

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Bruno DATO

Le 9 juillet 2021

Apprentissage permanent par feedback endogène, application à un système robotique

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Informatique et Télécommunications**

Unité de recherche :
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Marie-Pierre GLEIZES et Frédéric MIGEON

Jury

M. Olivier SIMONIN, Rapporteur
M. Laurent VERCOUTER, Rapporteur
M. Stéphane DONCIEUX, Examinateur
Mme Marie-Pierre GLEIZES, Directrice de thèse
M. Frédéric MIGEON, Co-directeur de thèse

Bruno Dato

APPRENTISSAGE PERMANENT PAR FEEDBACK ENDOGÈNE APPLICATION À UN SYSTÈME ROBOTIQUE

Directrice : Marie-Pierre Gleizes, Professeur, UT3

Co-Encadrant : Frédéric Migeon, Maître de Conférences, UT3

Résumé

Les applications robotiques sont liées à l'environnement sociotechnique dynamique dans lequel elles sont intégrées. Dans ce contexte, l'auto-adaptation est une préoccupation centrale et la conception d'applications intelligentes dans de tels environnements nécessite de les considérer comme des systèmes complexes. Le domaine de la robotique est très vaste. L'accent est mis sur les systèmes qui s'adaptent aux contraintes de leur environnement et non sur la mécanique ou le traitement du signal.

À la lumière de ce contexte, l'objectif de cette thèse est la conception d'un mécanisme d'apprentissage capable d'apprendre de manière continue en utilisant des feedbacks endogènes (i.e. des interactions internes) dans des environnements sociotechniques dynamiques. Ce mécanisme d'apprentissage doit aussi vérifier plusieurs propriétés qui sont essentielles dans ce contexte comme : l'agnosticité, l'apprentissage tout au long de la vie, l'apprentissage en ligne, l'auto-observation, la généralisation des connaissances, le passage à l'échelle, la tolérance au volume de données et l'explicabilité.

Les principales contributions consistent en la construction de l'apprentissage endogène par contextes et la conception du mécanisme d'apprentissage ELLSA pour Endogenous Lifelong Learner by Self-Adaptation. Le mécanisme d'apprentissage proposé est basé sur les systèmes multi-agents adaptatifs combinés à l'apprentissage endogène par contextes. La création de l'apprentissage endogène par contextes est motivé par la caractérisation d'imprécisions d'apprentissage qui sont détectées par des négociations locales entre agents. L'apprentissage endogène par contextes comprends aussi un mécanisme de génération de données artificielles pour améliorer les modèles d'apprentissage tout en réduisant la quantité nécessaire de données d'apprentissage. Dans un contexte d'apprentissage tout au long de la vie, ELLSA permet une mise à jour dynamique des modèles d'apprentissage. Il introduit des stratégies d'apprentissage actif et d'auto-apprentissage pour résoudre les imprécisions d'apprentissage. L'utilisation de ces stratégies dépend de la disponibilité des données d'apprentissage.

Afin d'évaluer ses contributions, ce mécanisme est appliqué à l'apprentissage de fonctions mathématiques et à un problème réel dans le domaine de la robotique : le problème de la cinématique inverse. Le scénario d'application est l'apprentissage du contrôle de bras robotiques multi-articulés. Les expériences menées montrent que l'apprentissage endogène par contextes permet d'améliorer les performances d'apprentissage grâce à des mécanismes internes. Elles mettent aussi en évidence des propriétés du système selon les objectifs de la thèse : feedback endogènes, agnosticité, apprentissage tout au long de la vie, apprentissage en ligne, auto-observation, généralisation, passage à l'échelle, tolérance au volume de données et explicabilité.

Bruno Dato

LIFELONG LEARNING BY ENDOGENOUS FEEDBACK
APPLICATION TO A ROBOTIC SYSTEM

Directrice : Marie-Pierre Gleizes, Professeur, UT3
Co-Encadrant : Frédéric Migeon, Maître de Conférences, UT3

Abstrac

Robotic applications are linked to the dynamic sociotechnical environment in which they are embedded. In this scope, self-adaptation is a central concern and the design of intelligent applications in such environments requires to consider them as complex systems. The field of robotics is very broad. The focus is made on systems that adapt to the constraints of their environment and not on mechanics or signal processing.

In light of this context, the objective of this thesis is the design of a learning mechanism capable of continuous learning using endogenous feedback (i.e. internal interactions) in dynamic sociotechnical environments. This learning mechanism must also verify several properties that are essential in this context such as : agnosticity, lifelong learning, online learning, self-observation, knowledge generalization, scalability, data volume tolerance and explainability.

The main contributions consist of the construction of Endogenous Context Learning and the design of the learning mechanism ELLSA for Endogenous Lifelong Learner by Self-Adaptation. The proposed learning mechanism is based on Adaptive Multi-Agent Systems combined with Context Learning.

The creation of Endogenous Context Learning is motivated by the characterization of learning inaccuracies that are detected by local negotiations between agents. Endogenous Context Learning also includes an artificial data generation mechanism to improve learning models while reducing the amount of the required learning data. In a Lifelong Learning setting, ELLSA enables dynamic updating of learning models. It introduces Active Learning and Self-Learning strategies to resolve learning inaccuracies. The use of these strategies depends on the availability of learning data.

In order to evaluate its contributions, this mechanism is applied to the learning of mathematical functions and to a real problem in the field of robotics : the Inverse Kinematics problem. The application scenario is the learning of the control of multi-jointed robotic arms.

The conducted experiments show that Endogenous Context Learning enables to improve the learning performances thanks to internal mechanisms. They also highlight the properties of the system according to the objectives of the thesis : endogenous feedback, agnosticity, lifelong learning, online learning, self-observation, knowledge generalization, scalability, data volume tolerance and explainability.

Ça y est, il est venu le temps de conclure et de remercier toutes les personnes qui ont contribué de près ou de loin à cette thèse. Car oui, ce travail est aussi l'émergence des interactions scientifiques, sociales et amicales que j'ai eues avec de nombreux agents coopératifs tout au long de ma vie et plus particulièrement de ces dernières années.

Je tiens tout d'abord à remercier les membres du jury. Merci à mes rapporteurs, Olivier Simonin et Laurent Vercouter, pour leur travail de relecture. Merci à mon examinateur Stéphane Doncieux pour avoir accepté d'évaluer mon travail de thèse. Merci à tous de vous être intéressés, d'avoir pris le temps de lire mon manuscrit, et d'avoir participé à cette soutenance.

Merci à mes encadrant·e·s Marie-Pierre et Fred qui m'ont guidé pendant cette longue épreuve (oui longue ! Trop longue ? Oui à la fin c'était un peu long !). Merci pour cette expérience, car pour moi, la recherche c'est avant tout une expérience et je suis ravi de l'avoir effectuée avec vous. Merci pour votre vision de la recherche, pour ces réunions où on partait dans tous les sens et après lesquelles je me posais encore plus de questions. Je suppose que c'est aussi ça le charme de la recherche. Ce qui compte c'est que c'était toujours avec la bonne humeur. Marie-Pierre, je ne suis toujours pas convaincu que Le Poutou de Toulouse soit mieux que Les Biroussans, sans rancune ! Fred, dommage que l'on n'ait pas eu plus d'occasions de faire des soirées de danse ou bien d'y initier l'équipe. Ce n'est peut-être pas encore trop tard ! Ensuite, je tiens à remercier une personne sans qui beaucoup de discussions n'auraient pas eu lieu. Merci au soldat de l'ombre, qui, quoi qu'il arrive, était toujours présent, pour discuter, m'aider, me dire que mes choix de couleurs sur les figures étaient moches, Pierre. Merci pour ta franchise, j'espère que tu te souviendras longtemps de mon ton plat et monocorde et de mon enthousiasme démesuré lorsque que je vous présentais mes résultats.

Mine de rien, j'aurais fait un bon bout de route avec vous, l'équipe SMAC. Merci à vous tous pour votre accueil chaleureux et la bonne ambiance que vous entretenez. Même si je n'étais pas souvent présent pour les cafés du matin, c'était un plaisir d'échanger avec vous lorsque j'étais là. Merci à Bob·Nicolas, Julien et Jérémy de m'avoir introduit à cette grande famille lorsque je suis arrivé comme stagiaire.

Nous partîmes sept, certains tombèrent dans la bataille, d'autres restèrent jusqu'au bout et certains se battent encore. Force à Tim et Doryan ! Maxime, dommage que l'on n'ait pas pu faire ce fameux semi-marathon à Mons. Jean-Baptiste, j'espère que je ne te pervertissais pas trop en t'enrôlant à l'escalade. Davide, merci.

Certains sont arrivés en renfort en cours de route : Damien, Quentin, Jean-François (merci pour les fameuses parties de Among Us pendant les temps difficiles de rédaction confinée), Guilhem (en réalité nous sommes arrivés en même temps. Félicitations encore pour le mariage !). Merci aussi aux anciens qui m'ont montré la route : Valérian (le plus grand des chanteurs et danseurs), Elhadi (grand amateur de cirque de Noël), Seb, Teddy, Florent, Fabrice, John, Sameh, Nicolas, Arcady et d'autres que j'oublie sûrement.

Merci ensuite aux fantassins sans qui tout cela n'aurait pas été possible : Axel, Maxime, Hugo, Matthieu, Aurélien et tous les autres. Merci pour les fameuses sorties barbecue à La Ramée. Merci encore à Hugo pour le travail formidable que tu as effectué. Merci à Matthieu et Aurélien pour les (ou la, je ne me souviens plus) sorties natation sauvage mais aussi

pour les soirées danse (ça, c'est surtout pour toi Aurélien). Merci aussi à Avi et Thibault qui contribuent récemment à poursuivre mon travail.

Merci au bureau 363 : le bureau du fond, le bastion que j'ai réussi à garder jusqu'au bout. Merci aux premiers camarades : Tanguy (pour tes rythmes endiablés et le bol d'air frais que tu me donnais en aérant le bureau tous les matins, même en hiver. Surtout en hiver !) et Alex (pour tous tes précieux conseils lors de mes premiers pas dans l'équipe et pour le meilleur des enseignements que j'ai pu faire : les TP Unity !). Maroun, d'abord, merci à toi d'avoir partagé ce bureau avec moi pendant la majeure partie de ma thèse mais surtout, merci à toi et Patri pour les journées de ski et de parc accrobranche. J'espère qu'on en fera d'autres. Patri, you were of course also part of the office 363. I hope you enjoyed my chair !

Enfin, merci à l'ordre 363 ! C'est grâce à vous : Kristell, Augustin et Walid, que j'ai pu mettre en forme mes plans machiavéliques et révéler mon véritable talent. Oui, vous étiez mon inspiration. Ou bien, vous étiez la cause de toutes ces âneries ? C'est avec beaucoup d'émotion que je quitte la grande institution de prestige que représente l'ordre. Non ! Je ne le quitte pas vraiment, il sera à jamais dans mon cœur tout comme cet hymne !

