actions, update of the environment, reward). Furthermore, if at each step the environment can be represented with an image, then the simulation of an episode can create a video. In this way, it is easy to see what the agent is doing in the environment.

There exist several algorithms considered as the foundation of RL:

- **Q-learning** is an algorithm based on the updating of a Q-function. This function gives the expected cumulative reward for an observation-action tuple, and is used by the policy to take decisions.
- **Policy Gradient** learns the policy directly, evaluating which state should be preferred. It then outputs a probability distribution on the possible actions.
- **Actor-Critic** is a combination of Q-learning and policy gradient. The actor is the policy, the algorithm that takes decisions, and the critic, a function similar to the Q-function.

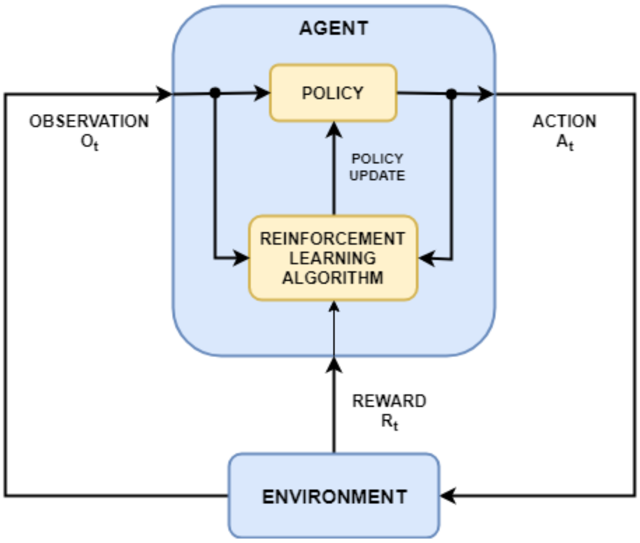


FIGURE 1.1: Reinforcement Learning Paradigm
Source: mathworks.com

This report will not go into further details on RL algorithm, two well-made crash courses being available here: Introduction to RL and Q-Learning and Policy Gradients and Actor-Critics. A complete dictionary (written by Shaked Zychlinski on towardsdatascience.com) is also available.

Deep Reinforcement Learning (DRL) is the Deep version of RL, where deep neural networks are used to model the agents. For example, Deep Q-learning is a Q-learning algorithm where the Q-function is approximated by a neural network.

1.3 Explainability

Nowadays, Machine Learning and Deep Learning techniques are progressing at an astonishing rate. Everybody now knows about the possibilities and potential of such methods. However, their use has been slowed down by a profound lack of trust in those methods, as they carry with them the heavy image of being black boxes, particularly with neural networks used in Deep Learning. Moreover, when using an algorithm to make predictions or take decisions on critical situations, legal problems may arise. For example, the GDPR set of laws (2018), introduces the right to explain, meaning that every person has the right to know how and why any algorythm decide. This creates the need for interpretable, certifiable, and accountable models and methods, which leads to a new field of research, eXplainable Artificial Intelligence (XAI), and in the same idea, eXplainable Reinforcement Learning (XRL).

XAI is a recent branch of machine learning, as most of the work on this topic has appeared after 2015. The aim of XAI is to provide tools to understand what deep neural networks do. The different type of methods will be described in the first part of the bibliography (see section 2.1).

In the literature, both terms can be found : Interpretability and Explainability being used as synonym. As no clear definition arises from the literature, both terms will be used as synonyms in the present thesis.

XAI methods can be applied with several purposes. It could be to justify that a decision or a group of decision was correct or to make a representation of the model that can be understood by humans. Nevertheless, all those purposes join in the scope of trusting the models. That is why, even if initial terms have a slightly different meaning, they are used in the same way in the literature.

1.4 Internship Objectives

The subject of this master thesis is post-hoc Explainability for Deep Reinforcement Learning. This section will introduce the context, the objectives and the missions I was in charge of during this internship.

1.4.1 Context

Delfox works for companies manipulating critic systems, those companies thus require models to be certified. For now, Delfox do not create RL-based agents that are used in real applications. What Delfox do is closer to proofs of concept. To enable the practical use of trained agents, Delfox is involved in :

- Proving that agents provide better solutions (more optimized and robust), than what humans or scripted algorithms can do.
- Convincing that agents can be trusted, hence the need for Explainability.

Moreover, Explainability will help improving Delfox's technologies, because, a better understanding of agents, makes it easier to adapt them and improve them.

1.4.2 The Objectives

The objectives of this internships are :

- Survey of the XAI scientific literature, particularly focused on post-hoc techniques methods,
- Choice, implementation and test of the most promising approaches,
- Adaptation to the specific context of deep reinforcement learning
- Application to real complex problems brought by our industrial clients.
- Propose tools that will allow Delfox to apply Interpretability methods easily in the future.
- This tool should aim to provide results interpretable by clients.

1.4.3 The Missions

To achieve those goals, the assigned missions are :

- **Bibliography**: List all possible solutions and select which one are relevant for Delfox. This was done through an exhaustive bibliography on existing XAI and XRL methods.
- **Application**: Apply those solutions on a project, and explore the possibility of several promising solutions. Using different types of solutions will provide complementary explanations and interpretations, leading to a more complete understanding of agents.
- **Communication and Visualization**: Those results need to be communicated. Hence, for each solution, a set of visualizations was provided to ensure that most of the information collected was accessible.
- **Documentation and Presentations**: To allow this work to be reusable complete documentation needs to be done. A documentation through an example of an extensive analysis and the means to interpret the results from the Interpretability methods. With the same goal, several presentations will be done to the R&D team, to present the methods and the results.

2. Bibliography

The aim of the bibliography is to explore existing solutions, understand them and select the interesting ones for Delfox. This bibliography and all the related documentation were done in the beginning of the internship in approximately 5 weeks. I began this bibliography with papers on the eXplainable Reinforcement Learning (XRL) field, but this field is even more recent than the eXplainable Artificial Intelligence (XAI) one. There were only one survey on XRL (Puiutta and Veith 2020), hence it was not possible to focus only on existing solutions in XRL. Therefore, an exhaustive literature on XAI was also performed to find methods that could be adapted to XRL.

To be able to present all the existing work to the team, the creation of a Taxonomy to classify XAI and XRL methods was necessary. Most of the XAI surveys proposed a Taxonomy for Interpretability methods, (Adadi and Berrada 2018), (Carvalho, Pereira, and Cardoso 2019), (Das and Rad 2020), and (Belle and Papantonis 2020). The majority agreed on three points that will be presented later in section 2.1.1. However, there is no agreed-upon Taxonomy considering the type, methodology, or principle of the methods. Moreover, no taxonomy was able to class every known methods and at the same time gives a hint on how the method is working. Therefore, creating a new taxonomy by merging existing ones was necessary (see section **??**).

After a description of the agreed-upon and the created XAI Taxonomy, this thesis will present the different categories of XRL methods in section **??**.

2.1 eXplainable Artificial Intelligence (XAI)

Making a Taxonomy is a difficult task which requires a complete understanding of the subject and a deep exploration of existing methods. However, thank to previously cited surveys, finding the related papers was easy and the proposed taxonomies have been used as inspirations. The three categories agreed upon by most surveys will be presented first and then the two proposed levels will be described. All studied papers on XAI methods are referenced in the appendix B.1.

The final taxonomy consist of 5 levels: (see figure A.1)

- **Specificity**: Model-Agnostic vs Model-Specific;
- **Application Time**: Intrinsic vs post-hoc;
- **Scope**: Global vs Local;
- **Type**: Type of the explanation;
- **Principle**: Mechanism of the interpretability method;

2.1.1 Agreed Upon Taxonomy

- **Specificity: Model-agnostic or Model-specific**, the specificity determines if the interpretability method is :
  - **model-agnostic**, *i.e*, a method that can be applied to any model or a large group of models.
  - Or **model-specific**, a method that can only be applied to one model or a smaller group of models.
- **Application Time: Intrinsic or Post-hoc**:
  - An **intrinsic** method is a method that needs to be built at the same time as the model itself is built. Those methods often require to deeply understand the model and to adapt the method to the precise model structure. Note that some models called transparent models are inherently interpretable.
  - **Post-hoc** methods are applied to a trained model or need to be trained at the same time as the model. They allow much more flexibility on the model choice.

  This category is closely linked to the previous one, as most intrinsic methods are model-specifics and most post-hoc methods are model-agnostics.
- **Scope: Global or Local**, the scope determines if an interpretability method aims at explaining a decision or the model globally.
  - A **global** XAI method is a method that tries to summarize the overall behavior of the model. One way is to make a simpler and easier to interpret model that will mimic the model to be to explained.
  - A **local** method focuses on a single decision, and tries to unveil the decision process for this decision.

Model-agnostic and post-hoc methods from XAI can be applied to Reinforcement Learning quite simply, for both local and global methods. However, model-specific or intrinsic methods can be much more complex to adapt. Therefore this bibliography and the following Taxonomy will focus on model-agnostic and post-hoc XAI methods. Furthermore, post-hoc methods are preferred by Delfox because they can be applied directly and do not need modifications of the RL algorithm. Besides, both local and global methods will be treated, note that, several local explanations may bring global comprehension of the model decision process.

2.1.2 Developed Part of the Taxonomy

The two developed levels of Taxonomy specific to model-agnostic and post-hoc are type and principle.

- **Type**: The types are four large category afterward divided in principles. They represent the kind of Explanation produced, while the principle describe the means to produce such Explanations The type is close to the proposition from (Adadi and Berrada 2018). It is less precise than principle as a type may bring together several principles.
  - **Simplifications** refer to the techniques that approximate an opaque model using a simpler one, which is easier to interpret.
  - **Feature Relevance** methods attempt to explain a model's decision (*i.e.* the output) by quantifying the influence of each input variable. This results in a ranking of importance scores, where higher scores mean that the corresponding variable was more important for the model.
  - **Example-based** methods extract representative instances from the training dataset in order to demonstrate how the model operates. They are local by definition.
  - **Visualization** methods aim at generating visualizations that facilitate the understanding of a model. They are local by definition, otherwise, the model is said to be transparent.
- **Principle**: The principle is about the mechanism used by the method to make the prediction. One clear example of the difference between type and principle would be back-propagation and perturbations which both aim at explaining features relevances while using different principle. The principle is a mix of proposition from (Das and Rad 2020), and (Belle and Papantonis 2020). Not all types will be described here, only the most relevant to be derived or used in XRL.
  - **Compressions** are global simplification methods that take an ensemble of models called the teachers, and a simpler (transparent) model called the student. The student is trained to replicate the teachers' behavior.
  - **Approximations** are local simplification methods, where the model's behavior is approximated around the studied data point, like in a Taylor development. See LIME (Ribeiro, Singh, and Guestrin 2016) as example.
  - Explanations generated by giving several slightly different input to a trained machine learning model and looking at the impact on the output fall in the **perturbations** principle. See Sensitive Analysis (SA) (Baehrens et al. 2010) and Occlusion (Zeiler and Fergus 2013) as example.
  - **Back-Propagation**, in contrast to perturbation methods, uses the backward pass of information flow in a neural network to understand neuronal influence and relevance of the input towards the output. The majority of gradient-based methods focus on either visualization of the activation of individual neurons with high influence or overall feature attributions reshaped to the input dimensions. See LRP (Bach et al. 2015) and Integrated Gradient (Sundararajan, Taly, and Yan 2017) as example.
  - **Prototypes** are example-based methods, therefore local methods. They provide data points (called prototypes) close (in the input space) to the studied one and have the same output as the decision that is studied here. The idea is to say, the output of X was

Y because those X′ are close to X and they also output Y. See MMD-critic (Lloyd and Ghahramani 2015) as example.

- **Counterfactuals** are example-based methods that show the closest data points that have different outputs. To give examples of how should have been the input to get a change on the output. See counterfactual (Wachter, Mittelstadt, and Russell 2017) as example.

2.2 eXplainable Reinforcement Learning (XRL)

Methods for XRL cannot be classified as strictly as methods for XAI. Indeed there are far fewer methods. Hence bringing them together is more complicated. Nevertheless, some general ideas can be isolated. For XRL, the three levels of taxonomy agreed upon for XAI can also be applied. Nonetheless, many methods will not be relevant for Delfox. All studied papers are referenced the appendix B.2.

In the bibliography, six groups of methods were represented:

- **Simplification**: This type is the same as the one in XAI, but here it can be applied to several parts of the RL algorithm. This class represents close to half of the studied papers. It can be applied to the policy, the actor, or the critic (for an algorithm that possesses either of them). Either of those functions can be simplified in a more interpretable function. Finally, observations space can also be simplified, such a method was applied during this thesis and will be described in section 3.3. See distillation (Rusu et al. 2015) as example.

- **Feature relevance**: It is the study of the influence of each observation n the actions. There was no paper on this subject in the XRL literature. But this type is the easiest to derive from XAI to XRL, and those methods could be applied to the policy, the actor, or the critic. (see methods from XAI, section 2.1.2).

- **Transparent models**: Model that are by construction interpretable are either really simple or there structure provides intuition on the model functioning. Hierarchical RL (Shu, Xiong, and Socher 2017) and Reward machine (Icarte et al. 2018) fall in this category. There are also methods to apply when building the model, but those methods may be difficult to apply for Delfox because they use existing libraries.

- **Interestingness elements**: Represented by only one paper (Sequeira and Gervasio 2020), it is a method which extracts elements from episodes simulation. Those elements gives insight on the agent behavior, what are usual sequence actions, what part of the observation space is never explored...

- **Autonomous explanations** are natural language explanations, developed for interaction between working robots and humans. This explanations can be seen as the robot directly explaining to a human what he is doing. For example company, to the question: "Why are you here?", the robot could answer: "I am going to the stock, but the main way was blocked.". See Autonomous Explanations (Hayes and Shah 2017) as example.