Davide, mais non, je ne t'ai pas oublié ! Merci à toi et à nos rivalités de bureaux qui se sont transformées en amour inconditionnel. È sicuramente grazie al sangue siciliano !

Merci aux copains de Master Victor, Ghita, Clément, Seb, Ronan, Adrien, plus particulièrement à Tristan et Marine pour les après-midi disco avec les TurtleBots ! Merci surtout à Evgeny (alias Genishka) : tovarishch ! Le premier camarade de master, l'homme simple !

Merci à Jesse Groenen et Florence Pettinari-Sturmel pour votre bienveillance et votre investissement dans les Parcours Spéciaux : mes premières années d'études qui m'ont donné goût à la recherche. Merci à la première promo des Parcours Spéciaux pour les nombreuses soirées que nous avons faites ensemble : Oriol, Cloé, Thibault, Nathanaël, Jérôme, Jonathan, Baptiste, Manon, Romain et les autres.

Ensuite, merci aux compagnons de "grimpe" qui m'ont suivi tout au long la thèse avec plus ou moins d'assiduité et parfois par hasard : Rémy, Sonny, Florian, Anthony, Audren, Dimitri et tous ceux qui sont venus s'essayer à ce sport exceptionnel. Merci à tous les membres de Swingood et notamment à Yvonnic, Ophélie, Fabrice et Gaelle pour les soirées et les stages de West Coast. Un grand merci à MGMU, mes camarades de route ces dernières années : Sandra, Gazo, Sophie, Enzo, Gach, Adrien, Romain, Clément et celui que j'ai déjà nommé. Toujours un plaisir de converser avec vous même si j'ai souvent eu la flemme de tout lire.

Merci à mes amis de toujours pour les semaines de vacances à la plage, à domicile ou à l'étranger. Mais aussi les soirées et les séances de muscu sur Discord. LE covid nous aura à nouveau rapprochés d'une certaine façon. Merci à l'Alliance SPX : Raphaël (GG pour ta thèse, t'as compris), Paulin (c'est pour quand le serveur Minecraft là ? !), Guillaume (l'amoureux des bêtes mais amateur d'art avant tout !), Lawrence (courage pour ton concours !), ainsi que Fabien et Clément.

Enfin, merci à mes amis d'enfance (oui le lycée c'est encore l'enfance !) : le club cirque avec Nicolas, Ashita et tous les autres, mais aussi les tennis et les semaines à la montagne avec Guillaume, Julien et Pierre. Les circonstances de la vie nous ont séparés géographique-

ment ces dernières années mais nous avons continué à nous voir et j'espère encore partager beaucoup de moments avec vous à l'avenir.

A mi familia, papá, mamá, gracias por siempre haberme apoyado a lo largo de mi tesis y todas las otras etapas de mi vida que hicieron la persona que soy hoy. Gracias por haberme preguntado todo este ultimo año cuándo sería mi defensa de tesis. No teníais porque preocuparos, por supuesto que os lo hubiera dicho. Sara, hermanita, gracias por ser la persona que eres con la que puedo compartir tantas cosas. Estoy seguro de que encontrarás un trabajo que te guste en el que podrás desarrollar todos tus talentos.

Je ne suis sûrement pas la personne la plus facile à vivre et cette dernière année n'a pas été des plus simples. Merci à toi Mathilde de m'avoir accompagné durant ces presque huit années maintenant. J'espère que l'on continuera ce chemin ensemble aussi longtemps que possible.

당신은 내 사랑의 경보를 울리고 있습니다. 사랑해

Enfin, tu n'étais pas là quand j'ai commencé ma thèse. Sans toi notre vie n'est pas complète. Quand tu es arrivé, tu nous as rempli d'amour et de poils. Avec toi notre appartement est saccagé et nos meubles resculptés. Merci à toi Yuri.

Contents

Introduction **1**

Contributions 1

Document organization 2

I Context **5**

1 From Robotics Towards Complex Systems **7**

1.1 The Future of Robotics 7

1.2 Towards Industry 5.0 8

1.3 Self-Adaptive Systems Challenges 9

1.4 Sociotechnical Environments 9

1.5 Designing in Complex Systems 10

1.6 Thesis Objectives 11

II State of the Art **13**

2 Machine Learning Foundations **15**

2.1 Supervised Learning 16

2.2 Unsupervised Learning 16

2.3 Semi-Supervised Learning 17

2.4 Learning by Rewards 18

2.5 Learning by Demonstrations 19

2.6 Probabilistic Learning 20

2.7 Synthesis 22

3 Towards Lifelong Learning **23**

3.1 Transfer Learning 24

3.2 Multi-Task Learning . . . 27
3.3 Online Learning . . . 28
3.4 Deep Learning . . . 29
3.5 Synthesis . . . 32

4 Endogenous Learning **35**
4.1 Developmental Learning . . . 36
4.2 Meta-Learning . . . 38
4.3 Few Shot learning . . . 39
4.4 Context Learning . . . 40
4.5 Endogenous Learning Challenges . . . 45
4.5.1 Global Synthesis . . . 45
4.5.2 Context Learning Shortcomings . . . 47

III Contribution **49**

5 Endogenous Learning Principles **51**
5.1 Endogenous Learning Objectives . . . 53
5.2 Endogenous Context Learning Objectives . . . 53
5.2.1 Exploration Objectives . . . 54
5.2.2 Model Objectives . . . 55
5.3 Neighborhood Definitions . . . 56
5.4 Learning Inaccuracies . . . 59
5.4.1 Communication in Context Agent Pairs . . . 59
5.4.2 Exploration Inaccuracies Detection . . . 60
5.4.2.1 Detection Distances . . . 61
5.4.2.2 Conflict NCS Detection . . . 62
5.4.2.3 Concurrency NCS Detection . . . 62
5.4.2.4 Complete Redundancy NCS Detection . . . 62
5.4.2.5 Partial Redundancy NCS Detection . . . 63
5.4.2.6 Range Ambiguity NCS Detection . . . 64
5.4.2.7 Incompetence NCS Detection . . . 64
5.4.3 Model Inaccuracies Detection . . . 67
5.4.3.1 Model Ambiguity NCS Detection . . . 67
5.4.3.2 Model Discontinuity NCS Detection . . . 68

5.4.4 Exploration Inaccuracies Resolution . . . 69
5.4.4.1 Conflict and Concurrency NCS Resolution . . . 69
5.4.4.2 Range Ambiguity and Model Discontinuity NCS Resolution 70
5.4.4.3 Incompetence NCS Resolution . . . 70
5.4.4.4 Complete Redundancy NCS Resolution . . . 71
5.4.4.5 Partial Redundancy NCS Resolution . . . 72
5.4.5 Model Inaccuracies Resolution . . . 73
5.4.5.1 Model Ambiguity NCS Resolution . . . 73
5.5 Learning Inaccuracies Computation . . . 73
5.6 Cooperative Neighborhood Learning . . . 75
5.7 Synthesis . . . 77

6 Endogenous Lifelong Learner by Self-Adaptation ELLSA 79
6.1 Endogenous Context Learning . . . 81
6.1.1 Percept Agents Cycles . . . 83
6.1.2 Learning Cycles . . . 85
6.1.2.1 Algorithm Description . . . 85
6.1.2.2 Learning NCS . . . 87
6.1.2.3 Learning Parameters . . . 92
6.1.3 Exploitation Cycles . . . 93
6.1.3.1 Algorithm Description . . . 93
6.1.3.2 Exploitation NCS . . . 93
6.1.3.3 Exploitation with Sub-Perceptions . . . 94
6.1.3.4 Exploitation Parameters . . . 94
6.2 Criticalities . . . 95
6.2.1 Learning Criticality . . . 95
6.2.2 Exploitation Criticality . . . 95
6.2.3 Metrics . . . 96
6.2.3.1 Generalization . . . 96
6.2.3.2 Experience . . . 96
6.2.3.3 Model . . . 98
6.2.3.4 Proximity . . . 100
6.2.4 Synthesis . . . 101
6.3 Lifelong Context Learning . . . 102
6.3.1 Motivations . . . 102

6.3.2 Weighted Model Update . . . 102
6.3.3 Artificial Learning Situations Generation . . . 103
6.3.4 Local Models Update Scenarios . . . 104
6.3.4.1 Immature and Mature Models . . . 104
6.3.4.2 Exogenous and Endogenous Learning . . . 105
6.3.5 Synthesis . . . 105
6.4 Learning Strategies . . . 106
6.4.1 Active Learning . . . 106
6.4.2 Self-Learning . . . 108
6.4.3 Synthesis . . . 110
6.5 Learning Reflections . . . 111
6.5.1 Context Transfer Learning . . . 111
6.5.1.1 Context Domain Adaptation . . . 111
6.5.1.2 Context Inductive Transfer . . . 112
6.5.2 Context Multi-Task Learning . . . 112
6.5.3 Context Reinforcement Learning . . . 113
6.6 Synthesis . . . 115
6.6.1 Endogenous Learning Principles . . . 115
6.6.2 ELLSA . . . 115
6.6.3 Parameters, Distances, Metrics and Criticalities . . . 117

IV Experimentations and Validation 119

7 Experiments on Mathematical Models 121
7.1 Parameters . . . 122
7.2 Experimental Metrics . . . 123
7.3 User Interface . . . 127
7.4 Learning Inaccuracies NCS . . . 128
7.5 Active Learning VS Self-Learning . . . 134
7.5.1 Non Linear Continuous Problem . . . 134
7.5.2 Non Linear Discontinuous Problem . . . 139
7.5.3 Multi-Model . . . 143
7.6 Lifelong Learning . . . 146
7.6.1 Noisy Problem . . . 146
7.6.2 Lifelong Exploitation Problem . . . 151

7.7 Few Learning Situations . . . 155
7.8 Scalability . . . 159
7.9 Transfer Learning . . . 165
7.10 Synthesis . . . 168

8 Robotic Problems **171**
8.1 Background . . . 171
8.1.1 Direct Kinematics . . . 172
8.1.2 Inverse Kinematics . . . 173
8.2 Inverse Kinematics Learning . . . 174
8.2.1 Experimental Metrics . . . 174
8.2.2 Centralized Control . . . 176
8.2.3 Distributed Control . . . 183
8.3 Synthesis . . . 191

V Conclusions and Perspectives **193**

9 Conclusions and Perspectives **195**
Conclusions . . . 195
Contributions . . . 201
Perspectives . . . 202

VI Appendices **205**

Bibliography **207**

List of Figures **223**

List of Tables **229**

List of Algorithms **231**

Glossary **233**

Introduction

In a near or distant future, robots will be able to evolve and learn as we humans do from our environment and the interactions it induces. However, there is still a long way to go. At the moment, robots are a key component of the worldwide industry and they are gradually expanding into personal services. Their learning competences are still narrow such as most *Artificial Intelligence (AI)* common technologies. The environments in which robots and *AI* are immersed are constantly increasing in complexity. This raises the question of how intelligent systems should learn in this dynamic. A lead is to provide learning systems with endogenous abilities: internal mechanisms enabling them to self-assess what they learn and self-enhance themselves. We call such mechanisms *Endogenous Feedback*.

The environment that surrounds us is defined by its unpredictability, openness, dynamicity and heterogeneity [Nigon et al., 2017]. All theses properties must be taken into consideration when designing a learning system made to interact with it. The challenge of this thesis is to design a self-learning mechanism implementing *Endogenous Learning* using *Endogenous Feedback* and dealing with all the characteristic of its environment.

Contributions

In this thesis, I define and characterize *Endogenous Learning* by identifying its motivations and how they are declined for a learning mechanism. My contribution is the implementation of *Endogenous Learning* with the paradigm of *Context Learning*: *Endogenous Context Learning*. It is a distributed learning approach relying on an *Adaptive Multi-Agent System*. This paradigm enables to identify and solve learning deficiencies using endogenous mechanisms. It results in a local cooperation of knowledge fragments that are agents with encapsulated models.

These agents are a part of a larger learning mechanism that I design: *ELLSA* for *Endogenous Lifelong Learner by Self-Adaptation*. *ELLSA* integrates *endogenous*, *agnostic*, *lifelong*, *online*, *self-observation* and *generalization* learning properties. It provides an active learning mode to request specific *learning situations* when they are available, and a self-learning mode to learn with fewer *learning situations*.

I assess the implemented learning mechanism on several learning experiments to highlight its strengths and limitations in relation with its learning properties. A robotic experiment is conducted with *ELLSA* to present how it performs on a concrete learning problem.

Document organization

The manuscript is structured as follows:

- Chapter 1: I present the context of this work. I argue how the future of robotic applications is related to the dynamic sociotechnical environment in which they are embedded. I defend how self-adaptation is a central concern in this scope and what are the future challenges for designing intelligent applications in complex systems. The field of robotics is very broad, the focus is made on systems that adapt to their environment constraints and not on mechanics or signal treatment. I finish by enumerating the objectives of the thesis.
- Chapter 2: I present the basis of *Machine Learning* in the literature. I explain how artificial learning works and what are the different degrees of supervision. I differentiate two main kinds of artificial learning that are rewards and demonstrations. I briefly present probabilistic approaches of learning. I finish starting a first reflection about these approaches according to the objectives of my thesis.
- Chapter 3: I continue the reflection on artificial learning by focusing on *Lifelong Learning*. I define what makes us human beings lifelong learners and what are the properties that are sought to be replicated in artificial learners. I present learning paradigms that are related to *Lifelong Learning*: *Transfer Learning*, *Multi-Task Learning*, *Online Learning* and *Reinforcement Learning*. I develop a more general questioning on the major protagonist of artificial learning today, *Deep Learning*. I finish updating the reflection about the presented approaches according to the objectives of my thesis.
- Chapter 4: I extend the reflection on artificial learning by focusing on a more psychological view of learning. I present learning from the Constructivism point of view which gave birth to *Developmental Learning*. Then, I explore other learning paradigms that are related to *Developmental Learning*: *Meta-Learning* and *Few Shot Learning*. I introduce the *Context Learning* paradigm upon which the work of this thesis is based. The use of *Adaptive Multi-Agent Systems* is defended by presenting their properties which are in adequacy with the motivations of my thesis.
- Chapter 5: I detail the construction of *Endogenous Context Learning*. It consists in the identification of learning hypotheses that allows to identify *learning inaccuracies*. The *learning inaccuracies* are learning defects that can be detected with local agents negotiations. To add local communication between the agents of a Multi-Agent based learning mechanism, the concept of *neighborhood* is introduced. *Learning inaccuracies* are characterized to determined how to detect and solve them. Their processing order is explained according to their priority. *Cooperative Neighborhood Learning* is presented. It is a mechanism allowing to generate internal *learning situations* to enhance the model of agents through local communication.
- Chapter 6: The functioning of the designed learning mechanism is presented: *ELLSA* for *Endogenous Lifelong Learner by Self-Adaptation*. The optimization of agents activation is detailed and the processes of learning and exploitation cycles are described. The *learning criticality* and the *exploitation criticality* are introduced, they enable to include performance, generalization and experience in the learning and exploitation processes. A

mechanism for dynamically updating learning models in a lifelong setting is proposed. The *Active Learning Strategy* and the *Self-Learning Strategy* are introduced. Their use depends on the availability of learning data. The chapter is ended by an analysis and a positioning about *AI* challenges.

- Chapter 7: Several experimentations made on the learning mechanism *ELLSA* are presented. The learned models are mathematical functions with properties of linearity, non linearity, continuity and discontinuity. These experimentations allow to highlight the strengths and weaknesses of the system according to the objectives presented in section 1.6 (*Endogenous Feedback*, *Agnosticity*, *Lifelong Learning*, *Online Learning*, *Self-Observation*, *Knowledge Generalization*, *Scalability*, *Any Data Amount* and *Explainability*).
- Chapter 8: I present several robotic experimentations made with the learning mechanism ELLSA. The learning is applied to the problem of *Inverse Kinematics* which is classical in the robotic community. The application scenario is the control learning for multi-joint robotic arms. These experimentations allow to highlight the properties of the system according to the objectives presented section 1.6 in a concrete problem.
- Chapter 9: In this final chapter, the conclusions and perspectives are provided in relation to the initial objectives of this thesis (section 1.6). The contributions are declined in 3 categories: *Context Learning*, *Machine Learning* and Robotics. The perspectives present exploration tracks concerning additional adaptation and optimization for *Context Learning*, leads to advance certain challenges of *Machine Learning* and *AI*, and possible extensions of the work carried out on robotics.

In order to facilitate reading, a glossary including most of the technical terms and notations is available at the end of the manuscript (Part VI). If you are reading this thesis in digital version, you can access the glossary by clicking on the terms that are referenced.

Lifelong Learning by Endogenous Feedback
Application to a Robotic System

Context

1 From Robotics Towards Complex Systems

In this chapter, I present the context of this work. I argue how the future of robotic applications is related to the dynamic sociotechnical environment in which they are embedded. I defend how self-adaptation is a central concern in this scope and what are the future challenges for designing intelligent applications in complex systems. The field of robotics is very broad, the focus is made on systems that adapt to their environment constraints and not on mechanics or signal treatment. I finish by enumerating the objectives of the thesis.

ROBOTS have revolutionized the industry and they will revolutionize our everyday life. However, the world in which humans live is complex and constantly changing. The robots of tomorrow will be intrinsically linked to adaptive systems that adjust to the constraints of their environment.

1.1 The Future of Robotics

The numbers of robots in our society is growing and and it will never stop increasing [Torresen, 2018]. From robotic arms to vacuum cleaners, many of them are already present and widely adopted in many homes and factories. We can categorize them into industrial robots which will be the majority and service robots.

- **Industrial robots**: they consist of articulated arms with a certain number of degrees of freedom and a terminal tool. The minimal number of degrees of freedom is 6 to move and to orient the tool in all directions. Industrial robots are commonly found in manufacturing chains to perform repetitive tasks.
- **Service robots**: they are semi- or fully autonomous robots that perform tasks for humans or equipment. They include personal and domestic robots (vacuum cleaners, toys or autonomous vehicles) and professional robots for different fields (space exploration, agriculture, medical care, surveillance ...).

Fig. 1.1 shows the industrial revolutions and their associated breakthroughs. Mechanical power from water and steam gave birth to the 1st Industrial Revolution. The 2nd Industrial

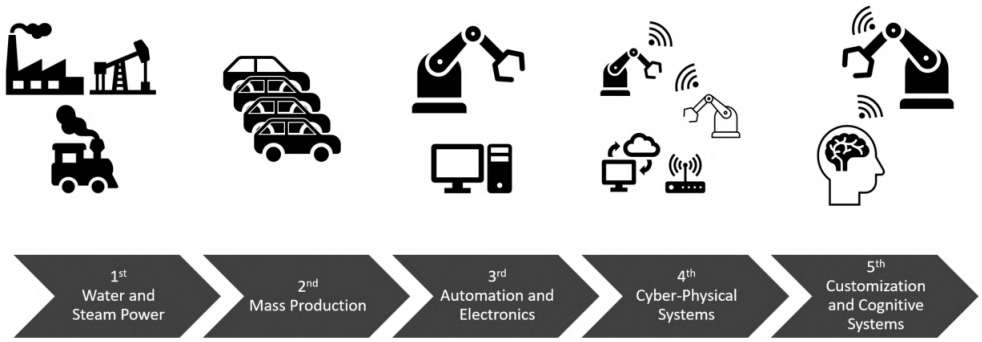


Figure 1.1 – Industrial Revolutions

Revolution came with electrical energy and the appearance of assembly lines. Automation and industrial robotics allowed the 3rd Industrial Revolution in the 1970s. The Internet of Things (IoT) and cloud computing brought the current 4th Industrial Revolution. With it, flexible manufacturing and cyber-physical systems emerged. Their goal is to provide real-time interfaces between virtual and physical worlds. Even if the 4th Industrial Revolution is still young and its challenges are not yet all accomplished, the 5th Industrial Revolution is coming [Nahavandi, 2019]. It aims at a massive customization putting human intelligence in the loop using autonomous cognitive manufacturing. For this revolution, *AI* and big data are the main innovative domains that are able to reach the objectives.

1.2 Towards Industry 5.0

Industry 4.0 emphasizes on improving efficiency ignoring the resulting human costs nor the environmental costs. As we know, the transition towards the replacement of human labor by its automation with robots promises to be complicated [Granulo et al., 2019]. It is in this sense that Industry 5.0 focuses on pairing humans and machines, to convert robots into human companions or cobots (collaborative robots). Cobots are like apprentices, they watch and they learn giving humans a feeling of satisfaction working alongside cobots.

One of the keys of Industry 5.0 is the design of intelligent autonomous systems that learn and make decisions under unforeseen circumstances [Nahavandi, 2019]. In particular, in the context of personalization, *Transfer Learning* is a critical aspect as it would allow to transfer knowledge from a virtual system to its physical twin.

Intelligent autonomous systems will also need to cooperate and learn in an adaptive manner as humans do it [Wickens et al., 2015, Georgé et al., 2011]. Cognition systems need to improve in order to self-adapt in an ever-changing situations. Systems with such properties are called *Self-Adaptive Systems (SAS)*. The smart factory of Industry 4.0 implementation is currently facilitated using *SAS* and self-organized reconfiguration [Wang et al., 2016].

SAS are well suited for the challenges of the Industry 5.0 [Skobelev and Borovik, 2017]. *SAS* adapt their behavior during runtime to face changing contexts and environments [Bernon et al., 2002, Picard and Gleizes, 2002, Capera et al., 2003, Kephart and Chess, 2003].

With the Industry 5.0, autonomous systems will need increasing functionnalities and

it will no longer be beneficial to specify all behaviors at design because there will be too many potential states. This is why the popularity of *Self-Adaptive Systems* is growing and is expected to evolve into mainstream adopted solutions [Wong et al., 2021].

1.3 Self-Adaptive Systems Challenges

Self-Adaptive Systems (*SAS*) are well suited for the Industry 5.0 but they also face several challenges.