- **Neurons activation**: This is a group of method where the activation of neurons is studied. There are no paper specific one of those methods for XRL, but such method was used in a paper (Jaderberg et al. 2019) from DeepMind to analysis their agents. Besides, a method from this group was applied during this thesis, the probe sensing (see section 3.4).

In RL, in an episode, decisions are ordered this allows a time related analysis in XRL that was not available in XAI. To perform this analysis, tools specific to XRL or complex adaptation of XAI methods are necessary. For instance, if explanations from the application of local XAI methods on one decision are grouped together, it can provide global insight.

Many RL algorithms have a policy divided in an actor-critic pair, the actor decide on the action and the critic estimate how good are each actions. (more detail on Policy Gradients and Actor-Critics course). In this thesis, the interpretability methods have been applied to the actor, but they could have been applied to the critic.

In XRL as in XAI, several different Interpretability methods should be applied. They are often complementary, one method can neither work for every model nor give every possible explanations.

3. Application

After the bibliography, the next step was to select some methods, choose one or several projects and apply the selected methods to the project. The chosen project is called Heaxplain and it is introduced in section 3.1. The methods were applied to only one project because it allows more extensive researches on each methods. Each applied method will be presented after the introduction of Heaxplain. The selected methods are :

- **Feature Relevance** (section 3.2): it is a group of methods adapted from XAI. The idea is to study the influence of the observations on the actions. The concept is intuitive and natural even for people not familiar with interpretability or Deep Reinforcement Learning, hence for example, for Delfox's clients. It highlights the important factors for a decision. This kind of methods brings many information and enables effective representations.
- **Observation Clustering** (section 3.3): this method is specific to XRL, the idea is to divide the observations spaces into sub-spaces where the agent behavior is comparable. This method is quite intuitive and the results are easy to understand.
- **Probe Sensing** (section 3.4): this method is also specific to XRL. The idea is to see what are the information that the model can compute and have access to in order to make a decision. This method is far less intuitive and is difficult to explain but the results provide information on the representation the agent have of the environment.

3.1 Heaxplain

The name Heaxplain refers to a project conducted in collaboration between 4 companies, the one that created the project, a company that formulated the physical constraints, Delfox that developed the environment and the RL models and the company that worked on the interpretability of the RL agent behavior.

3.1.1 The Project

The problematic is a capture-the-flag game between two planes. Each plane have his side, begin in his camp and need to capture the flag and bring it back. The agents have missiles and can destroy each other. The figure 3.1 show an image extracted from the video of an episode of Heaxplain. On this image, the elements aforementioned can be seen, the two agents are the red and blue arrows, the flag is the green square in the middle, the red and blue lines delimit the sides or camps and the red point is a missile fired by the red agent. The small dots represent the trajectories and the arrows in front of the agents show there directions, those two elements are only visual elements and do not exist in the simulation.

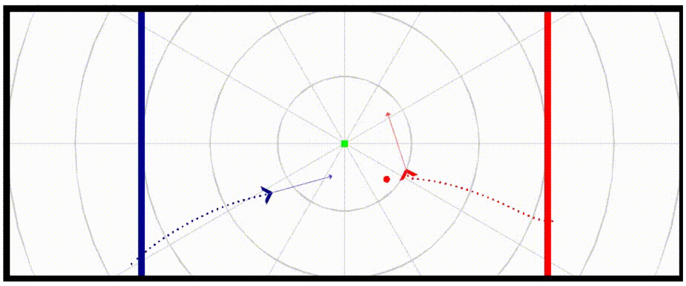


FIGURE 3.1: Image Extracted from a Video of Heaxplain

3.1.2 Reinforcement Learning in Heaxplain

In the Heaxplain environment is a simplified simulation of the reality where planes fight and try to capture a flag.The agents are the planes but for one agent, the other agent is part of the environment. The observations are what the agent see of the environment and the actions are the way the agent interact the environment. In Heaxplain, there are 19 observations and 4 actions.

First, the observation : (position and speed have two elements and the direction have three)

- Global observations
  - the side of the agent
- Observations on the agent itself
  - position
  - speed
  - direction
  - number of missiles left
- Observations on the enemy
  - position
  - speed
  - direction
- Observations on the flag
  - position
  - if the flag has been captured

Second, the actions:

- **Gas** (Gaz): The intensity of the acceleration (a negative value serve to break)
- **Turn** (Virage): The intensity of the turn (the directions right or left are decided by the sign)
- **Fire-range distance** (D_tir): this distance delimit a zone (the fire-range) around the agent, if the enemy is in this zone and the agent still have missiles, the agent fires a missile.
- **Number of missiles** (Nb_msl): The number of missiles that should be fired, used to decide if the agent should shoot or not.

3.1.3 The Reasons To Select Heaxplain

The reasons to choose Heaxplain are multiple :

- The project was already finished before the thesis. Therefore, the agent was already trained and efficient. There was only one agent to analyze.
- As aforementioned, interpretability was an important part of the project's scope and has thus been already studied. However, the method that has been used to explain the models is patented and does not belong to Delfox. The results from this study were available and can serve has a baseline for for Feature Relevance (see section 3.2) and Observation Clustering (see section 3.3).
- This project is well balanced. It is complex enough to have interesting behavior to be interpreted and simple enough to be interpretable and to allow effective visualization of the interpretation.

3.2 Feature Relevance

Feature Relevance (FR) is the first kind of methods that has been studied. Those methods were developed for XAI and can be applied to neural networks. However there are neural networks in Deep Reinforcement Learning. Therefore FR methods are naturally transportable to XRL. For those methods, the python's library DeepExplain was used.

This section section will first present the necessary elements to understand the FR methods : The concept, the objectives of the use of such methods, the different methods, the different possible levels of study and the pipeline. After this presentation, this section will present two different axis of analysis. Those axis are linked to the levels that will be presented beforehand.

3.2.1 The Concept and Objectives

The concept

Feature Relevance methods aims at computing the impact of the input (features) on the outputs for one decision. This impact is called relevance of a observation for one action, the terms relevance and influence will be used as synonyms. There are two principles of FR methods :

- **Perturbation**, methods that analyze the consequences of the modification of one or several features on the output.
- **Back Propagation**, methods where influences are computed by Back-propagation. The value of the actions are back-propagated using the model weights to get the relevance of the observations. (for further explanations and examples, see section 2.1.2).

In the case of RL as aforementioned in section 1.2, the studied model is the actor, the observations are the inputs and the actions the outputs. In Heaxplain there are 19 observations and 4 actions, therefore, at each time-step, a FR method will produce $19 * 4 = 76$ relevances.

The Objectives

Feature Relevance (FR) methods are based on an intuitive reasoning. The first thing a human would ask to understand a decision is what where the elements that influenced the choice. Then he would ask how much did each factor influence a decision and if it was negatively or positively. This is exactly the answers that FR methods provides. The primary objective of such methods are thus to answer those questions.

Those objectives are the initial objectives inherited from XAI, those objectives are naturally kept. However, with the time relative analysis, FR methods can provide much more complex information and explanations. The analysis of a sequence of actions will lead to the analysis of a behavior. The new objectives of such analysis is the analysis of patterns in the behavior, the study of phases in the behavior or comparison between episodes.

The objectives of having such information is dual. First, for the clients of Delfox, the results of those methods are part of the information they are looking for. They can easily give sense to what they see and have a natural interpretation of such results. Secondly, for the Machine Learning Engineer (MLE) that is building the agent. The influences highlight the utilities of each observation and if the MLE detects illogical relevances, it allows him to make improvements or to discover interesting behaviors.

The methods

There exist many different methods for computing feature relevances, each one of them having its particularities. They have advantages and disadvantages, but none of them stood out. There were four methods used :

- **Layer-wise Relevance Propagation (LRP)** (Bach et al. 2015) : a back-propagation method.
- **Integrated Gradient** (Sundararajan, Taly, and Yan 2017) : a back-propagation method.
- **Occlusion** (Zeiler and Fergus 2013) : a perturbation method.
- **SHAP** (Lundberg and Lee 2017) : a perturbation method.

The Levels

Initially, those methods are local methods, meaning they explain one decision. However, several local explanations can bring global insight on the decision process. In RL it is possible to concatenate the relevances for each decision in an episode. The order of the concatenation is the time order. This create another level of study of Feature Relevance applied to DRL. There also exist other levels of study. Below is a list of the possible level of study :

- The **level of a decision** is the necessary first step to build the other level. But a comportment is a sequence of actions, hence this level only allow limited explanations and it will not be presented in this document.
- The **level of an episode** is the concatenation of the method applied to all time-steps of an episode. It will be describe in section 3.2.2.
- The **level of an ensemble of episodes** could be seen as the mean of the relevances over several episodes. At this level, relevances are still arranged temporally and this level will also be presented after. This level will be called the tendencies. It will be presented in section 3.2.3
- The **level of an ensemble of decisions** (not temporally linked) this level compare the distribution of the value of each observations and actions. The information provided by this level

are not meaningful alone and their analysis is more complicated to explain than the two previous levels. Thus, it will not be presented in this thesis. However, this level allow further analysis if used after the two previous ones.

The Pipeline

One of the objectives of the internship was to build a framework to allow the RD teams to use interpretability methods easily. This framework uses a trained model to simulate episodes and then apply the following pipeline to those episodes to generate visualizations. The visualization produced by each level of study are separated. As depicted in the figure 3.2, he pipeline consist of the following steps :

- **Load the model**
- Use the model to **simulate episodes**
- Use the episodes to **compute densities**, (level of an ensemble of decisions).
- Use the episodes and the model to **compute the relevances**, the relevances are computed with the library DeepExplain, what is called relevances here is a matrix of size $19 * 4$ of the relevances, (level of a decision).
- **Aggregate the relevances** from the level of a decision to get the level of an episode, the concatenation gives a matrix of size $76 * totalnumberofsteps$.
- **Compute tendencies** from the level of an episode by computing the mean of several episodes for each step. The matrix size is still $76 * totalnumberofsteps$. (level of an ensemble of episodes).
- From all levels, make visualizations.

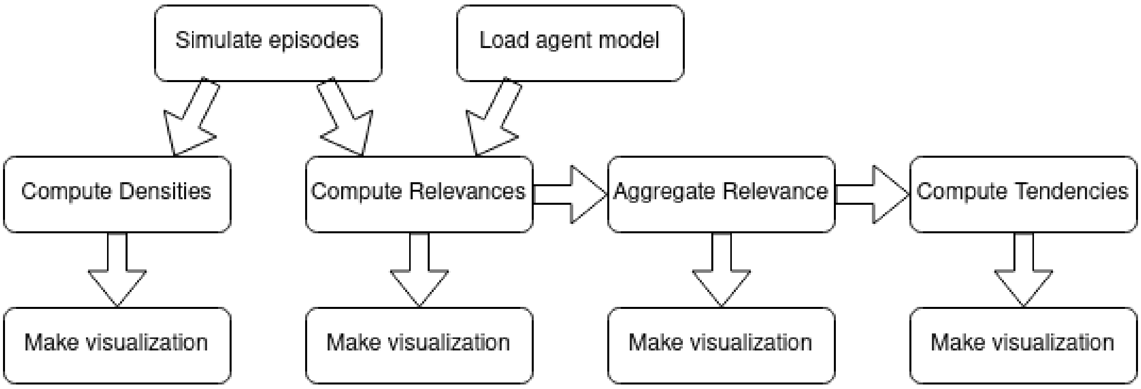


FIGURE 3.2: Feature Relevance Pipeline

The pipeline works for every method presented before but a more robust result was obtain with the mean of each methods. Therefore, to computation the relevance, the first step is to compute the relevance for each method and then take the mean mean of those methods. The output is still 76 relevances for each step and episode.

3.2.2 Study at the Level of an Episode

The study at the level of an episode is the study of an ensemble of decisions (actions) ordered temporally, (all the actions of one simulation). A sequence of actions represent the behavior of the agent in a given situation. This study allow to reveal the important observations for an agent depending on the step. This level allow to study the behavior of the agent through an example. It is the easiest to understand and the most natural level, it makes this level very interesting for Delfox when thy need to provide explanations on their agents behaviors.