As we said, *SAS* focus on guarantees under uncertainty [Weyns, 2018]. *SAS* need to deal with uncertainty as modern software systems are increasingly embedded in open worlds that are constantly evolving (changing environment, users behaviors and additional requirements). Anticipating this evolution is almost impossible during the design of such applications, they need to self-adapt during their lifetime [Bernon et al., 2002, Baresi and Ghezzi, 2010]. The system has to adapt to its task and no longer needs to be designed for a specific one. This results in different sources of uncertainty that must be taken into account: uncertainty related to the system suitability, uncertainty related to the system goals, uncertainty related to the system executions context and uncertainty related to humans interacting with the system [Mahdavi-Hezavehi et al., 2016].

Other challenges of *SAS* are the decentralization of adaptation, dealing with changing goals and dealing with complex types of uncertainties [Gleizes, 2012, Weyns, 2018]. Centralization is not a sustainable approach with the increasingly large and complex systems that need control. Changing goals dynamically is a step beyond dealing with uncertainty. Allowing a complex system to self-adapt its strategy helps it recover from run-time abnormalities [Georgé et al., 2011, Gerostathopoulos et al., 2019]. Dealing with changing goals would mean that the system could synthesize and verify new models by itself. Complex types of uncertainties can be model inadequacy, model bias or model inaccuracy. To deal with uncertainty sources, the use of feedback loop mechanisms is required [Muccini et al., 2016].

Current challenges are focused on the exploitation of *Artificial Intelligence* in order to perform complex tasks as *AI* techniques also focus on uncertainty. Machine Learning techniques can overcome *SAS* challenges by learning from some experiences. Machine learning is able to address the uncertainty in *SAS* by learning new adaptation rules dynamically and modifying the existing rules [Saputri and Lee, 2020]. Then, the step after dealing with the anticipated changes (known unknowns) is dealing with unanticipated changes (unknown unknowns) [Oreizy et al., 1998].

1.4 Sociotechnical Environments

The Industry 5.0 and the *Self-Adaptive Systems* are defined by the Sociotechnical Environment in which they are situated. This environment in which the systems operate is defined using several assumptions that we present here inspired by Russel & Norvig [Russell and Norvig, 2016].

- **Non-determinism**. The environment potentially contains a multitude of systems that

act simultaneously, the next state of the world is not only determined by the system's current state.

- ▷ **Locality**. The perception of the world is reduced, so is the effect of actions. An application is neither omniscient nor omnipotent and is therefore a located system.
- ▷ **Continuity**. The values of perceptions and actions evolve in a continuous space and perceived values are real values.
- ▷ **Dynamicity**. The environment evolves while the system is processing its tasks.
- ▷ **Uncertainty**. The world is not ideal but material. It is therefore perceived by sensors that are subject to noise and breakdowns.

The coupling of *Self-Adaptive Systems* and their Sociotechnical Environment defines a Complex System with additional properties that have to be dealt with [Nigon et al., 2017]. Among these, we found openness and heterogeneity. Complex Systems are open in the sense that entities or applications can appear and disappear at any time. They are heterogeneous because all interacting components are not necessarily identical in terms of communication, observation and control.

1.5 Designing in Complex Systems

Designing an intelligent application capable of evolving in the Sociotechnical Environment and facing the future challenges of the Industry 5.0 is a tough work. I categorize the designing of an autonomous system or robot focusing on the task it has to perform. An analogy with the industrial revolutions is made as they led to more and more complex and abstract tasks.

- ▷ **Known Task**. When the task is well known in a constrained environment, the designer can implement an *ad hoc* application where all situations can be defined. It was the case for the Industry 3.0. The Industry 4.0 starts deviating from this context with the need for flexibility. With the complexity of future applications and environments, anticipating all situations will no longer be a possible solution [Baresi and Ghezzi, 2010, Gleizes, 2011]. *Ad hoc* systems badly deal with unknwon situations and dynamics. An intelligent application must adapt to its environment by learning from its experiences, this is why learning is an inevitable path.
- ▷ **Measurable Task**. If the task is measurable, a common approach is to use Learning by Rewards. The learner explores its environment by acting in it and it is rewarded or punished to guide its learning. Depending on the complexity of the task, the reward can be also be difficult to design *a priori* and it will also need to evolve with the changing environment.
- ▷ **Observable Task**. A task is observable when an operator can control the system or the robot to show it how to behave. In this case, we call it Learning by Demonstrations. It fits to the spirit of the Industry 5.0 where the human is put back among the Cyber-Physical Systems to show its expertise and collaborate with robots [Nahavandi, 2019]. Yet, demonstrations may be biased and may not reflect the knowledge gained from lifelong experiments.

- **Unknown Task**. One of the challenges of robotic interactive systems is to learn without having any intrinsic knowledge of the tasks they will have to solve. To do so, they need internal curiosity mechanisms to avoid biases, maximize genericity and provide adaptation to their environment [Oudeyer et al., 2014]. Such a system is also called an agnostic system [Kearns et al., 1994]. To design it, it is essential to generate knowledge through processes and actions that are purely internal to the system.

This thesis focuses on the last setting, the design of interactive learning systems without prior knowledge on any task. A step towards this goal, and the main objective of this thesis, is the generation of *Endogenous Feedback*. They are internal interactions whose goal is to improve learning.

1.6 Thesis Objectives

The goal of this thesis is to design a self-adaptive learner that fits the challenges of the Industry 5.0 and deals with the characteristics of Sociotechnical Environment. We present the main objectives that we want our learning system to achieve.

- ***Endogenous Feedback***. The mechanism needs to learn from *Endogenous Feedback*. It can learn from external entities but also from endogenous experiences. In the second case, knowledge must be generated through processes and interactions that are internal to the system: between its known experiences and between its actions and perceptions through active learning or self-learning. We call this type of process: "learning by endogenous feedback". In Economics, we find homologous definitions: learning by doing [Arrow, 1962] but also endogenous learning [Creane, 1995]. The contribution according to this objective is detailed in chapter 5, validated in section 7.4 and evaluated all along the experimentations in the chapters 7 and 8. This objective was the subject of the following publication [Dato. et al., 2021b].
- ***Agnosticity***. The learning system is agnostic towards the task it has to learn. The learning method is not specialized to one task. It can learn any task using the same approach without making preliminary assumptions [Kearns et al., 1994]. It has to focus on the task of learning itself. The learning approach can be applied to any interactive system possessing sensors and actuators i.e. a system capable of interacting with its environment, no matter the number of devices it possesses nor the nature of them. This objective is assessed through the different experiments conducted chapters 7 and 8.
- ***Lifelong Learning***. The system never stops learning. It learns in an incremental way to be able to improve its previous knowledge and it decides if a new experience is worth learning or not. If tasks are related through some underlying structure, the learning system may share knowledge between these tasks to improve learning performance. At any time, it may be asked to solve a problem from any previous learned task, and to also maximize its performance across all learned tasks [Thrun and Mitchell, 1995]. The contribution according to this objective is presented in section 6.3 and evaluated in sections 7.6 and 7.9.

- ***Online Learning***. The learning process is executed online which means that it can sequentially receive new data and be asked for decision making. The processing of this new data does not interfere with the exploitation of previously learned tasks [Rosenblatt, 1957] unlike offline learning mechanisms that receive the training data all at once. The learning process of this work intrinsically uses online learning and is detailed in section 6.1.
- ***Self-Observation***. The system has the capability of *Self-Observation* to self-adapt. It is able to provide itself feedback on what it knows, how precisely it knows it and what it could know better. In Psychology, we found the analogous principle of introspection. It is the examination of one's conscious thoughts and feelings [Schultz, 2013]. Self-observation is also an inherent property of this thesis learning approach. It is used for the *Endogenous Feedback* contribution in chapter 5 but also during the experiments in chapter 7.
- ***Knowledge Generalization***. The learning mechanism generalizes while it learns. It recognizes if a new experience is similar to a previous one and it establishes connections between them permitting it to improve a previously learned task and to learn faster the new one. This is also called inductive transfer [Pan and Yang, 2010]. Generalization can be thought of as interpolation between known examples, and extrapolation, which requires going beyond a space of known training examples [Marcus, 1998]. The contribution according to this objective is presented in section 6.2 and specifically evaluated in chapter 7 across all experiments.
- ***Scalability***. Future robotic or interactive systems promise to be composed of large amounts of sensors and actuators [Revzen and Koditschek, 2017]. Natural intelligence is multi-dimensional and so will be generalized *Artificial Intelligence* [Gardner, 2011]. Learning must be applicable to large-scale complex learning systems. Its complexity must be linearly dependent on the number of sensors or actuators. This objective is addressed all along this work and it is evaluated in section 7.8.
- ***Any Data Amount***. Learning from a lot of learning examples is usual in machine learning but learning from few situations still remains a challenge [Bendre et al., 2020]. Learning must be performed from any sample set size. New behaviors should be obtained from a few training examples or from a lot. Learning should be done in every situation the learning system is facing, regardless of the amount of examples it has access to. The contribution that concerns this objective is detailed in sections 5.6 and 6.4.2 and it is evaluated in section 7.7.
- ***Explainability***. The way the system takes decisions is not hidden in a black box. The reason that brought the learning mechanism to choose a behavior instead of another one during an exploitation is explainable and observable by the system itself so that it can improve itself [Samek et al., 2017]. This objective is addressed by the implementation of a user interface described in section 7.3.
- ***Robotic Application***. The last objective of this work is to apply the implemented learning mechanism to a robotic application. The contribution concerning this goal is detailed and evaluated in chapter 8. This objective was the subject of a publication [Dato. et al., 2021a].

State of the Art

2 Machine Learning Foundations

In this chapter, I present the basis of Machine Learning in the literature. I explain how artificial learning works and what are the different degrees of supervision. I differentiate two main kinds of artificial learning that are rewards and demonstrations. I briefly present probabilistic approaches of learning. I finish starting a first reflection about these approaches according to the objectives of my thesis

MACHINE Learning is a vast field where one can easily get lost in front of the immensity of the different works and approaches. I will first present a definition of learning in *AI* and what are the main classical techniques keeping in mind the goals of this work. Then, I will seek for my thesis objectives (section 1.6) in more recent learning approaches to identify the current advances and challenges.

Learning in *AI*, such as learning in our society, involves a learner and a teacher. The learner can be an intelligent application, a robot, a self driving car ... It is an entity that is designed to process examples from a teacher and transform them into knowledge to generalize and be able to make decisions for unforeseen situations when the teacher is not present anymore. The knowledge or learning models, as well as the teacher or the oracle, can have many forms. This is what differentiates each learning fundamental techniques. A learning mechanism has inputs, they are its perceptions, how it perceives its environment. And it has outputs, they are its predictions. They are the result of its perceptions processed by its models (Fig. 2.1).