For this part, only one graphic will be shown, the figure 3.3, this graphic provides all the information contained by the matrix of relevances as it is a heatmap. There are 4 heatmaps because the influences are computed for each action. In those heatmaps, each line represents an observation and each column represents a time step. This means that a line represents the evolution of the relevance of one observation on one action through time in one episode. Red and blue pixels represent high influences and gray pixels, no influence. Blue pixels represent negative influences. (For reminders on Heaxplain, see section 3.1)

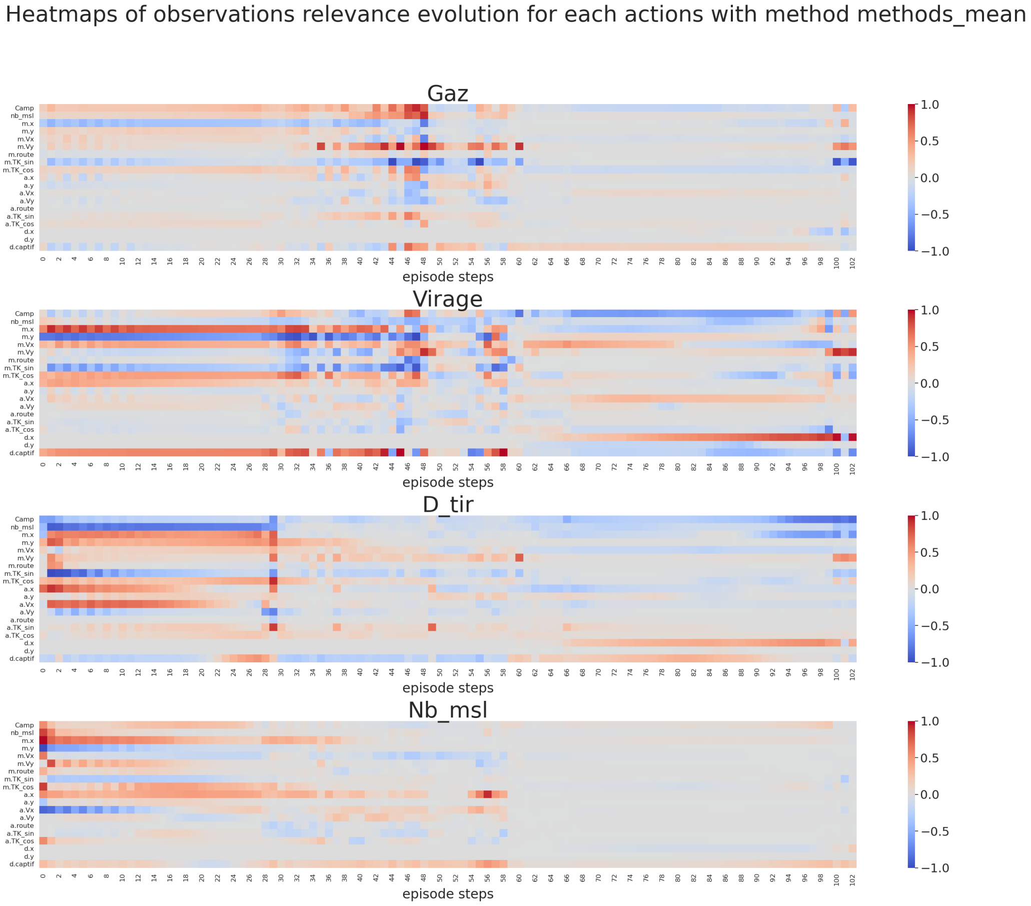


FIGURE 3.3: Heatmap of Feature Relevance Throughout an Episode

The figure 3.3 reveals phases in the comportment of the agent. A phase in the comportment is determined by a clear change in the relevances at a given step for at least one action. In this episode, 3 main phases appear. With information from the information and with the video of the episode, it is possible to characterize those phases :

- **Approach**, steps 0-30 : The agent goes from his camp to the flag.
- **Fight**, steps 30-60 : The enemy enters fire-range and the agent fires his missiles. The agent is still looking for the flag.
- **Return** steps 60-100 : The agent has just captured the flag and is coming back to his camp.

This graphic is interesting when it is put in parallel with the evolution of the observation and action values. In this case, the value of the action D_tir is 0 (minimum) till the step 30 and 70000 (maximum) after. On the figure 3.3, the relevances of action D_tir are negligible after the step 30 (too small). Therefore, the value 70000 seems to be the default value for this action. Hence what is important to explain is the reason for the diminution on the 30 first steps. In the case of this method, this explanation is to give the observations that have a negative influence on the value of D_tir during the first 30 steps. Those observations are here the number of missile and the direction of the agent. An interpretation of those explanation could be : *The agent still have missiles and it faces the wrong decision. Therefore it decides to wait before firing the missiles. Reducing D_tir is a way to wait before firing.*

3.2.3 Study at the Level of an Ensemble of Episodes

The study at the level of an ensemble of episodes, also called tendencies, consist in the merging of the relevances of several episodes. The scope of this level is not to analyze the behavior of the agent but to see if what was analyzed on one episode can be extrapolated to an ensemble of episodes, *i.e* discover tendencies in the behavior. The transitions between phases in the behavior will be fuzzy in the heatmaps, because the initial position of the agent is random and the approach phase may be longer. Another application of this level of study is to compare the behavior of the agent depending on the environment parameters. For example in Heaxplain, it is possible to change the starting side of the agent or the type of enemy it faces.

The plots that can be exported from tendencies are the same as in the level of an episode. Therefore the figures 3.4 and 3.5 are also heatmaps. It will be easier to compare and to understand. For this level, each line will not be analyzed precisely, the interesting information come from the comparison between heatmaps. There are two kind of comparison possible :

- The comparison of the heatmaps of relevances between tendencies and one episode (where the episode could have been in the episode batch of the tendencies). This comparison allow to see if the behavior of this episode is representative of the policy (global behavior).
- The comparison of the heatmap of relevances between two tendencies with different environment parameters. This comparison will provide information on the impact such parameters have on the agent behavior.

The figures 3.4 and 3.5 both represent a batch of 100 episodes. The figure 3.4 correspond to a start of the agent on the left and the figure 3.5 correspond to a start of the agent on the right side.

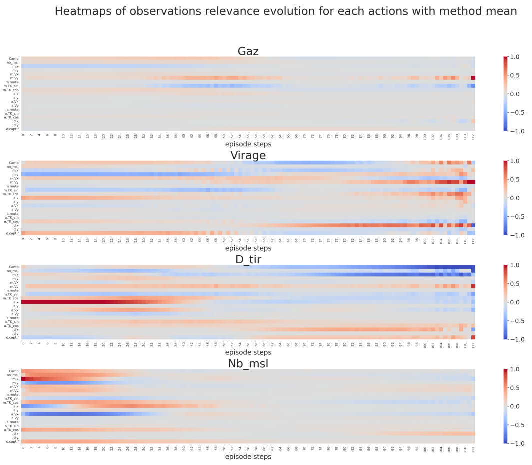


FIGURE 3.4: Feature Relevance Left Tendency Heat-map

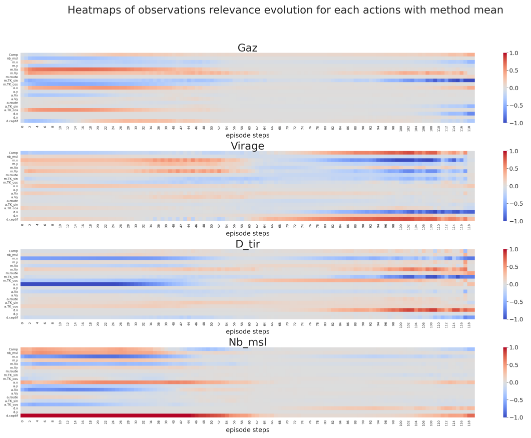


FIGURE 3.5: Feature Relevance Right Tendency Heat-map

The first kind of comparison aforementioned is possible between the figure 3.3 from previous level and figure 3.4. Indeed, the figure from previous level describe an episode where the agent started from the left side. This comparison indicates that the episode is a particular because the phases in D_tir are far less clear. This affirmation is verified with the comparison of the action distributions, in fact, with a start from the left side, the D_tir value usually stay at 70000 during the whole episode. The episode that served as an example for previous level was selected because the described behavior was easy to see and to explain. Nevertheless, the other part of the heatmaps are quite similar, this indicates that in this episode, a particular situation led to a particular behavior and it may be interesting to deepen the analysis on this situation. There are many other things that could be interpreted from those graphics, this was just an example.

The second kind of comparison is also possible, this time between the figures 3.4 and 3.5. Apart from the three phases that appear there seems to be no evident similarity. In fact there are some common features because depending the side, some features may have a different sign and this sign impact the sign of the relevances. Hence positive relevances for the left figure may indicate a similar behavior as negative relevance in the right figure. Even so, this rectification do not justify alone the differences, therefore, it can be said that the policy depends on the starting side and is not symmetric.

3.3 Observation Clustering

Observation Clustering is the second method that was applied. This method was inspired by what was done by the other company that worked on explainability on Heaxplain. The objective was to provide comparable information, but the process presented here is completely different from theirs. This section will first describe the Concept and objectives of this method, then the process used to build such a method (the clusterings and the merge of those clusterings). Finally, the results of this method will be described through an example.

3.3.1 Concept and Objectives

This section will present the idea behind observation clustering, the objectives or the form of results expected from this method and the pipeline used to process the data, build the method and export the visualization.

The Concept

The Observation Clustering method group observations together in super states where the agent have the same comportment. The number of expected clusters is not known, but analyzing different number of clusters may bring different information. This method focus on the possible observation space and divide it in sub-spaces, those sub-spaces are the clusters.

The Objectives

- Simplify the observation space, group observations that the agent consider as similar together. Understand how he agent see the environment.
- Simplify the study of the behavior, the analysis of several well delimited behaviors is easier and more relevant than the direct analysis of the global behavior. Understand the phases of the comportment and make a temporal study of behavior clusters chaining. See if there are one or several global behaviors and what make the agent choose one or the other.
- Produce of a Markov chain that represent the agent global behavior with the transitions between each behavior cluster. This is easy to interpret for a client.
- Produce several complementary graphics to provide information in parallel of the Markov chain.

The Pipeline

As shown in the figure 3.6, there are three main steps in the pipeline, each one of those are separated in two sub-steps:

- The generation of the data for the clustering:

  - Retrieve data from the simulation of the episodes (the observations and actions).
  - Compute the feature relevances with the method presented in section 3.2.
- The creation of a clustering based on the data from episodes and the feature relevances.
  - Make several clusterings on various part of the data with different number of clusters required. (presented in section 3.3.2)
  - Merge those clusterings to get a final one for each possible final number of clusters. (presented in section 3.3.3)
- The analysis of the clustering, this part uses the clustering and information directly from the data:
  - Extract the information.
  - Export Graphics built from the gathered information.

The clusterings made on one dataset were not satisfactory no matter the clustering method, the hyperparameter optimization and the dataset. They were not robust, the most interesting clustering was never created with the same parameters. Therefore a the idea to merge several clustering emerged.

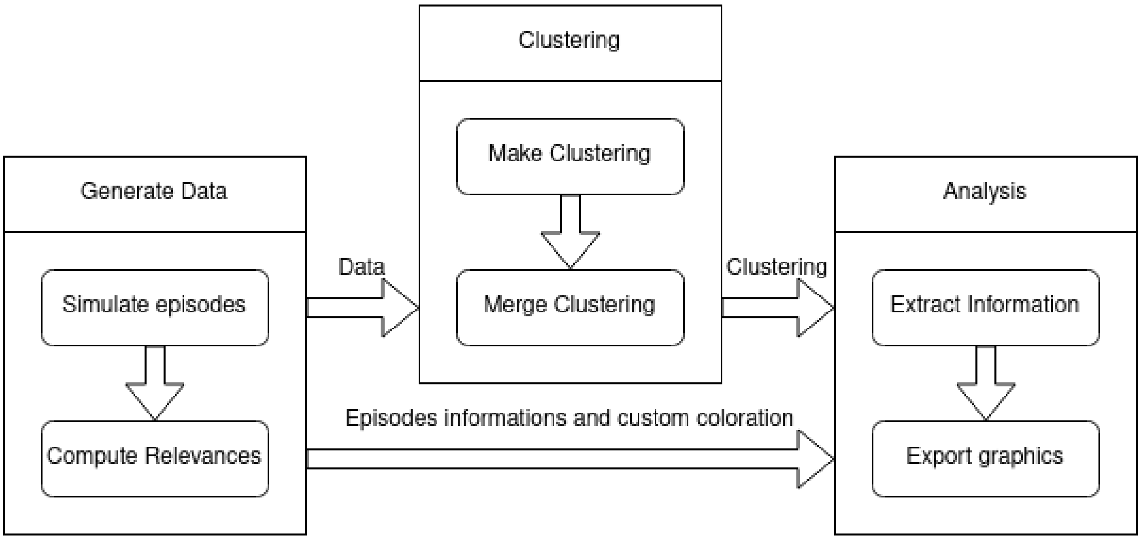


FIGURE 3.6: Observation Clustering Pipeline

3.3.2 Clustering

This section will describe how to build the different clusterings that will be merge. To do this, the explanation is divided in three parts:

- Details on the choice of the clustering method,
- Further information on the necessity to merge several clustering,
- Description of the parameters of the different clusterings that will be generated.

The Clustering Method

The chosen clustering method is Gaussian Mixture model, many known clustering methods from the Scikit-Learn library were tried, but Gaussian Mixture was by far the most interesting. No matter the hyperparameters used, the other methods created one huge cluster most of the time. Therefore, with a manual analysis of the clustering and thanks to metrics, Gaussian Mixture always appeared better. This method was therefore selected and it was the only one used.

To do the hyperparameter optimization, the Silhouette metric was used. The Silhouette metric have the advantage to be computed only with the clustering and the data, this way the learning is completely unsupervised. The Davies Bouldin Index was also tried, but the Silhouette metric produced more robust results and most of the time clusters with comparable size. In fact, the process is not completely unsupervised, the clustering was tuned to find "interesting" results. Therefore, the clustering may be biased, one way to verify this would be to try this method on other projects.

The Necessity to Merge

It was mentioned several time before, but none of the hyperparameters setting or method had robust results. Therefore, a solution with interpretable results was needed. Some clustering created interesting separations between observations but there were always some too small clusters and it was never for the same set of hyperparameters or dataset. The expected clustering was a mix of several of those clusterings and boosting models proved their efficiency in classification. Therefore, a method to merge several clustering was developed. The particularity of this method is that it allow to merge clustering without using the features. This is useful here because the interesting clustering were done on several different datasets.

The necessity to merge is in fact a necessity to find a more robust solution and the merge of clustering is a solution, the solution used here. But before the merge of clustering, not all clustering should be merge, this lead to the selection of the different interesting clusterings.

The Different Clusterings

The different clusterings come from a combination of two parameters, the dataset used and the number of clusters in the clustering. The number of clusters go from two to five clusters (4 possibilities) and there are seven datasets (one for the observations, one for the actions and one for each feature relevance method.) This leads to the creation of 28 different clusterings.

For each one of those clusterings, there is an hyperparameter optimization and the best hyperparameters are selected with the Silhouette metric. This leads to the computation of several hundreds of clusterings and it take some time, but it brings satisfying results.

3.3.3 Merge of Clusterings

After the creation of several clustering, the merge can be done. This method is divided into three parts, the creation of minimal clusters, the merging of those minimal clusters and the selection of the best merge.