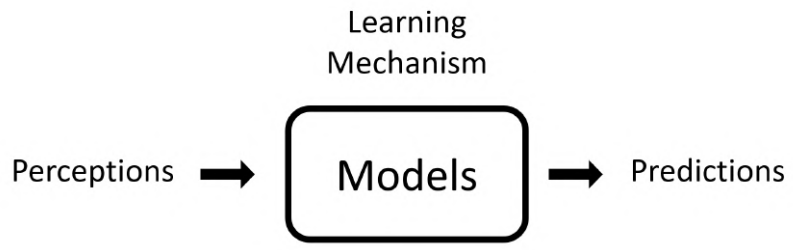


Figure 2.1 – Learning Mechanism Inputs and Outputs.

In the following and in all the chapters of the state of the art, learning approaches are introduced and a discussion is conducted with respect to the learning objectives described sec-

tion 1.6 : *Endogenous Feedback*, *Agnosticity*, *Lifelong Learning*, *Online Learning*, *Self-Observation*, *Knowledge Generalization*, *Scalability*, *Any Data Amount*, and *Scalability*. At the end of each chapter, a synthesis is made and a summary table is drawn.

2.1 Supervised Learning

Supervised Learning is an essential method in *Machine Learning*. For this technique the teacher takes the form of labeled examples associated to perceptions. This is what we call the training data or the learning data. Learning can be seen as mapping input observations to output labels. If outcomes are discrete, the mapping is called **classification**. If it is continuous, the problem becomes predicting an output as close as possible to the target one, it is **regression**. Common learning models in the literature are k-nearest neighbors [Fix and Hodges Jr, 1951], decision trees [Breiman et al., 1984], support vector machines [Vapnik, 2013] or Artificial Neural Networks [Yegnanarayana, 2009]. They can be used for classification, regression, and ranking problems. The labeled data reduces the search space during the learning process, but it also limits the learning to the training samples. The majority of these approaches are offline and the labeled data are collected depending on the task to solve.

Recent approaches that we will present and develop later focus on *Multi-Task Learning*, *Transfer Learning* or *Online Learning*. Ruvolo and Eaton [Ruvolo and Eaton, 2013] performed Online *Multi-Task Learning* and *Transfer Learning* thanks to a general algorithm that supports different base learners (linear regression and logistic regression). *Curriculum Learning*, a step towards *Lifelong Learning*, proposes a supervised *Multi-Task Learning* mechanism where the order in which the learner solves the set of tasks influences the overall performance [Pentina et al., 2015].

Discussion

Recent researches in this domain roughly match what we are looking for in this theses. There is no mention of any mechanism related to *Endogenous Feedback* in any form. *Supervised Learning* is agnostic in relation to the data because it is not specialized in one specific task or domain. Some works are starting to focus on *Lifelong Learning* and *Online Learning* problems. Regarding *Self-Observation*, it is not a concern of classical *Supervised Learning*. The work about *Knowledge Generalization* is in progress with the appearance of *Multi-Task Learning* and *Transfer Learning* that we will present in the next chapter. The *Scalability*, the needed data amount and the *Explainability* are usually model dependent so it is difficult to give any arguments about it.

2.2 Unsupervised Learning

For this approach there is no teacher. *Unsupervised Learning* methods make predictions for all unseen points and they only receive unlabeled training data. *Unsupervised Learning* uses only unlabeled data for data aggregation (clustering) or dimensional reduction

problems [Hastie et al., 2009]. Usual methods are k-means [Hartigan and Wong, 1979], Self-Organizing Maps (SOM) [Kohonen, 1998], Hopfield networks [Hopfield, 1982] and Boltzmann machines [Ackley et al., 1985]. A recent example of *Unsupervised Learning* application is the assessment of video similarity [Papoutsakis and Argyros, 2019].

Discussion

These mechanisms are purely endogenous to the learning mechanism because they seek correlations in amounts of data without external labeled information. *Unsupervised Learning* is related to learning with *Endogenous Feedback* by the use of unlabeled data. However, it is not focused on enhancing any learning process. It is data agnostic because no prior information is needed concerning the application domain. These methods are useful to organize large amount of learned data through *Lifelong Learning* and lessen *Scalability* problems [Henaff et al., 2011]. However, the *Unsupervised Learning* approach alone cannot overcome the challenges of *Lifelong Learning* [Silver et al., 2013]. In the recent years, little work is concentrated on online implementation of *Unsupervised Learning* [Nunes and Demiris, 2019]. There is some kind of *Self-Observation* toward the training data because it is an internal reflection on it. It is *Knowledge Generalization* when clustering is performed as data is aggregated but not in the sense of incremental experiences and their respective relations in an evolving environment. *Unsupervised Learning* is specialized in large amounts of data so it makes little sense to consider using it with little data. *Unsupervised Learning* seems promising according to *Explainability* concerns because it allows to build representations of correlated data.

2.3 Semi-Supervised Learning

In *Semi-Supervised Learning*, the learner trains with samples consisting of both labeled data and unlabeled data to make predictions for all unseen points. The goal is to use the additional unlabeled data to achieve better performances than classical *Supervised Learning*. For instance, [Lampert et al., 2009] performs object classification systems without training images but with human-specified high-level description. [Palatucci et al., 2009] focused on predicting novel classes that were omitted from a training set using common features to both novel classes and the training set. It can be seen as seeking to achieve better performance by combining unlabeled and labeled data rather than using exclusively labeled or exclusively unlabeled data [Li et al., 2017]

This approach tends towards *Active Learning*, which consists of letting the learner chose its training data to lessen the examples in comparison to simple *Supervised Learning*. We can find three mains settings in the literature: membership query synthesis, stream-based selective sampling and pool-based active learning [Settles, 2009]. With membership query synthesis, the learner can request any situation in the perceptions space, self-generated or not. A problem with this scenario is that queried situations may not always be labeled by an oracle. Stream-based selective sampling or sequential *Active Learning* provides unlabeled training data and the learner decides whether to query or discard it. This guarantees that all inputs can be labeled. Pool-based *Active Learning* is similar to the stream-based scenario

except that the learner is provided with the full set of unlabeled situations before making any queries for labeled situations. There are several strategies that seek which situations to query. Uncertainty Sampling seeks less confident instances. Query-By-Commitee uses vote mechanisms to query a situation that generates the most disagreements. The queries can also be to seek which new labeled situation would most impact a model. Statistical techniques synthesize queries by minimizing the variance of the learner's errors.

Discussion

Part of our objectives focus on actively pointing out specific situations to enhance the learning process. *Active Learning* matches our needs concerning the self-generation of perceptions for *Endogenous Feedback*. As for *Supervised Learning*, *Semi-Supervised Learning* and *Active Learning* are not domain or task specific. Most work concerning this learning technique was focused on pool-based approaches which is by definition offline. However, in the last decade, online *Active Learning* techniques are appearing [Lughofer, 2017]. To my knowledge, *Lifelong Learning* and *Knowledge Generalization* are not dominant concerns in this domain of *Machine Learning*. *Self-Observation* is present in *Active Learning* as it allows to generate the queries. As for *Supervised Learning*, *Scalability*, data amount and *Explainability* are model dependent.

2.4 Learning by Rewards

In the family of learning by rewards, we found two principal approaches that are *Reinforcement Learning* and *Genetic Algorithms*. *Reinforcement Learning* mixes training and testing phases during which the learner actively interacts with its environment. Here, the teacher takes the form of a reward function that the learner questions to get positive or negative feedback for the actions it makes in specific states. To learn which actions are relevant, it seeks to maximize a reward function defined *a priori*. Classic methods are Q-learning [Watkins and Dayan, 1992], SARSA [Sutton, 1996] and case-based reasoning [Aamodt and Plaza, 1994]. One of the strengths of *Reinforcement Learning* is that it carries out a broader exploration of the research space and that this exploration is online [Sutton and Barto, 1998]. However, the counterpart is that this exploration can be costly in terms of execution time and it can damage the system if it is evolving in a dangerous environment [Mannucci et al., 2017]. Recent work using long term task descriptors enabled this method to generalize to unseen situations [Schaul et al., 2015]. As designing the reward function for a specific task can be immensely difficult, promising leads are to learn it from the environment.

This is what *Inverse Reinforcement Learning* does. It is an interesting approach because it permits the system to extract and learn a reward function from observed behavior of an expert [Abbeel and Ng, 2011].

Derived from *Reinforcement Learning*, *Genetic Algorithms* also address the problem of exploration. *Genetic Algorithms* belong to the family of evolutionary algorithms. These algorithms draw on the theory of evolution to solve various problems. They provide an ap-

proximate solution to an optimization problem when there is no exact solution or when the solution cannot be found in a reasonable time. *Genetic Algorithms* use the concept of natural selection and apply it to a population of potential solutions [Holland and Reitman, 1978]. In order to find better solutions than those existing in the population, the potential solutions are subjected to mutations or crosses. Solutions that bring improvements are mutated or crossed in order to find better solutions. An evaluation function or fitness function is used to measure the effectiveness of each solution and select the best performing solutions.

Discussion

Reinforcement Learning is not related to learning with *Endogenous Feedback* because it learns from its environment by interacting with it and there is no self-generation of endogenous situations nor *Self-Observation*. Recent work goes in this direction by choosing which reward must be taken [Krueger et al., 2020]. The dependence on a complex reward function makes it domain-specific. *Inverse Reinforcement Learning* allows to resolve this issue by inferring the reward function. *Reinforcement Learning* mainly introduces *Online Learning*. Lifelong and generalizing characteristics are in development thanks to recent research. [Silver et al., 2013] argues that *Reinforcement Learning* will be useful for *Lifelong Learning*.

Scalability is generally an issue, but recent papers managed to overcome this problem [Zhan et al., 2017]. More over, *Reinforcement Learning* usually relies on discrete states that represent a constrained environment different from a continuous one like the Sociotechnical Environment we described in section 1.4. Almost no work was found about learning with any data amount. [Duan et al., 2016] is one of the few works that achieves *Reinforcement Learning* with few training. *Explainability* in *Reinforcement Learning* learning is still an emerging research field where most of the approaches are task specific [Heuillet et al., 2020].

Genetic Algorithms or more generally evolutionary algorithms provide solutions to optimization and search problems by relying on bio-inspired operators [Mitchell, 1998]. They present interesting characteristics due to their *Online Learning*. We find here the same drawback that *Reinforcement Learning* showed which is having to rely on a fitness function that is task dependent and does not adapt with the evolving environment.

2.5 Learning by Demonstrations

Learning by Demonstrations is a *Supervised Learning* technique applied to many robotic applications. It is used to make robots learn behaviors thanks to two phases: firstly, the gathering of demonstration examples and secondly, the generation of a policy from the demonstration. The teacher here is an operator that usually shows how to perform a task. It is an interesting method because it provides a behavior to a system with an intuitive communication medium for human teachers, specially non-robotic experts [Argall et al., 2009, Verstaevel, 2016]. This mechanism is inherently linked to the information provided in the demonstration dataset. As a result, learner performance is heavily limited by the quality of this information [Mazac et al., 2015].

Discussion

Learning by Demonstrations is far from learning by *Endogenous Feedback* because it does not involve any self-generation of knowledge. [Silver et al., 2012] proposes a step towards this objective by making the learner request specific demonstrations. *Learning by Demonstrations* is a generic technique which brings *Agnosticity* because it is open to all the tasks that could be demonstrated but it is an offline approach because it works in two faces, learning and exploitation. It lacks *Lifelong Learning* properties because a learning system can learn from several demonstrations but there is no generalization across all the tasks it can learn. The learning is limited to its training and adaptation to unknown situations is not handled. [Mendez et al., 2018] was the first to propose a step towards *Knowledge Generalization* and *Lifelong Learning* using *Inverse Reinforcement Learning*.

Intention Learning [MacGlashan and Littman, 2015] is more interesting because it enables the system to seek and learn the motivations behind a demonstration which is a step towards *Endogenous Learning* (i.e. learning with *Endogenous Feedback*). In addition to the states of the world, *Intention Learning* uses the transitional dynamics of these environmental states to search for the goals that motivate the tutor's behavior. Thanks to the knowledge of these objectives, the robot behaves correctly when it encounters new situations and generalizes when simple demonstration learning could not. Planning algorithms are usually used to derive the behaviors which can lead to large computational cost for complex tasks and *Scalability* issues which is not the case in demonstration learning. The approach often used is the same as for an *Inverse Reinforcement Learning* problem in which the oracle's intention is modeled infering the reward function of the demonstrator [Ng and Russell, 2000]. *Intention Learning* is also one of the few fields where *Few Shot Learning* can be found, it is learning with few experiences. No work on *Explainability* according to *Learning by Demonstrations* nor *Intention Learning* was found in the literature.

2.6 Probabilistic Learning

The *Probabilistic Learning* framework is necessary in *Machine Learning* to represent and manipulate uncertainty about models and prediction. Even though there is controversy about how important it is to fully represent uncertainty, it yet plays a central role in scientific data analysis, *Machine Learning*, robotics, cognitive science, and *AI* [Ghahramani, 2015]. At the lowest level, uncertainty is noise on a measurement and at the highest level, it concerns the choice of the appropriate general structure of the model (linear regression, neural network...). Probabilistic approaches handle *Multi-Task Learning* [Lu and Tang, 2015], commonsense reasoning [Freer et al., 2012], model based *Machine Learning* [Bishop, 2013], Bayesian *Reinforcement Learning* [Vlassis et al., 2012] and investigate advanced memory management [Lucic et al., 2017]. There is also growing numbers of probabilistic programming languages. They are extensions of classical programming which permits to manipulate random variables and conventional deterministic variables [Roy, 2011].

Easily constructing hierarchical models is a powerful feature of the probabilistic framework to deal with large amounts of data and help to self-organize and facilitate *Self-*

Observation on data structures. Bayesian methods are at their most powerful with limited supply of data. There is yet a need to develop methods that scale up well to computational large data-sets. *Probabilistic Learning* is a significant part of *Machine Learning* theory. In my work this path was not explored any further, the focus being made on other learning approaches presented in the following chapters.

2.7 Synthesis

After assessing the fundamental approaches of *Machine Learning*, a synthesis is presented and a summary table is drawn up.

Among the classical approaches of learning, *Agnosticity* is what is most present. Indeed, the definition of learning is the acquisition of knowledge or skills through study, experience, or teaching. Nothing in its definition states that it should be specific to any domain of task.

The objectives that are most missing are learning with *Endogenous Feedback* and *Self-Observation*. Most methods lack introspection and self-reflection. Some learning fields are starting to focus on this aspect by seeking more information than the one that is provided by the training data, oracle, or demonstrator.

Lifelong Learning is a developing concern that is becoming present in many approaches. Closely related to *Online Learning* and *Knowledge Generalization*, it will be the major focus of the next chapter. Learning from *Any Data Amount* as it involves learning from few data such as from many data is poorly represented. Only few approaches address this problem.

As for *Scalability* and *Explainability*, both tend to be model dependent making then difficult to assess. Moreover, *Explainability* is late concern in *AI* which explains why the foundations of *Machine Learning* make very little reference to it.

Table 2.1 summaries the strengths and weaknesses of the presented learning techniques according to our objectives.

	Endogenous Feedback	*Agnosticity*	*Lifelong Learning*	*Online Learning*	*Self-Observation*	*Knowledge Generalization*	*Scalability*	*Any Data Amount*	*Explainability*
Supervised Learning	-	+	- +	- +	-	- +			
Unsupervised Learning	- +	+	- +	-	- +	- +	+		+
Semi-Supervised Learning	-	+	-	- +	- +	-			
Active Learning	- +	+	-	- +	+	-			
Reinforcement Learning	-	-	- +	+	-	- +	- +	- +	- +
Inverse Reinforcement Learning	- +	+	- +	+	-	- +	- +		- +
Learning by Demonstrations	-	+	- +	-	-	- +			
Intention Learning	- +	+	- +	-	-	- +	-	- +	

Table 2.1 – Synthesis on classical learning approaches with respect to the thesis objectives. Legend: + positive match ; - + in development or relative match; - negative match.

Next chapter will focus on *Lifelong Learning* and how it is related to other learning paradigms.

3 Towards Lifelong Learning

In this chapter, I continue the reflection on artificial learning by focusing on Lifelong Learning. I define what makes us human beings lifelong learners and what are the properties that are sought to be replicated in artificial learners. I present learning paradigms that are related to Lifelong Learning: Transfer Learning, Multi-Task Learning, Online Learning and Reinforcement Learning. I develop a more general questioning on the major protagonist of artificial learning today, Deep Learning. I finish updating the reflection about the presented approaches according to the objectives of my thesis.

LIFELONG Learning is one key to stronger *AI*. *Machine Learning* techniques often focus on how make the best model on a specific task instead of focusing on the process of learning itself [Hong et al., 2018]. Humans can learn a variety of tasks and build relations between them to learn new ones faster or consolidate the previously learned ones. Classical *Machine Learning* approaches don't make any connections between the tasks they learn, they learn in an isolated way (cf. chapter 2). They focuses on isolated single task learning. Knowledge is not accumulated and learning is performed without considering past learned experiences. They lack mechanisms for storing and reusing knowledge. Traditional *Machine Learning* allows to solve specific tasks, but if the perceptions change (Fig. 2.1), the models have to be rebuilt from scratch.

Lifelong Learning aims to overcome the isolated learning paradigm and uses knowledge acquired for one task to solve related ones. *Lifelong Learning* connects the learning of new tasks with the previous ones. This way, the learning process can be faster, more accurate and needs less training data. The learning power of humans is such that by using their basic knowledge and skills, they can exercise themselves to acquire new knowledge from previous experiences and self-motivated curiosity. We call this ability "Learning to learn" [Thrun and Pratt, 2012]. The mission of Lifelong *Machine Learning* is to replicate the possible connection between learning experiences through time in learning algorithms. *Lifelong Learning* is defined by three key characteristics: continuous learning process, explicit knowledge retention and accumulation, and use of previously learned knowledge [Chen and Liu, 2018]. In the literature, we find the following learning paradigms that are related to *Lifelong Learning*.

- ***Transfer Learning***. Also called domain adaptation, *Transfer Learning* is closely related to *Lifelong Learning*. It focuses on transferring knowledge from a source task/domain to a

target task/domain.

- ***Multi-Task Learning***. *Multi-Task Learning* seeks to simultaneously learn a group of related tasks and optimizes over all of them.
- ***Online Learning***. *Online Learning* concerns all algorithms where training data is provided progressively and not all at once. New data does not erase previous training data like in *Offline Learning*.
- ***Reinforcement Learning***. As seen before, *Reinforcement Learning* is a learning problem where an agent learns a behavior policy through trial and error interactions with a dynamic and labeled environment.

As *Reinforcement Learning* as already been presented in section 2.4, we will not discuss it in this chapter. *Reinforcement Learning* is strongly related to *Lifelong Learning* because it learns by trial and error in an online manner. *Reinforcement Learning* does not yet involve accumulation of knowledge across several tasks. A paradigm that is central in the learning community today is *Deep Learning*. Although it is not directly related to *Lifelong Learning*, many studies concerning Lifelong *Deep Learning* are emerging.

3.1 Transfer Learning

Transfer Learning is a research problem in *Machine Learning* that focuses on storing knowledge that is gained while solving one problem, and applying it to a different but related problem [West et al., 2007]. *Transfer Learning* is a *Machine Learning* method where a model developed for a task is reused as the starting point for a model on a second task. *Transfer Learning* and domain adaptation refer to the situation where what has been learned in one setting is exploited to improve generalization in another setting [Bengio et al., 2017]. It also focuses on learning properties and knowledge consolidation. Traditional *Machine Learning* allows to solve specific isolated tasks whereas *Transfer Learning* works on associating the learning of new experiences with the previous ones. *Transfer Learning* is a method enabling to build models that do not need to be retrained every time the data changes a little. Every time knowledge is missing or more relevant knowledge can be learned, the prediction function is refined by knowledge transfer, specially when data is of a different nature (Fig. 3.1).

Formalism

[Pan and Yang, 2010] proposed a formalization of *Transfer Learning* by defining domain and task. A domain $\mathcal{D}$ consists of two components : $\mathcal{D} = \{\mathcal{X}, P(X)\}$ with $\mathcal{X}$ the feature space and $P(X), X = \{x_1, \ldots, x_n\}, x_i \in \mathcal{X}$ a marginal probability distribution representing the features. A task, $\mathcal{T}$, can be defined as a two-element tuple: a label space $\mathcal{Y}$, and a predictive function, η (equation 3.1). The predictive function can also be denoted as $P(\mathcal{Y}|X)$ from a probabilistic view point. The predictive function η is learned from feature/label pairs, $(x_i, y_i), x_i \in \mathcal{X}, y_i \in \mathcal{Y}$. For each feature vector in the domain, η predicts its corresponding label: $\eta(x_i) = y_i$

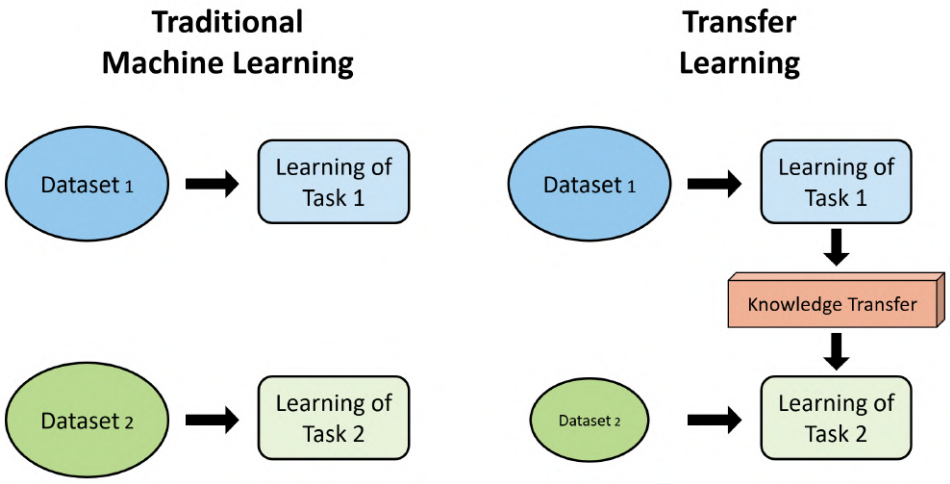


Figure 3.1 – Traditional *Machine Learning* VS *Transfer Learning*

$$T = \{\mathcal{Y}, P(Y|X)\} = \{\mathcal{Y}, \eta\} \qquad Y = \{y_1, \ldots, y_n\}, y_i \in \mathcal{Y} \tag{3.1}$$

Given a source domain $\mathcal{D}_S$, a corresponding source task $\mathcal{T}_S$, as well as a target domain $\mathcal{D}_T$ and a target task $\mathcal{T}_T$, the objective of *Transfer Learning* is to learn the target conditional probability distribution $P(\mathcal{Y}_T|\mathcal{X}_T)$ in $\mathcal{D}_T$ with the information gained from $\mathcal{D}_S$ and $\mathcal{T}_S$ where $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$ (Fig. 3.2). In most cases, a limited number of labeled target examples are provided, which is exponentially smaller than the number of labeled source examples that are assumed available.