Minimal Clusters

The method is based on two hypothesis :

- Two points (observations) are in the same cluster if they were clusterized together in all the clusterings.
- The number of time two minimal clusters were clusterized differently is a distance.

Do I need to prove that it is really a distance ?

The minimal clusters are the clusters formed following the first rule. The points in each minimal cluster were clusterized together in all clusterings. Hence it is possible to have clusters with only one point and a cluster with more than 90% of the points if all clusterings were unbalanced in the same way.

Those minimal clusters can be seen as the clusters on which all clustering agree, there is no doubt that those points should be together. Then the distance between minimal clusters can be seen as how much do clustering disagree on the fact that two minimal clusters should be merged. For example, a distance of 1 between two minimal clusters means that only one clustering separated those two group of points. Therefore, for the merge of those minimal clusters, the merge will be done with the smallest distance first, (hierarchical merge). A representation of an example of minimal clusters in shown in the figure 3.7.

Merge of Minimal Clusters

The merge of the minimal clusters produces a new clustering with the desired number of clusters. However, as aforementioned, the exact number of clusters is not known and several number of clusters can be interesting and give different interpretations. Therefore, several clustering are created, one for each number of clusters between 2 and 8 (2 and 8 are only parameters and are easy to change). Each one of those clustering is then automatically analyzed and the corresponding plots are exported.

The merging method is hierarchical clustering where each minimal cluster is treated like a point. The applied hierarchical clustering method was from the Scikit-Learn library. This method output clusters of minimal clusters, then the level of minimal clusters are forgotten and the clusters of minimal clusters are transformed in normal clusters.

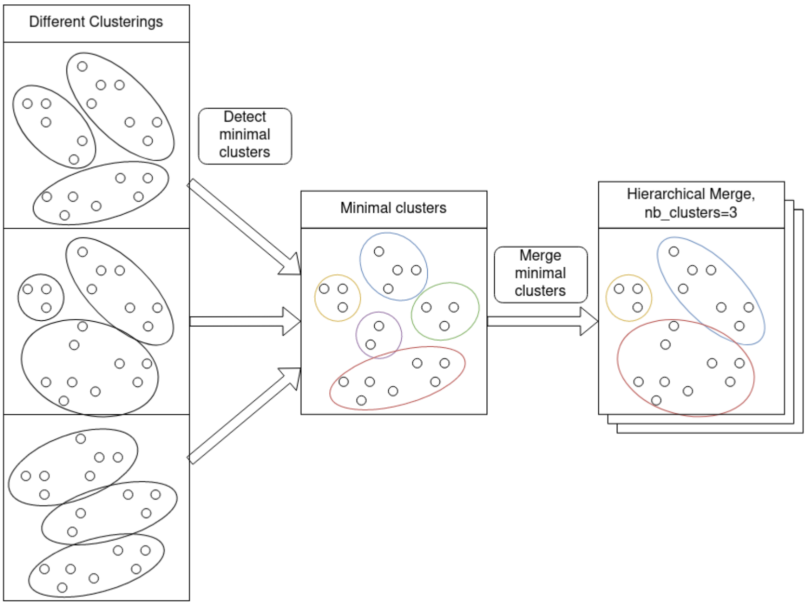


FIGURE 3.7: Observation clustering, minimal clusters

The Problem of too Small Clusters

This method produce results that are far better than what I could get with only one method. But there is still a problem, this method do not prevent output clusters from having a size of one observations. Clusters with really small size are not a problem themselves, but they make the analysis more complicated. Small clusters represent rare and specific behaviors, therefore, they hinder a first pass analysis. In fact, most of the time those clusters were represented by one point that was an outlier.

Therefore a method that will limit the minimum size of clusters was needed. But this method should also allow small clusters if they represent a specific behavior and not outliers. (The idea to make a cluster with outliers was not treated).

A new solution was developed to solve this problem. It was to merge the small clusters before the closest clusters. The small clusters are determined with a minimum size under which, each minimal cluster is considered too small. Then, it will be merged with the closest cluster (the resulting cluster could still be under the minimum size.) The new distances between clusters were computed with a weighted mean (the weights are the size of each clusters). After this first merge, the same hierarchical clustering as before was applied.

Several minimum size were tried, but for each final number of clusters, the best minimum size was different. Therefore, for each final number of clusters the best minimum size was selected with the Silhouette metric. The clusterings with the minimum size parameter to one correspond to the initial merge of clusterings, (this size was also included). The figure 3.8 is a representation of this method, the initial process is also included because it is used.

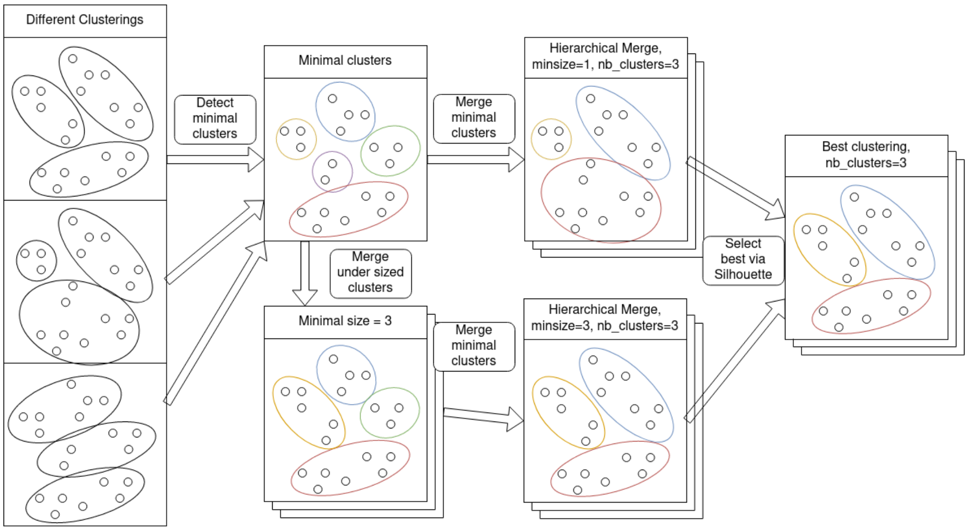

Different Clusterings
Detect minimal clusters
Minimal clusters
Merge minimal clusters
Hierarchical Merge, minsize=1, nb_clusters=3
Merge under sized clusters
Minimal size = 3
Merge minimal clusters
Hierarchical Merge, minsize=3, nb_clusters=3
Select best via Silhouette
Best clustering, nb_clusters=3

FIGURE 3.8: Observation clustering, minimal clusters with minimum-size

Finally, the clustering was satisfactory, every selection was made automatically with the Silhouette metric. (An example is available in the Appendix D). There were only interpretable clusters in the final clusterings. The number of expected clusters could be selected manually. (the possibility to output only the most interesting final number of clusters was not explored.)

3.3.4 Results and Interpretations

This section will present the clustering results through an example. All the following graphics were generated from the same clustering, on the same batch of episodes. Only a part of the available graphics will be presented, those plots are complementary and should be interpreted together.

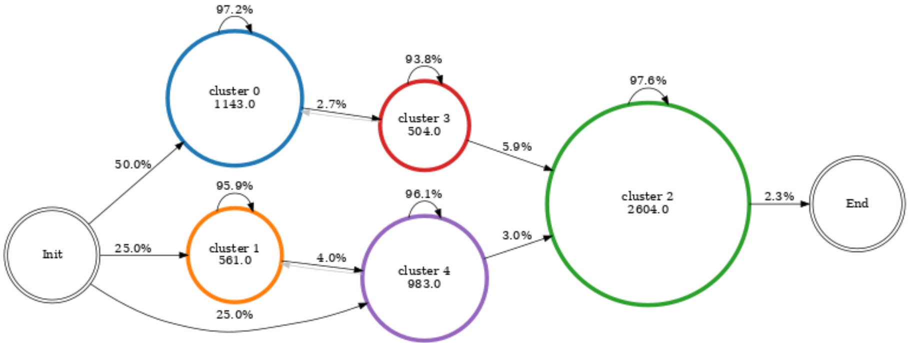
The results were obtained with the processing of 60 episodes. In those 60 episodes, half correspond to the right side and the other half to the left side. (Th agent can begin on the left or the right side). There are also three possible enemies that can be faced, 20 episodes were simulated for each enemy. Hence for each pair side-enemy, there are 10 episodes. This number is quite low, but it does not impact significantly the clustering.

The example that will be treated is a clustering with five final clusters. This number of clusters allow balance between interpretability and complexity. More complex graphics would make the interpretations too long and this example provide interesting results. For this example, three plots were selected, the Markov chain, the distribution of clusters through time and the spacial distribution of clusters. Each one of those graphics will be described and analyzed.

Markov Chain

In the Markov chain, the states represent the clusters, the size of the states represent the number of steps that were assigned to the corresponding cluster. The transition between two clusters describe the probability of going from one cluster to the other. It is calculated taking all pair of consecutive steps. Here, most of the time, two consecutive steps belong to the same cluster.

There are many possible legends that could be applied to the Markov chain, the size of states and transitions could be associated with many different values. Furthermore, states can also be colorized with a pie-chart, this provides information to interpret the states. The first Markov chain (3.9) is not colorized because it is easier to understand when this graphic is seen for the first time. However, the second one (3.10) is colorized, this coloration was done arbitrarily based on the phases discovered with the Feature Relevance method (see section 3.2). Thee coloration correspond to a manual clustering, and the pie-chart show the portion of the steps in one clusters that are associated to the manually created clusters. This coloration could be seen as the objective of the clustering but it was used to verify if the clusters made sens.



Markcov-Chain for 60ep_b_Mix_merge_5c discarding transitions with p<0.01 and sized by cumulative

FIGURE 3.9: Markov chain from the 5c-clustering

The analysis of those Markov chains will begin with the interpretations that can be done without the coloration. Then the interpretations allowed by the coloration will complete the analysis:

Interpretations from the simple Markov chain:

- The transitions between states have low probabilities. Which means that **clusters are stable**, the clusters represent behavior and the agent do not oscillate between behaviors.
- Each state only have two output transitions, itself and another cluster. Which means that **the order of the comportment phases is fix**.
- There are three different states between the *init* and the *end* states. Therefore, **the global behavior an agent can be divided in three phases**. It could be more, but with five clusters, this is what comes out. This reinforce the conclusion made with the Feature Relevance method.
- There are three possible sequences to go from the *init* state to the *end* state. Hence, there are **two or three global behaviors**.

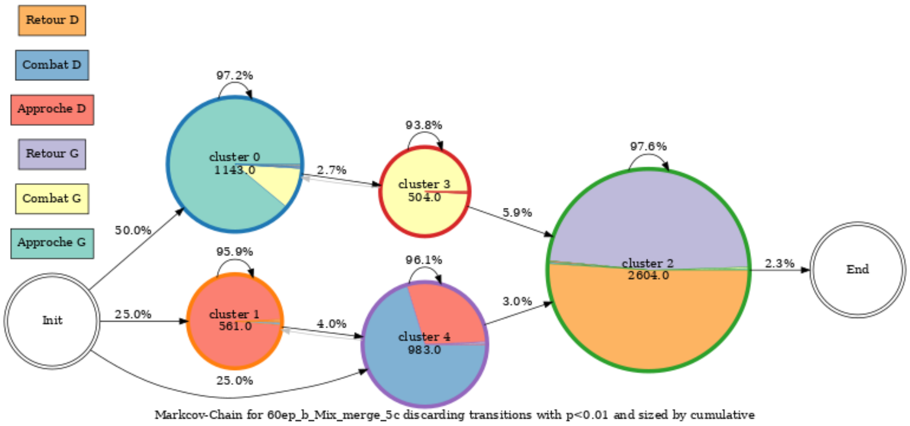


FIGURE 3.10: Markov chain from the 5c-clustering, arbitrarily colorized

Interpretations from the colorized Markov chain:

- The three big phases are Approach, Fight and Return.
- The two behaviors are linked to the starting side.
- The comportment in the return phase is similar for both starting sides.

Distribution of Clusters Through Time

The distribution of clusters through time show the distribution of episodes between cluster at each step. The x-axis represent the time in steps. There are 60 episodes therefore the maximum number of episodes associated to a cluster at a given step is 60. The sum between all clusters should also be 60 episodes. The length of the episodes is not fixed, it goes from 85 to 120 steps, that is why the green curve starts decreasing after the step 85. The colors and the number associated to each cluster is the same for all graphics, this allow an easier accumulation of information between plots.

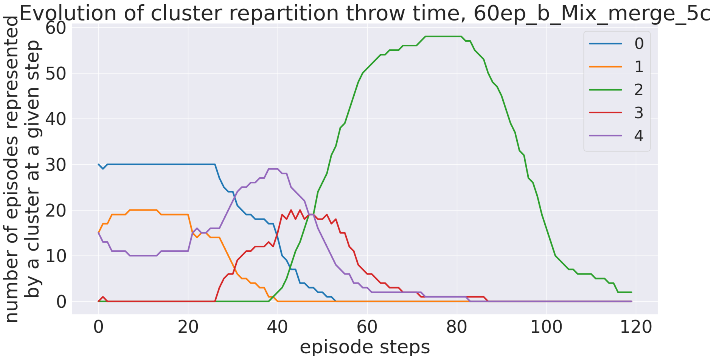


FIGURE 3.11: Distribution of clusters through time

With this graphic, some affirmations from the previous graphic are confirmed. The cluster 0 and 3 are complementary, there sum is 30 (all the episodes corresponding to one side). The same can be said for the clusters 1 and 4. The two groups do not sum to 30 anymore after the 40th step, (40 is the shortest time needed for the agent to capture the flag, start of the return phase). This graphic does not bring many information in comparison to the previous one. However it is much easier to read and allow a faster representation of the different behaviors and phases for someone new to those graphics.

Spacial Distribution of Clusters

The spacial distribution of clusters show the position of the agent at each step for the 60 episodes. There is one subplot for each cluster and the last one is a superposition of all the others. The lines in those plots correspond to parts of the agent trajectories that are associated to a cluster.

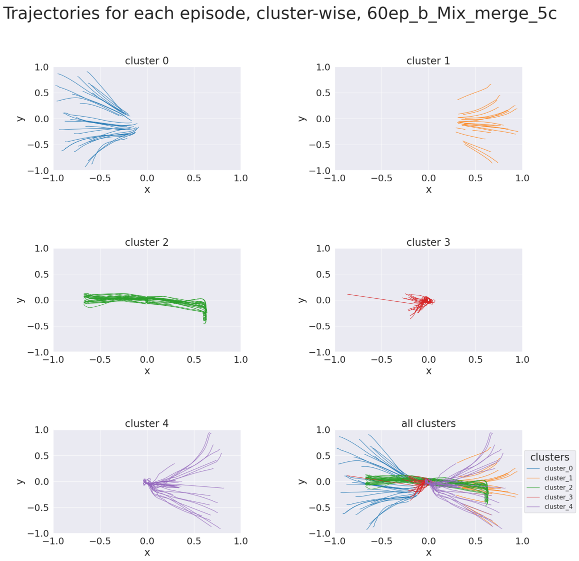


FIGURE 3.12: Spacial distribution of clusters

With this graphic, it is clearly shown that the clustering divide the episodes between the left and right starting sides for the approach and fight phases. This graphic also show that the return phase is similar for both starting sides. But this graphic also highlight a problem that was visible in the Markov chain, some part of episodes are associated to the cluster 4 while they may look like they initially belong to cluster 1. This graphic confirms that the behavior is not symmetric between both side, but it does not explain why. With the help of other graphics and analysis, it was shown that the behavior also depends on the position of the enemy that is not shown here.

3.4 Probe Sensing

Probe Sensing (PS) is the third and last method that was applied. This method is an idea from Delfox inspired by *cite quake III*. This section section will first present the necessary elements to understand the PS methods : The reasoning, the process, the objectives of the use of such methods, the different possible levels of study, the shifted probes and the pipeline. After this presentation, this section will present two different axis of analysis. Those axis are linked to the levels that will be presented beforehand.

3.4.1 Concept and Objectives

The Reasoning

The reasoning for this method is more complex than for the two previous methods. Understanding the intuition requires to be familiar with neural networks. Those are the steps of the reasoning :

- The agent's action requires a complex calculation, otherwise there would be no need for neural networks.
- This computation transforms the inputs into a new representation of the environment that is specific to the agent.
- It is from this representation that the agent takes its decisions, this representation is located the last layer of neurons.
- The part of the neural network between the input and the last layer of neurons must be seen as a function that transforms the observation space into a representation of this space.
- Probe sensing is the study of this representation. It aims to discover what information the agent has access to and/or the information it learned to compute.
- Those information are the information that can be calculated from this representation.

However, it is not possible to be sure that one information has been used in a decision or that it is usually used. Probe Sensing cannot provide certainty.

The Process

The Probe Sensing method uses a trained model to build a probe sensing model and train this model to predict the aforementioned information. The following list of steps are represented by the figure 3.13 :

- Consider the first part of the network as a black box.

- Replace the output corresponding to the four actions by a unique output. The output of prediction of an information. This information is called a probe. (For examples of probes, refer to section 3.4.1)
- Recover the weights of the first part of the model and freeze them. Only the weights corresponding to the calculation of the information from the representation are trainable.
- Train this new neural network to predict information, this network will be called the probe sensing model.
- Study the prediction capacities and the accuracy of the probe sensing model.

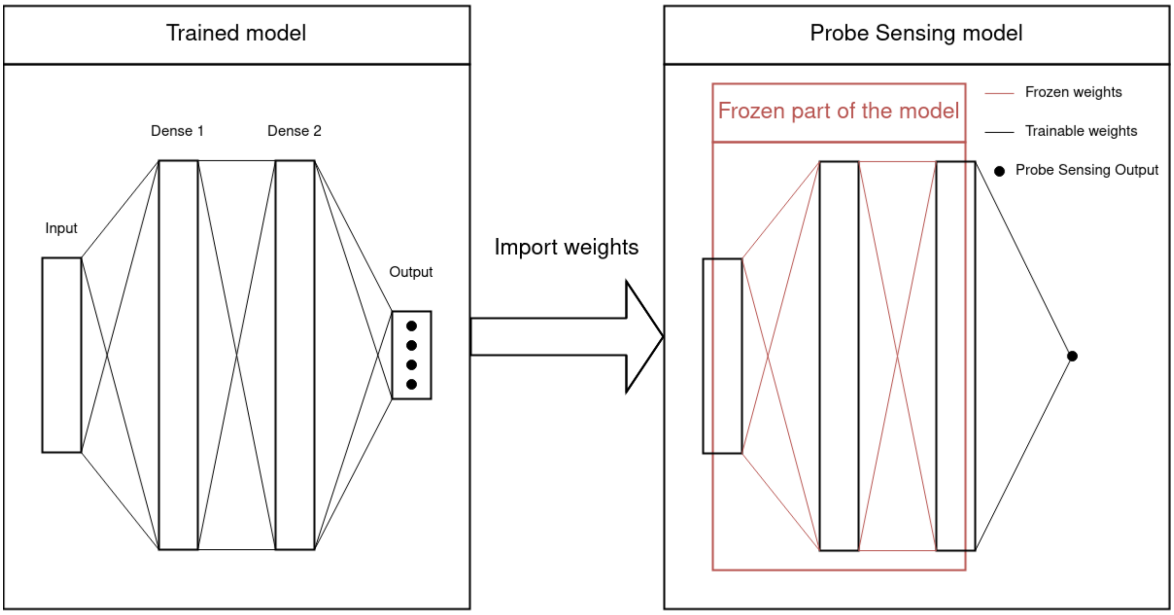


FIGURE 3.13: Schema of the Probe Sensing Process

One probe sensing model is made for each different probe. The data used to train those model are from simulated episodes and the probes are calculated manually. All model are trained with the same hyperparameters so that the results can be compared.

The Different Types of Probes

Probes can be seen as questions, a question on one information and the value that need to be predicted by the model is the answer to this question for the given data. There exist many different types of probes, to simplify the model, they were classed into two big categories:

- **Binary probes**: Question were there are only two solutions. For example: Is the flag captured? Is the enemy dead?
- **Numerical probes**: Question that ask to evaluate a value: How far is the enemy? How long before the flag is captured?

The Objective

The objective of this method is to comprehend the representation the agent have of the environment. In fact, this method provides a way or have an intuition on the information the agent has access to and/or the information it learned to compute. This may seems like a little step but it is not really possible to think like an AI. This method, like the others aims to be part of an ensemble of methods that provide complementary information.

The levels

This method have the same particularity has the Feature Relevance methods, it can be applied at different levels. This method allow local and global explanations.

- **The level of a decision** is the necessary first step for the other levels. It is the comparison of the predictions and the reality.
- **The level of an episode** is the concatenation of the method applied to all time-steps of an episode, this level is still considered a local explanation. This level also allow to directly compare predictions with reality. In one graphic it is possible to see all predictions and real probes value. Therefore, with the presentation of this level, a presentation of the level of decision is not necessary. This level is presented in section 3.4.2.
- **The level of an ensemble of episodes**, this level computes global explanations. It provides the accuracy of the probe sensing model. This level gives the margin of error with which affirmations are made in the previous level. This level is presented in section 3.4.3.

The Shifted Probes

The aforementioned probes are information from a step t that need to be predicted from the observations of step t. However, it is possible to predict information on step $t+k$ with the observations of step t. This creates a new set of probes called shifted probes, all probes can be shifted, the shift can be positive or negative.

The shifted probes allow to analysis the capacity of the agent to predict future information or remember past information. Those prediction may not be easy if the model does not have a memory. The shifted probes can also be analyzed with the three levels presented above.

The thesis will not present the analysis of Heaxplain with the shifted probes because the model in Heaxplain does not have a memory and results were less interesting than the normal probes. For most probes the mean error grows linearly with the shift.

The Pipeline

For all levels, the first part of the pipeline is common. The level of a decision is not represented because there are no visualization generated from this level. In fact, the level of a decision can be seen in the level of an episode. As depicted in the figure 3.14, the common part of the pipeline is divided into three big parts:

- **Generate Data**: The generation of the data necessary to train the probe sensing model is done in three parts:
  - **Simulate episodes**, as for Feature Relevance and Observation Clustering, the first step is to simulate the episodes. This creates the input of the probe sensing models.
  - **Compute probes**, the probes are manually computed from the episodes data. They are the expected output of the probe sensing models.
  - **Compute shifted probes**, the shifted probes are built from the probes. The outputs are just associated with different inputs.
- **Build the probe sensing models**: The creation of the probe sensing models is also done in three part. It only needs the weights of the trained model of the agent.
  - **Build structure**, the first step for every neural network is to set the structure.
  - **Import weights from trained model**, when the structure is built according to the process aforementioned, it is possible to import the weights.
  - **Freeze the weights from trained model**, those weights are what is studied, therefore they should not be updated, thus they are frozen.
- **Train the probe sensing models**, when the data have been generated and the models have been initiated then it is possible to train the models. There are many different models to train, there is one model for:
  - **Each probe**: All probes have a specific model, the weights are trained for each probes.
  - **Each layer**: It will be presented in section 3.4.2 but the concept of probe sensing can be applied to other layers, not only the last layer.
  - **Each shift**: The study of shifted probes, multiply by a huge amount the number of models that need to be trained. Each pair probe-shift have a specific model. Therefore, to reduce this number, only a part of the probes are studied here, the ones that are considered interesting.

After those common parts, when the models are trained, different levels of study can be applied:

- **The level of an episode**, at this level, predictions are made with the observations from one episode and directly compared to the real values of the probe. This level is presented in section 3.4.2).
- **The level of an ensemble of episodes**, this level evaluates the trained probe sensing models with different metrics and analyze results. This level is presented in section 3.4.3.

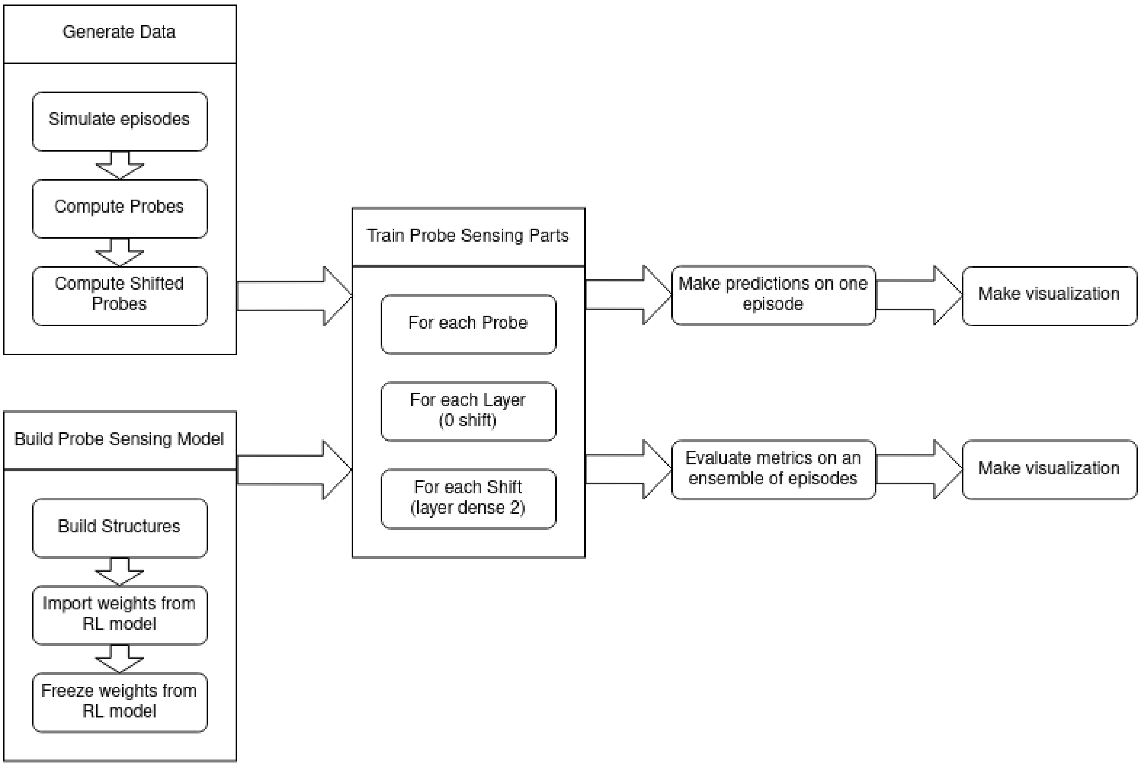

Generate Data
Simulate episodes
Compute Probes
Compute Shifted Probes
Build Probe Sensing Model
Build Structures
Import weights from RL model
Freeze weights from RL model
Train Probe Sensing Parts
For each Probe
For each Layer (0 shift)
For each Shift (layer dense 2)
Make predictions on one episode
Make visualization
Evaluate metrics on an ensemble of episodes
Make visualization

FIGURE 3.14: Probe Sensing Pipeline

3.4.2 Study at the Level of an Episode

This section will present how is built this level, introduce the graphics that are exported from this level, show two examples and provide their respective analysis.

The level of an episode for the probe sensing method is the comparison between true probe values and the probe sensing model predictions. The predictions are made for all steps of an episode, then they are concatenated temporally ordered. The same is done for the true values of the probes, hence for each probe and episode, there are two vectors of values with the length of an episode.

In the visualizations created at this level and thus for the following figures 3.15 and 3.16, the two vectors are represented by two lines. The blue line for the predictions and the green line for the true values. The x-axis represent the time in number of step from the beginning of the episode and the y-axis represent the probe values. For binary probes, a one signifies true while zero means false. On those graphics, if the two lines are close, then the predictions accurate.