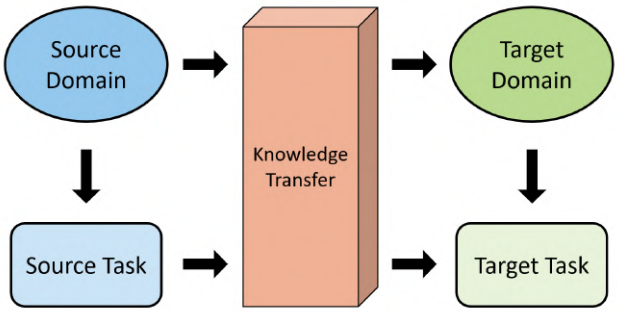


Figure 3.2 – *Transfer Learning* Formalism

Strategies

Depending on the availability of data and on the changing domain and task, *Transfer Learning* is decomposed into several strategies (Fig. 3.3).

- **Inductive *Transfer Learning*** The source and target domains are the same, the source and target tasks are related but different from each other. The algorithms utilizes the inductive biases of the source domain to help improve the target task. Depending upon whether the source domain contains labeled data or not, Inductive *Transfer Learning* is

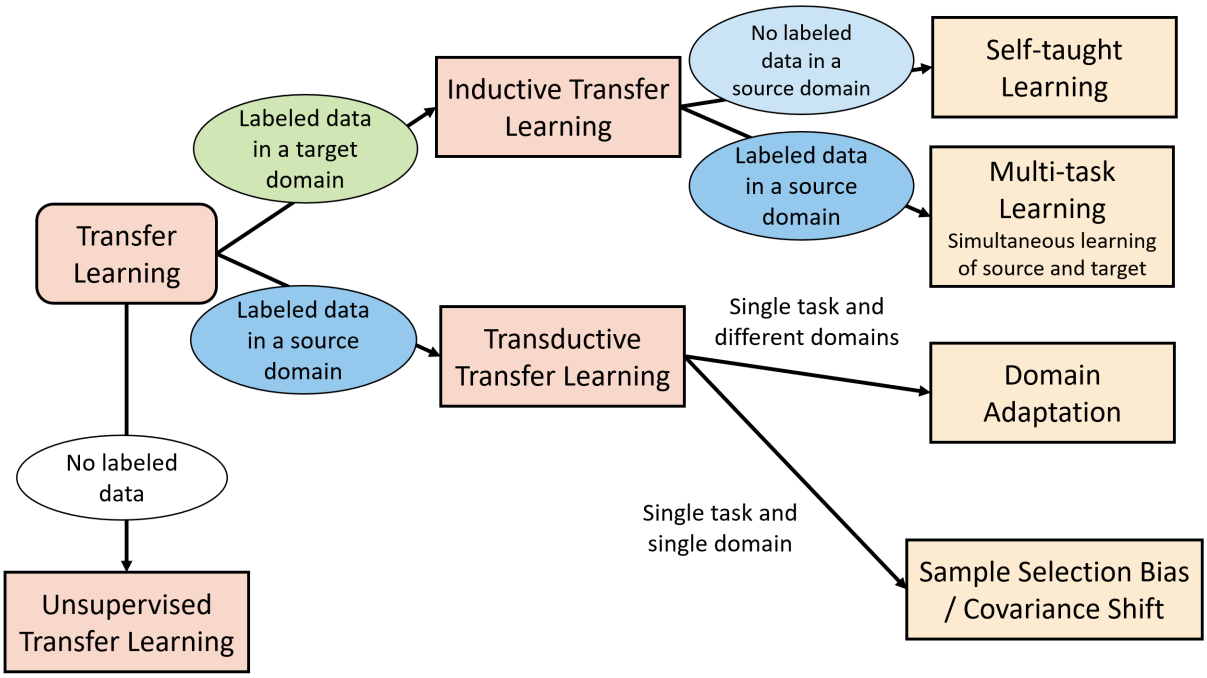


Figure 3.3 – *Transfer Learning* Strategies

divided into two subcategories, *Multi-Task Learning* and self-taught learning. Tasks are usually regression or classification.

- **Unsupervised *Transfer Learning*** The source and target domains are similar, but the tasks are different. In this scenario, labeled data is unavailable in either domains. Tasks are clustering or dimensionality reduction problems.
- **Transductive *Transfer Learning*** There are similarities between the source and target tasks, but the corresponding domains are different. In this setting, the source domain has a lot of labeled data, while the target domain has none. If domains are different, it is domain adaptation. In the other case, it is sample selection bias or covariance shift. Tasks are also regressions and classification problems.

Discussion

Transfer Learning does not match all the requirements of *Lifelong Learning* for several reasons [Chen and Liu, 2018]. Learning is not continuous and it is not concerned about knowledge accumulation. Learning is unidirectional and it usually involves only two domains (the source domain and the target domain).

If we now focus on the other objectives of section 1.6, the main strength of *Transfer Learning* is *Knowledge Generalization* from a task to another one. It is an agnostic approach as we see the many applications it is involved with [Weiss et al., 2016]. Most *Transfer Learning* works are offline but online *Transfer Learning* is attracting attention in the *Machine Learning* community [Wu et al., 2017]. *Transfer Learning* suffers from *Scalability* issues when the number of tasks to learn is large [Zhan et al., 2017]. Only a few works deal with adaptation in *Transfer Learning*. [Cao et al., 2010] focused on adapting the *Transfer Learning* schemes by automati-

cally estimating the similarity between a source and a target task. However, our main goal is to self-enhance learning through *Endogenous Feedback* in an adaptive manner using *Self-Observation* properties which *Transfer Learning* lacks. Recent work is focusing on *Few Shot Learning* and *Meta-Transfer Learning* [Sun et al., 2019]. *Meta-Learning* is a close paradigm to Learning by *Endogenous Feedback* because it is focused on speeding up and improving *Machine Learning* design at a higher level. We will discuss this topic with more detail in the next chapter. To my knowledge, in the literature about these learning approaches, there is no work commenting on the *Explainability* of *Transfer Learning* techniques.

3.2 Multi-Task Learning

As we have seen in the previous section, *Multi-Task Learning* can be considered as a *Transfer Learning* scenario. The approach focuses on learning simultaneously related tasks and using their relations to achieve better performances on each task [Li et al., 2009]. Common techniques are feature learning approach, low-rank approach, task clustering approach, task relation learning approach, and decomposition approach [Zhang and Yang, 2017].

Deep Learning also tackled the problem of *Multi-Task Learning*. [Liu et al., 2015] presented a multi-task DNN for learning representations across multiple tasks that where query classification and web search ranking. [Huang et al., 2013a] applied Deep Neural Networks to multilingual data. [Zhang et al., 2014] used deep *Multi-Task Learning* for the problem of facial landmark detection. Other applications like name error detection in speech recognition [Cheng et al., 2015], mutli-label learning [Huang et al., 2013b] and phoneme recognition used deep *Multi-Task Learning* [Seltzer and Droppo, 2013].

Discussion

As for *Transfer Learning*, *Multi-Task Learning* does not accumulate any knowledge over time and it does not use the concept of continuous learning. Optimizing all tasks simultaneously when a new task is added online is difficult in a single process as they can be too numerous and diverse. As we have seen previously, a challenge of *Lifelong Learning* is to combine the objectives of *Transfer Learning* and *Multi-Task Learning* in a continual manner. [Ruvolo and Eaton, 2013] proposed the Efficient *Lifelong Learning* Algorithm (ELLA) an online method using both *Transfer Learning* and *Multi-Task Learning* tending towards continuous learning. Still, some local and distributed optimizations are needed [Chen and Liu, 2018]. We have seen that *Multi-Task Learning* can be applied to numerous applications making it an agnostic approach. [Zhang and Yang, 2017] argues that *Multi-Task Learning* lacks flexibility and robustness to outlier tasks which is due to few *Self-Observation*. Moreover, most studies focus on *Supervised Learning* and not on *Unsupervised Learning* nor on *Active Learning* of which *Endogenous Learning*, as we define it, is closer. *Multi-Task Learning* with few experiences is still little explored in literature. [Zhan et al., 2017] states that *Multi-Task Learning* faces *Scalability* issues. To overcome this problem, [Zhang et al., 2018] proposed a distributed online *Multi-Task Learning* approach to distribute computation with large amounts of tasks. *Explainability* is an incipient field in *Multi-Task Learning*,

[Chen et al., 2019, Wang et al., 2018] recently proposed explainable recommendation systems using *Multi-Task Learning*.

3.3 Online Learning

Online Learning is a learning paradigm where training data is sequentially provided to the learning mechanism. The new data updates the models built so far instead of re-training the models on all the available data like it would be done in *Offline Learning* techniques. *Offline Learning* (also called *Batch Learning*) suffers from re-training cost when they have to deal with new training data. *Online Learning* is a promising approach for real world large-scale applications as data comes from a continuous stream. It is already involved in a wide variety of tasks such as classification for spam email filtering, regression analysis, recommendation systems or clustering. *Online Learning* can be divided into three categories: online *Supervised Learning*, online *Supervised Learning* with limited feedback and online *Unsupervised Learning* [Hoi et al., 2018].

Online *Supervised Learning* provides full feedback information to the learner [Shalev-Shwartz et al., 2011]. For online *Supervised Learning* with limited feedback, the learner has to make a compromise between exploiting known knowledge and exploring unknown information. It is an important branch of *Online Learning* called *Bandit Online Learning* [Gittins et al., 2011]. We also find online *Active Learning* where the learner receives unlabeled training data and picks which data should be labeled. Common settings are selective sampling [Orabona and Cesa-Bianchi, 2011] and *Active Learning* with expert advice [Hao et al., 2018]. Finally, online *Semi-Supervised Learning* also belongs to online *Supervised Learning* with limited feedback. Online *Semi-Supervised Learning* uses sequential labeled and unlabeled training data [Goldberg et al., 2011]. Online *Unsupervised Learning* is an extension of *Unsupervised Learning*. Typical problems are also clustering [Aggarwal, 2013], dimension reduction, anomaly detection [Kloft and Laskov, 2012] and density estimation [Qahtan et al., 2016] but with a continuous stream of training data.