The probe presented in figure 3.15 is a binary probe, it corresponds to the question: "Is the agent shooting?". The agent have 12 missiles and is therefore shooting for 12 consecutive steps, from step 38 to 50. The model detect that the agent is shooting one step after the agent is actually shooting and detect the stop one step earlier. This leads to an accuracy of approximately 98%. The fact that the agent is shooting is complex to predict because the actions of the agent are involved and the probe sensing model do not have access to them. Nevertheless, there are errors, it is not

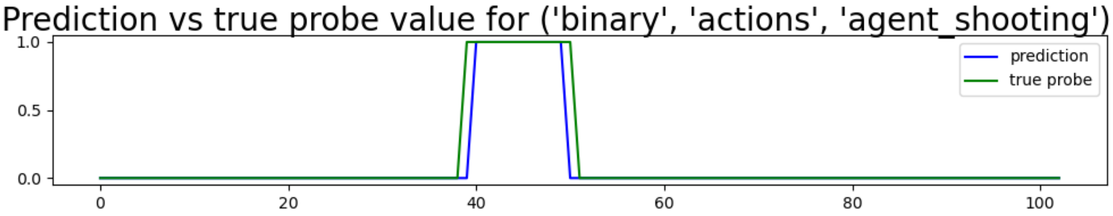


FIGURE 3.15: Comparison Between Probe Sensing and Reality on Probe: "Is the Agent Shooting?"

possible to say that agent knows when it is shooting. This probe is a great example because with binary probes, the errors are essentially done at the transition between true and false. This is the moment where the model is the most uncertain.

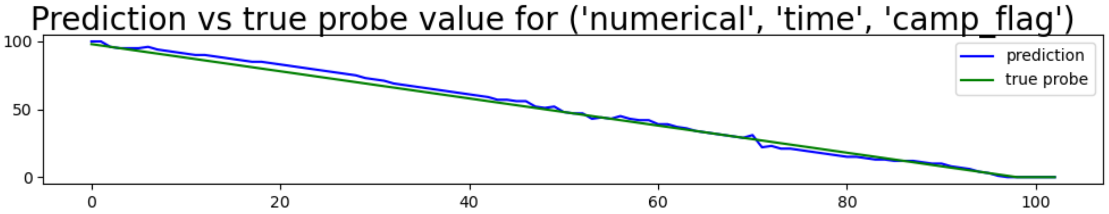


FIGURE 3.16: Comparison Between Probe Sensing and Reality on Probe: "How Long Before the Agent Reach his Camp with the Flag?"

The probe presented in figure 3.16 is a numerical probe, a time estimation probe, it correspond to the question: "How long before the agent gets to his camp with the flag?". This probe gives the number of steps remaining before the agent wins, with 5 extra steps to be sure it stays in his camp. On the figure, the two lines are close and the maximal distance between the two is 3 steps (the y-axis also represents steps here). The agent can predict with less than 3 steps of error, the time it will need to win since the beginning. It is possible to predict the length of the simulation only with the initial positions. This is weird because the agent often miss the flag once or twice during an episode and this delay the end of the episode by than more than 3 steps. However, the probes with accurate results like this one are useful for Delfox. This means that during a simulation it is possible to show the time before the end of the episode estimated by the agent. When this time is accurate, this shows to Delfox's clients that the agent knows what it is doing and understands its environment.

3.4.3 Study at the Level of an Ensemble of Episodes

The level of an ensemble of episodes will be presented first theoretically then through the example of one type of graphic. There are other possible graphics but one type is enough to illustrate the uses of a method. For this graphic, a new metric has been defined and probe sensing has been applied in a more complex way as explained before. Hence those two steps will be presented before the examples and their analysis.

The level of an ensemble of episodes is the study of metrics on the probe sensing models. It allows to take a step back from the results from the previous level and provide mathematical justifications to the previous affirmations. Those justifications are based on statistics on more precisely on the metrics computed between the predictions and the probes true values. It is then possible to compare those metrics between probes, between shift, between layers or even between times of an episode.

The New Metrics

There are many metrics but the most common and easy to understand is the accuracy. However, the accuracy have disadvantages, for example: it does not differentiate between the two type of errors. Hence it is easy to get a high accuracy with an unbalanced dataset. Also, binary probes are often unbalanced, sometimes there are more than 95% of zeros in the dataset.

To solve this problem, the idea of a baseline was proposed, this baseline was made with a dumb classifier. A dumb classifier always output the same value, which is the most common value in the training set for binary probes and the mean for numerical probes. Then a new metric was developed, the accuracy gain from the baseline:

$$acc_{gain} = \frac{acc - acc_{baseline}}{1 - acc_{baseline}}$$

Note that if $acc_{baseline} = 1$ then all real value of a probe are the same, hence the probe is not interesting and should be removed.

Nonetheless, the accuracy does not exist for non-binary values, thus it was necessary to create a comparable metric from the mean absolute error. (mean square error could have been used but it is harder to interpret). The transformed mean absolute error will be noted mae^t and is computed with the mae and the maximum error mae_{max} in the following way:

$$mae^t = \frac{mae_{max} - mae}{mae_{max}}$$

Note that the mae is always positive and lower than mae_{max}, thus mae^t is between 0 and 1, the closer it is to one, the smaller is the mae. Then the same calculation as for the accuracy can be executed.

This new metrics have a huge advantage, it is that binary and numerical probes can be treated the same way. Therefore the graphic for both types are the same and the interpretations are also the same. Hence Delfox only have to explain one graphic to his clients, which are usually familiar with metrics that go from 0 to 1 where 1 is the best result. Additionally, the results are easy to treat for the R&D team.

Application of Probe Sensing to Several Layers

After this metric, a new idea emerged, the idea to apply probe sensing to the first layer of the neural network and to the input. The utility of such method would be to compare the results between layers, *i.e* compare the representations of the environment. If it is easier to compute a probe with two layers of the neural networks than with just one layer, this means that the added layer helps the computation of the probe.

The interpretation of such result must not be made hastily. The amelioration of the result on a probe could be due to the fact that the model learned to compute another information and that the means to compute this information helped to compute the probe. However, when the results on the last layer are worse than those on the input, then the probability that the information is used in the decision process is close to zero. Nevertheless, it is still possible that the agent does not need the precision available in the observations and thus losses information. Those two points highlight the fact that probe sensing cannot provide certain results and must be interpreted sparingly.

The process to apply probe sensing to the other layers of the neural network is close to the original one. The only difference between the two process is the the part of the original model that is exported to the probe sensing model. As aforementioned, the frozen part of the model can be seen as a black-box where the output is connected. The difference between each layer is the black-box, this is depicted in figure 3.17. The probe sensing applied to the input corresponds to when there is no black box and that the observations are directly connected to the output. The baseline that is noted benchmark in the figure can be seen as the output that is not connected to anything, it can only train its bias. There is one probe sensing model for each pair probe-layer.

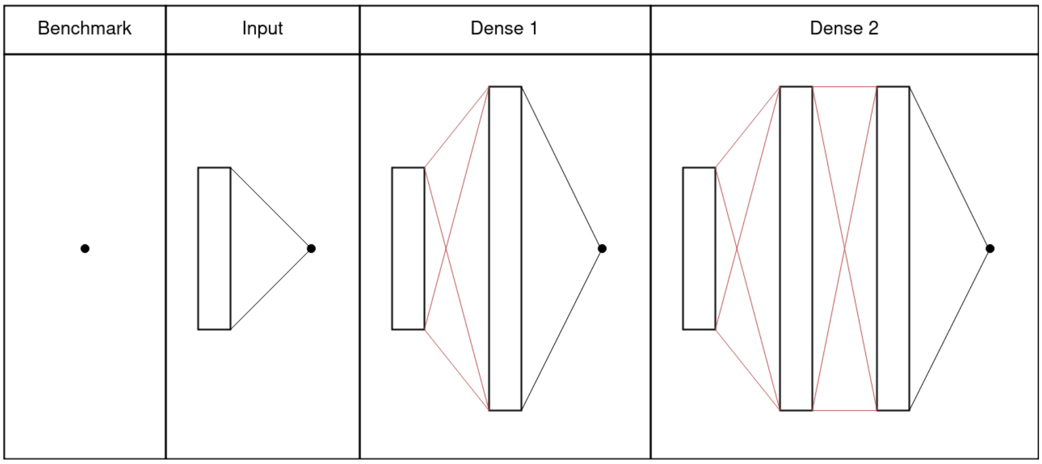


FIGURE 3.17: Schema of Probe Sensing Applied to Several Layers

The Examples and the Analysis

The following graphic uses both the new metric and the three layers. To build this graphic, the first step is to compute the metrics, then, from those metrics create the new metric using the baseline. Finally, compare the new metrics obtain on each layer.

The following graphics are barplots that show this comparison for several probes. The objective in this section is not to interpret the result of each probe and make conclusion on the behavior of the agent. The objective is to illustrate the probe sensing method and provide examples of interpretations. But naturally, the link with the behavior of the agent was done in the documentation provide to Delfox.

For figures 3.18 and 3.19, there are three bars for each probes, each one represents respectively the probe sensing applied to the input, to the first layer and to the second layer (blue, orange and green). The height of each bar corresponds to the value of the new metric obtained on the corresponding probe sensing model.

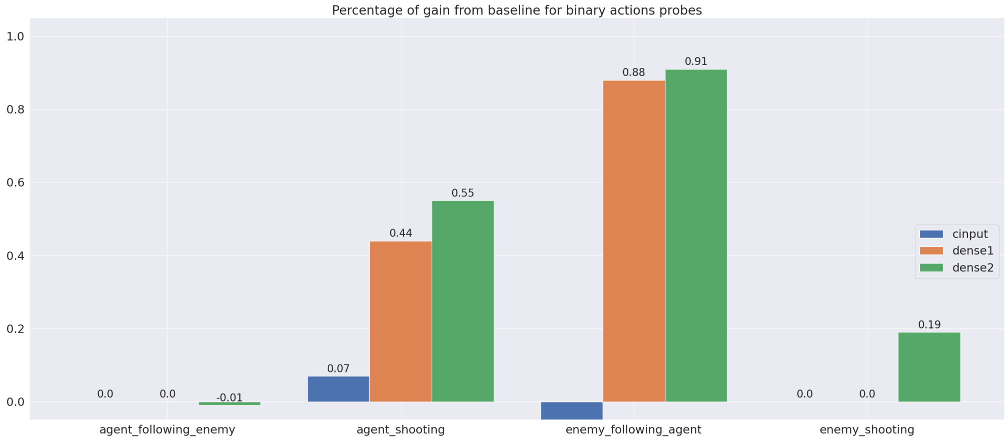


FIGURE 3.18: Comparison of Probe Sensing Result on Different Layers for Binary Probes on Agent's Actions

The figure 3.18 group together binary probes linked to actions of the agent or the enemy, such as shooting or following the opponent. For the first probe, there is a negative value because the trained probe sensing model have a worse accuracy than the baseline. It is thus impossible for such information to be used in the decision process. The second probe shows a clear gain between layers, but practice has shown that such schema cannot prove that the probe is used. The third probe shows however an important gain with layers one and two, hence the agent has learned to compute this probe (or a similar information) and there is a high probability that it is used in the decision process. The fourth one may seem like a bad result in the first place. But it shows that the probe is complicated to compute and that two layers were necessary to gain accuracy. Furthermore, this probe is indeed complicated compute, it is unbalanced (98% of false) and it needs information on the actions of the enemy that are not provided to the agent. Therefore, the agent seems to use something closely linked to this probe.

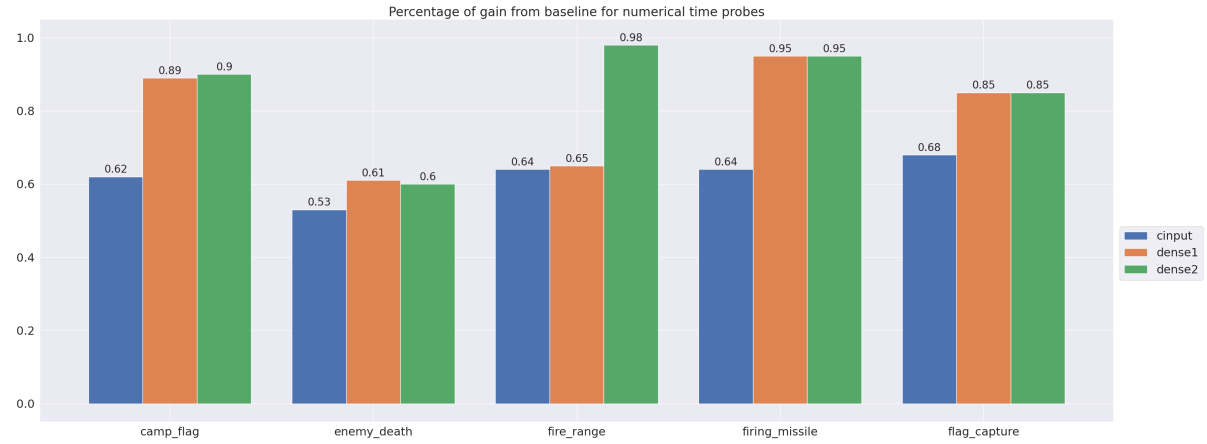


FIGURE 3.19: Comparison of Probe Sensing Result on Different Layers for Numerical Probes on Time Estimation

The figure 3.19 group together numerical probes of estimation of the time remaining before an event. This graphic serves as an example to show that binary and numerical probes can here be analyzed with the same process. The first, fourth and fifth probes follow the same schema, information are provided in the input and the first layer make the computation necessary for those probes. There is a high probability that a highly correlated information (or the probe itself) is used in the decision process. The second probe show only a small improvement from the input to the second layer, this means that the probe is not complicated to compute and that further precision is not necessary. Finally, the third probe, it shows a gain close to 100% for the second layer while the gain for the input and the first layer is about 2/3. Therefore this probe is complicated to compute and it is possible to say that it is used in the decision process.

4. Conclusion

To conclude this thesis and make clear what was done during this internship, the conclusion part will begin with a review of the obtained results, enumerate a list of the contributions, present the perspectives, provide my feed back on the internship and finally reflect on the environmental and societal impacts of this thesis.

4.1 Review of Obtained Results

This section compare the results of this internship to the original objectives. The original objectives proposed in the subject of this thesis are:

- Survey of the XAI scientific literature, particularly focused on post-hoc techniques;
- Choice, implementation and test of the most promising approaches;
- Adaptation to the specific context of deep reinforcement learning;
- Application to real complex problems brought by our industrial clients.

The idea of a framework was not precised in the subject but brought soon after the first application of the methods. To summarize, this internship was done to explore the possibilities of explainability and provide means to use it if the results were conclusive.

As shown in this report, an exhaustive bibliography of possible methods was effectuated (see section 2). From this bibliography, the three most promising methods were applied to a project from an industrial client. Some of those methods were adapted from XAI to XRL.

To apply those methods, a framework was built, the idea was to provide a tool to apply the methods automatically. The framework takes as input a trained model of the agent and the means to do simulations. The output of the framework is a large set of visualizations providing explanations on the agent decisions and behavior.

The explanations provided through the various methods were proven satisfactory through the feedback of Delfox's team after the presentation of the results. Each method had its advantages and their explanations were complementary.

4.2 Contributions

The contributions of this internship can be divided in two groups, the contributions to Delfox and the contributions to research. Even if the contributions to research may only affect the company and were done in order to complete the objectives of the internship.

4.2.1 Contributions to Delfox

The contributions to Delfox represent the work that have, will or may impact Delfox:

- The presentation of the state of the art in XAI and XRL and the presentation of the possible explainability methods for Delfox and the objectives of this thesis.
- The documentation of the bibliography that was effectuated.
- The framework that automatically apply the three presented methods and export all the visualizations.
- The presentation of those methods and the presentation of their results.
- The documentation of the framework, an explanation of each method, an explanation of each graphic and an example of analysis.
- For Observation Clustering, the results were better than the baseline.

4.2.2 Contributions to Research

The contribution to research can be seen as the list of methods and tools that were developed for this thesis but did not exist beforehand:

- Creation of a new Taxonomy for XAI through the merge of other taxonomies.
- Development of the idea of levels of study in XRL.
- Adaptation of Features Relevance to XRL with time related analysis and several level of analysis.
- A method to apply Observation Clustering.
- A method to merge different clusterings.
- The development of Probe Sensing and a method to apply it with several levels of study.

4.3 Perspectives

The initial objectives of the internship have been fulfilled, nonetheless, there are still many possibilities to explore and improvements to provide to the framework:

- The three methods have been studied extensively, but there are still possibilities to explore.
- Three methods were applied, this do not mean that there are no other interesting methods, for example distillation. (see section 2.2).
- The framework have been tested on Heaxplain, but it needs to be validated on other projects. It has been developed so that it can be adapted easily, however, it is not possible to know how well it works without trying it.
- Delfox is now developing his first product which aims at making reinforcement learning accessible to industrial experts without requiring any IA skill. The integration of interpretability to such a product is a strategic point and a future objective.
- The explanations and graphics generated by the framework were judged satisfactory by Delfox. Nonetheless, the real entity that needs to be convince are the clients of Delfox. Therefore, the opinion of the clients is the most important factor to decide if the explanations were relevant. Therefore, those explanations should be confronted to clients opinion.

4.4 Feed Back

This thesis contributed to Delfox, but it also contributed to my development, this section will present my feed back through the encountered difficulties and what was learned.

4.4.1 The difficulties

During an internship or any type of work, facing difficulties is bond to happen. For this thesis, the main difficulties were:

- **Begin a bibliography**: I had never done it before. Therefore I was lost between: the number of papers, the order in which I should read them and how to organize my thoughts.
- **Conflicts between libraries**: At some point in the internship, I tried to apply Feature Relevance (FR) to another project of Delfox (which had a really particular structure.) After several days of trials, I discovered that the library for FR and a library used for the RL algorithm were incompatible. There were no solutions to use both of them at the same time.
- **Redaction**: To write the documentation and the reports, it requires redaction's skills in which I am lacking. Therefore, to produce qualitative work, I needed a lot more time than I expected.
- **Methods Problems**: The application of each method came with several difficulties, however there was nothing too important.

4.4.2 What was learned ?

Difficulties, errors and critics are what allows us to grow, hence those difficulties made me learn or improve on the following points:

- To begin a a bibliography, the first thing to do is to look for surveys around your problematic. They usually go through more papers than you have the time to, they provide summarize the key elements from all those papers and provide a list of papers that may help for your problematic.
- Sometimes, libraries are incompatible, it is necessary to change at least one of the two or develop it yourself. Therefore, before selecting a library, one should always check the compatibility with all the other libraries already used. However, it was not possible here because this incompatibility was not obvious, I did not find a solution to prevent this problem.
- Through the critics and remarks of my tutor at Delfox, Dr. Xabier Jaureguyberry, I improved my writing skills a lot. It is still one of my main flow but I know it and I am working on it.
- The application of the several interpretability methods naturally enhanced my comprehension of explainability, machine learning and deep learning.
- I also improved in communication, I had three presentations to do and the key element to explainability is the ability to convey information.

Bibliography

Baehrens, David et al. (2010). "How to Explain Individual Classification Decisions". In: *J. Mach. Learn. Res.*

Zeiler, Matthew D. and Rob Fergus (2013). "Visualizing and Understanding Convolutional Networks". In: preprint arXiv: 1311.2901.

Bach, Sebastian et al. (2015). "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation". In: *PLoS One*.

Lloyd, James R and Zoubin Ghahramani (2015). "Statistical Model Criticism using Kernel Two Sample Tests". In:

Rusu, Andrei A. et al. (2015). "Policy distillation". In: preprint arXiv: 1511.06295.

Ribeiro, Marco Túlio, Sameer Singh, and Carlos Guestrin (2016). ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: preprint arXiv: 1602.04938.

Hayes, Bradley and Julie A. Shah (2017). "Improving Robot Controller Transparency Through Autonomous Policy Explanation". In:

Lundberg, Scott and Su-In Lee (2017). "A unified approach to interpreting model predictions". In: preprint arXiv: 1705.07874.

Schulman, John et al. (2017). "Proximal Policy Optimization Algorithms". In: preprint arXiv: 1707.06347.

Shu, Tianmin, Caiming Xiong, and Richard Socher (2017). "Hierarchical and Interpretable Skill Acquisition in Multi-task Reinforcement Learning". In: preprint arXiv: 1712.07294.

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan (2017). "Axiomatic Attribution for Deep Networks". In: preprint arXiv: 1703.01365.

Wachter, Sandra, Brent D. Mittelstadt, and Chris Russell (2017). "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR". In: preprint arXiv: 1711.00399.

Adadi, Amina and Mohammed Berrada (2018). "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)". In: *IEEE Access*.

Icarte, Rodrigo Toro et al. (2018). "Using Reward Machines for High-Level Task Specification and Decomposition in Reinforcement Learning". In:

Carvalho, Diogo V., Eduardo M. Pereira, and Jaime S. Cardoso (2019). "Machine Learning Interpretability: A Survey on Methods and Metrics". In: *Electronics*.

Jaderberg, Max et al. (2019). "Human-level performance in 3D multiplayer games with population-based reinforcement learning". In: *Science*. ISSN: 0036-8075.

Belle, Vaishak and Ioannis Papantonis (2020). "Principles and Practice of Explainable Machine Learning". In: eprint: 2009.

Das, Arun and Paul Rad (2020). "Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey". In: eprint: 2006.

Lowe, Ryan et al. (2020). "Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments". In: preprint arXiv: 1706.02275.

Puiutta, Erika and Eric MSP Veith (2020). "Explainable Reinforcement Learning: A Survey". In: preprint arXiv: 2005.06247.

Sequeira, Pedro and Melinda Gervasio (2020). "Interestingness elements for explainable reinforcement learning: Understanding agents' capabilities and limitations". In: *Artificial Intelligence*.

Taxonomy

For the description of the levels, see section 2.1

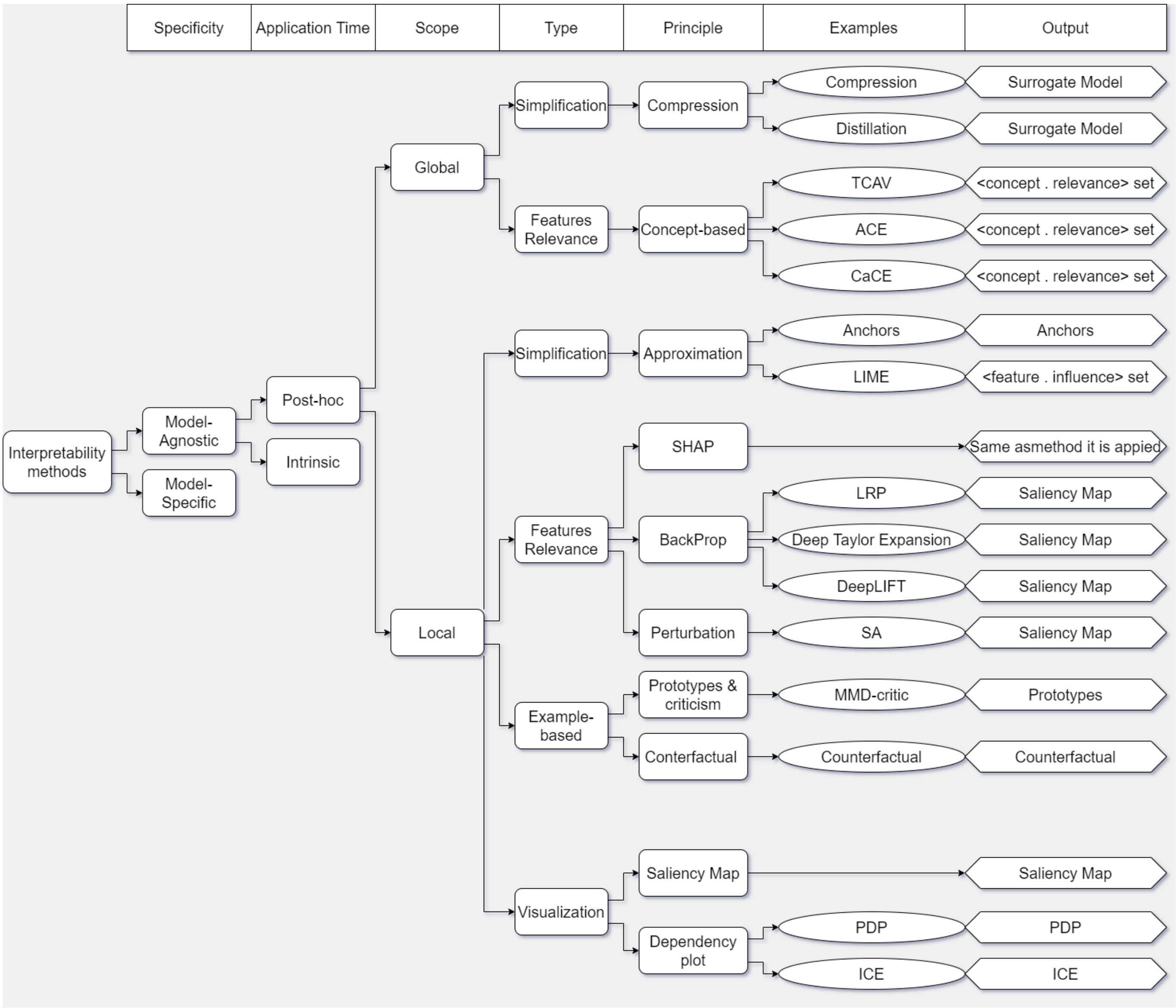
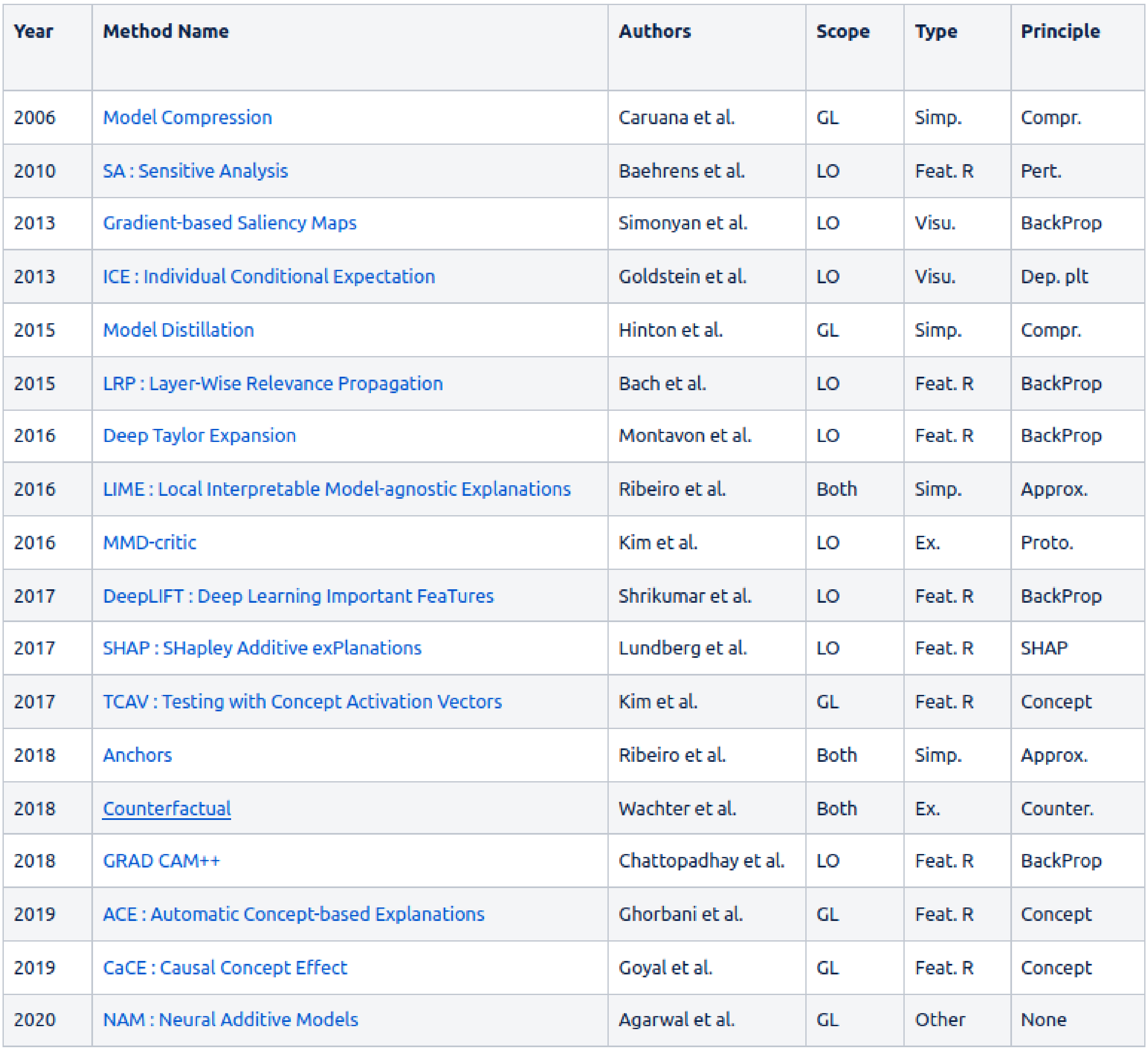


FIGURE A.1: Taxonomy Tree Diagram

Methods Grid

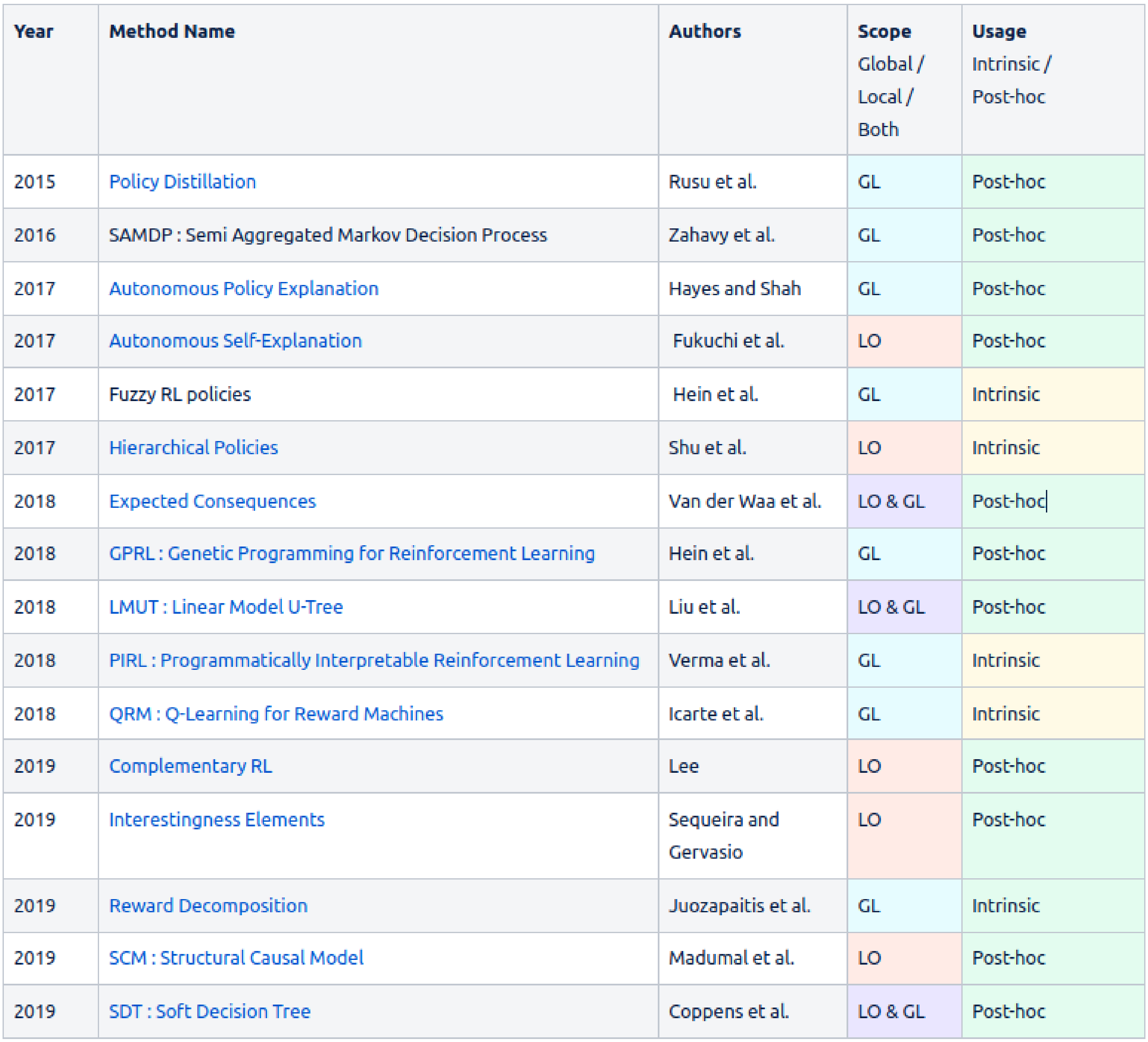
For details, see 2.1. List of abbreviations:

- GL: Global,
- LO: Local,
- Simp.: Simplification,
- Feat. R.: Feature Relevance,
- Visu.: Visualization,
- Ex.: Example-based,
- Compr.: Compression,
- Pert.: Perturbation,
- BackProp: Back-Propagation,
- Dep. plt: Dependency plot,
- Approx.: Approximation,
- Proto.: Prototypes,
- Counter.: Counterfactuals



Year	Method Name	Authors	Scope	Type	Principle
2006	Model Compression	Caruana et al.	GL	Simp.	Compr.
2010	SA : Sensitive Analysis	Baehrens et al.	LO	Feat. R	Pert.
2013	Gradient-based Saliency Maps	Simonyan et al.	LO	Visu.	BackProp
2013	ICE : Individual Conditional Expectation	Goldstein et al.	LO	Visu.	Dep. plt
2015	Model Distillation	Hinton et al.	GL	Simp.	Compr.
2015	LRP : Layer-Wise Relevance Propagation	Bach et al.	LO	Feat. R	BackProp
2016	Deep Taylor Expansion	Montavon et al.	LO	Feat. R	BackProp
2016	LIME : Local Interpretable Model-agnostic Explanations	Ribeiro et al.	Both	Simp.	Approx.
2016	MMD-critic	Kim et al.	LO	Ex.	Proto.
2017	DeepLIFT : Deep Learning Important FeaTures	Shrikumar et al.	LO	Feat. R	BackProp
2017	SHAP : SHapley Additive exPlanations	Lundberg et al.	LO	Feat. R	SHAP
2017	TCAV : Testing with Concept Activation Vectors	Kim et al.	GL	Feat. R	Concept
2018	Anchors	Ribeiro et al.	Both	Simp.	Approx.
2018	Counterfactual	Wachter et al.	Both	Ex.	Counter.
2018	GRAD CAM++	Chattopadhay et al.	LO	Feat. R	BackProp
2019	ACE : Automatic Concept-based Explanations	Ghorbani et al.	GL	Feat. R	Concept
2019	CaCE : Causal Concept Effect	Goyal et al.	GL	Feat. R	Concept
2020	NAM : Neural Additive Models	Agarwal et al.	GL	Other	None

FIGURE B.1: grid of XAI methods



Year	Method Name	Authors	Scope Global / Local / Both	Usage Intrinsic / Post-hoc
2015	Policy Distillation	Rusu et al.	GL	Post-hoc
2016	SAMDP : Semi Aggregated Markov Decision Process	Zahavy et al.	GL	Post-hoc
2017	Autonomous Policy Explanation	Hayes and Shah	GL	Post-hoc
2017	Autonomous Self-Explanation	Fukuchi et al.	LO	Post-hoc
2017	Fuzzy RL policies	Hein et al.	GL	Intrinsic
2017	Hierarchical Policies	Shu et al.	LO	Intrinsic
2018	Expected Consequences	Van der Waa et al.	LO & GL	Post-hoc
2018	GPRL : Genetic Programming for Reinforcement Learning	Hein et al.	GL	Post-hoc
2018	LMUT : Linear Model U-Tree	Liu et al.	LO & GL	Post-hoc
2018	PIRL : Programmatically Interpretable Reinforcement Learning	Verma et al.	GL	Intrinsic
2018	QRM : Q-Learning for Reward Machines	Icarte et al.	GL	Intrinsic
2019	Complementary RL	Lee	LO	Post-hoc
2019	Interestingness Elements	Sequeira and Gervasio	LO	Post-hoc
2019	Reward Decomposition	Juozapaitis et al.	GL	Intrinsic
2019	SCM : Structural Causal Model	Madumal et al.	LO	Post-hoc
2019	SDT : Soft Decision Tree	Coppens et al.	LO & GL	Post-hoc

FIGURE B.2: grid of XRL methods

Explanation properties

C.1 Properties of Individual Explanations

The following list was extracted from (Carvalho, Pereira, and Cardoso 2019). Definitions have not been changed, only shortened for some.

- **Accuracy**: It is related to the predictive accuracy of the explanation regarding unseen data.
- **Fidelity**: It is associated with how well the explanation approximates the prediction of the black box model.
- **Consistency**: Regarding two different models that have been trained on the same task and that output similar predictions, this property is related to how different the explanations are.
- **Stability**: It represents how similar the explanations are for similar inputs for a fixed model.
- **Comprehensibility**: This property is one of the most important but also one of the most difficult to define and measure. It is related to how well humans understand the explanations.
- **Certainty**: It reflects the model's confidence on the correctness of the prediction.
- **Importance**: It is associated with how well the explanation reflects the importance of features.
- **Novelty**: It describes if the explanation reflects whether an instance comes from a region in the feature space that is far away from the distribution of the training data.
- **Representativeness**: It describes how many instances are covered by the explanation.

C.2 Properties of Explanation Methods

The following list was extracted from (Carvalho, Pereira, and Cardoso 2019).

- **Expressive power**: It is the language or structure of the explanations the method is able to generate. These could be, e.g., rules, decision trees, or natural language.
- **Translucency**: It represents how much the explanation method relies on looking into the inner mechanism of the ML model, such as the model's parameters. For example, model-specific explanation methods are highly translucent. Accordingly, model-agnostic methods have zero translucency.
- **Portability**: It describes the range of ML models to which the explanation method can be applied. It is inversely proportional to translucency, meaning that highly translucent methods have low portability and vice-versa. Hence, model-agnostic methods are highly portable.
- **Algorithmic complexity**: It is related to the computational complexity of the explanation method. This property is very important to consider regarding feasibility, especially when computation time is a bottleneck in generating explanations.

Example of the Minimum Size Results

The figures D.2 and D.1 are TSNE's representations of clustering. The figure D.2 show the results when there are no minimum size for the clusters. (There are four clusters but only three can be seen). However on figure D.1, they clearly appear.

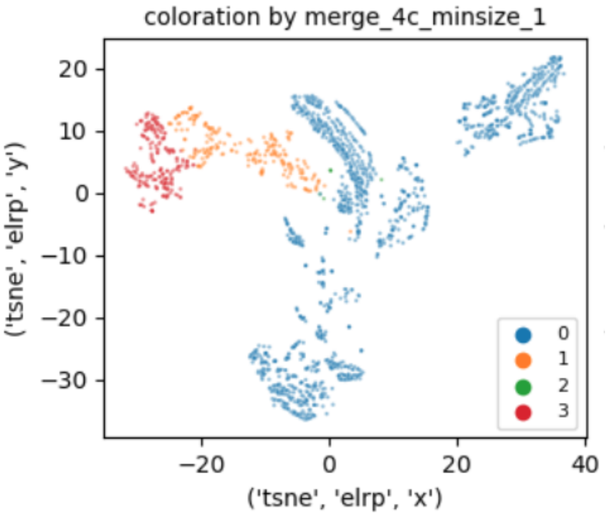


FIGURE D.1: Clustering Results Without the Minimum Size

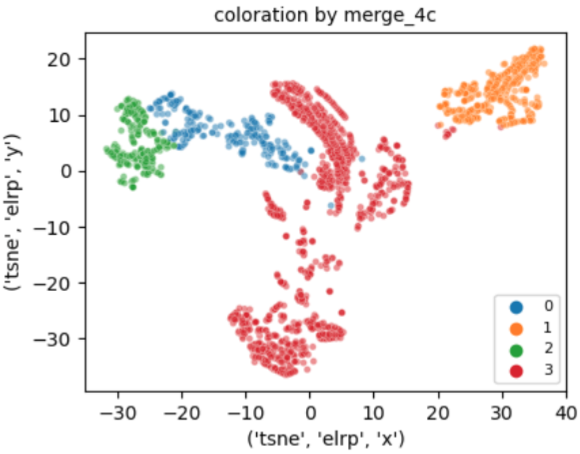


FIGURE D.2: Clustering Results Without the Minimum Size