Other learning paradigms are also closely related to *Online Learning*. *Online Learning* can also be called *Incremental Learning*. *Incremental Learning* generally uses sequential training data like *Online Learning* but with limited space and computational costs [Wu et al., 2019]. *Sequential Learning* also focuses on learning sequences taking into account possible past and future sequences [Aljundi et al., 2018]. Stochastic learning is related to *Online Learning* because it is motivated by accelerating learning speed to achieve large scale learning tasks [Bottou, 2010]. Another close paradigm is *Adaptive Learning*. [Verstaevel et al., 2017] proposed a self-adaptive Multi-Agent approach for real world lifelong *Machine Learning*. Interactive *Machine Learning* generally interacts with human users to iteratively refine the models of the learning mechanism [Dudley and Kristensson, 2018]. As we already discussed *Reinforcement Learning* is by definition composed of sequential trials of online exploration. Finally, *Continual Learning* is commonly use to refer to *Lifelong Learning* as the ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences [Parisi et al., 2019].

Discussion

From a *Lifelong Learning* point of view, the strength of *Online Learning* is that it can adapt to sequential training data provided during run-time. It does not focus on different tasks and on the knowledge relatedness between them to accomplish *Knowledge Generalization* [Chen and Liu, 2018]. Online *Supervised Learning* with limited feedback is close to Learning by *Endogenous Feedback* because it uses labeled and unlabeled training data but this data is not self-generated by the learner. *Online Learning* is not specialized in a task in particular which makes it an agnostic approach. *Self-Observation* is emerging in this domain thanks to Multi-Agent Systems which focus on complex real world problems. [Hoi et al., 2018] argues that *Online Learning* approaches are still far from tackling real world applications because of their lack of *Scalability* and their lack of robustness to noise. However, as the training data is processed incrementally, *Online Learning* is a powerful tool to deal with large data amounts as well as small data amounts. To my knowledge there is not any work focusing on the *Explainability* of *Online Learning* approaches.

3.4 Deep Learning

During the last decade, Artificial Neural Networks (ANN) and Deep Neural Networks (DNN) have proved themselves on many applications: object recognition [LeCun et al., 2015], speech recognition [Weng et al., 2014], video games [Mnih et al., 2015], board games [Silver et al., 2017]. They have achieved performances that beat humans in some respects [Gibney, 2016].

Despite all these successes, a part of the learning community is concerned about the future of *Deep Learning*. For several reasons, [Marcus, 2018] argues that *Deep Learning* cannot be considered as a general solution to *Artificial General Intelligence (AGI)*. *AGI* is the future of Narrow *AI*, its opposite. Narrow *AI*, as defined by [Kurzweil, 2005], refers to specific intelligent behaviors in specific contexts. Ideal *AGI* is a numerical version of the human brain with infinite generality, adaptability and flexibility [Goertzel, 2014]. So far, *Deep Learning* is not a current promising lead for the following reasons.

- **Data hungry**. *Deep Learning* lacks a mechanism for learning abstractions. It relies on large numbers of labeled examples [Sabour et al., 2017]. *Deep Learning* is often not an ideal solution when training data is limited.
- **Shallow and limited capacity for transfer**. *Deep Learning* patterns extracted are usually more superficial than they initially appear [Kansky et al., 2017, Jia and Liang, 2017].
- **No natural way to deal with hierarchical structure**. Recurrent Neural Networks are limited in their capacity to represent and generalize rich structure in a faithful manner [Lake and Baroni, 2018]. Issues emerge when complex hierarchical structures are needed, in particular when a system is likely to encounter novel situations. Superficially, Neural Networks do not operate on symbols but on inputs that represent (micro) features [Lyre, 2020].
- **Transparency and Explainability**. The *Explainability* of Neural Networks is a major focus of discussion [Samek et al., 2017, Ribeiro et al., 2016]. This is known as the

problem of opacity or the black box problem of Deep Neural Networks [Lyre, 2020, Zednik, 2019]. Understanding the internal structure of an *AI* system is indispensable to assign cognitive or intelligent properties to it.

- **Prior knowledge**. Prior knowledge is often deliberately minimized. Culture in *Machine Learning* emphasizes competition on problems that are inherently self-contained. Commonsense reasoning essentially lies outside the scope of what *Deep Learning* is appropriate for [Davis and Marcus, 2015].
- **Semantic Grounding**. *Deep Learning* learns complex correlations between input and output features, but with no inherent representation of meaning nor causality. [Lyre, 2020] states that semantic grounding (divided into functional, causal and social grounding) is one of key directions for *AGI* and it is not fully present in the late success of *Deep Learning* such as DeepBlue, AlphaGoZero or AlphaZero.
- **Presumes a largely stable world**. The logic of *Deep Learning* is such that it is likely to work best in highly stable worlds, which have unvarying rules, and less well in systems such as politics and economics that are constantly changing [Lazer et al., 2014]. Deep *Reinforcement Learning* has proven some generalization properties but the specification of a special target function, with respect to which a system then allows generalization, is still a clear limitation [Lyre, 2020].
- **Cannot be fully trusted**. *Deep Learning* systems are quite good at some large fraction of a given domain, yet they can be easily fooled [Nguyen et al., 2015].
- **Difficult to engineer with**. It's easy to make systems that work in some limited set of circumstances, but it is quite difficult to guarantee that they will work in alternative circumstances with novel data that may not resemble previous training data [Sculley et al., 2014]. *Deep Learning* lacks the incrementality, transparency and debuggability of classical programming. Moreover, replicability of state-of-the-art methods can be a tough issue [Henderson et al., 2018].

Discussion

Artificial Neural Networks excel at solving closed-end classification problems. Still, *Deep Learning* systems face issues with limited amounts of available training data. Usual neural networks approaches need large amounts of data to achieve the high performances they are famous for, on task dependent problems and offline trainings. Deep Neural Networks tend to overfit when only few learning samples are used [Sun et al., 2019].

Simple multilayer perceptrons cannot generalize outside their training space [Chollet et al., 2018]. Contemporary Neural Networks do well on challenges that remain close to their core training data. But they start to break down on cases further out in the periphery. They do not flexibly generalize to new tasks [Ciregan et al., 2012]. Yet, a recent study supports promising leads to improve *Deep Learning* towards human-like structured cognition [Lake et al., 2017]. One of these is Differentiable Neural Computer (DNC), inspired from Neural Turing Machines (NTM) [Graves et al., 2014]. They are augmented Neural Networks with an external memory which enables them to represent and manipulate complex data structures and to learn to do so from data [Graves et al., 2016]. Thanks to external memory,

[Vinyals et al., 2016] managed state-of-art performance on one-shot classification tasks but with task-dependent and *Scalability* drawbacks.

Lifelong Learning is a long-standing challenge in Neural Network models since the continual acquisition of data generally leads to catastrophic forgetting or interference. This limitation is a major drawback for state-of-the-art Deep Neural Networks that typically learn representations from stationary training data [Parisi et al., 2019]. Gradient Episodic Memory (GEM) [Lopez-Paz et al., 2017] is a Neural Network approach focused on transferring knowledge and not forgetting during a continuous learning thanks to episodic memory with few training examples. GEM demonstrated competitive performance against the state-of-the-art [Kirkpatrick et al., 2017, Rebuffi et al., 2017].

Online Learning is a known problem for Deep Neural Networks as they are trained in a batch learning setting. [Sahoo et al., 2017] recently identified the issues and proposed an online setting for *Deep Learning* using shallow and deep networks.

Only few recent work managed to use limited amounts of training data. Late discoveries, thanks to additional memories, managed to perform one-shot learning, *Knowledge Generalization*, *Self-Observation* on data structures and worked on lifelong mechanisms, but still with task dependency and *Scalability* drawbacks. Considered like black boxes by a growing number of researchers, *Deep Learning* faces *Explainability* and design difficulties.

At this point, *Deep Learning* did not concentrate on learning by *Endogenous Feedback* nor self-generation of learning situations.

3.5 Synthesis

As we have seen, *Transfer Learning* and *Multi-Task Learning* are the most related paradigms to *Lifelong Learning* because they lead to *Knowledge Generalization*. However, they don't enable to learn continuously and to retain or accumulate learned knowledge explicitly. *Online Learning* and *Reinforcement Learning* allow continuous learning but they focus on a single task with a time dimension. *Lifelong Learning* is an emerging concern and the work in this field is under development. *Online Learning* has interesting characteristic concerning *Agnosticity*, *Scalability* and learning from *Any Data Amount*. *Self-Observation* is a recurrent lack such as the generation of *Endogenous Feedback* as the two are linked. *Explainability* is still an incipient worry in the presented approaches that possess no innate characteristics according to this objective.

According to [Chen and Liu, 2018], the following challenges need to be overcome in order to achieve *Lifelong Learning*.

- **Correctness of knowledge**. Knowing if a piece of past knowledge is crucial. This meets the need for *Self-Observation* to self-assess any learn knowledge.
- **Applicability of knowledge**. Estimating if a piece of knowledge is applicable or not in the context of a new learning task is essential.
- **Knowledge representation and reasoning**. There is a need to return to the early days of *AI* and reintroduce reasoning in learning mechanisms.
- **Heterogeneity of tasks**. Learned tasks must be of multiple types and/or from different domains to achieve full generalization.
- **Self-motivated learning**. If robots will interact with their environment and learn continuously, they need to collect their own training data. The exploration process must be guided by a sense of internal curiosity and interest for the unknown.
- **Compositional learning**. Any level of granularity must be used to share knowledge in the learning system.
- **Social learning**. The learning must be able to speed up its learning using interactions with other systems or humans.

After this chapter, it is reasonable to conclude that current learning techniques need more introspection and self-motivated mechanisms in their process of learning. One lead is to explore *Unsupervised Learning* but more in a sense of *Self-Learning*. Futures approaches need to focus on autonomous goal setting. If we could design systems that set their own goals with higher abstract level, major progress might follow. Another lead is also Symbolic *AI* or symbol manipulation. Hybrid approaches allow to bring more realistic classification and additional contextual information [Rueda et al., 2019]. A multidisciplinary track is cognitive and developmental psychology. Understanding the innate machinery of human mind will be valuable for designing the future *AI*. Infants are born with core knowledge to understand space, time and objects [Spelke and Kinzler, 2007]. This knowledge is the key to unlock the first stages of learning in each newborn child. This will be the major topic of the next chapter.

Table 3.1 summaries the strengths and weaknesses of the presented learning techniques according to our objectives.

	Endogenous Feedback	*Agnosticity*	*Lifelong Learning*	*Online Learning*	*Self-Observation*	*Knowledge Generalization*	*Scalability*	*Any Data Amount*	*Explainability*
Transfer Learning	- +	+	- +	- +	- +	+	-	- +	
Multi-Task Learning	-	+	- +	- +	-	+	- +		- +
Online Learning	- +	+	- +	+	- +	-	+	+	
Deep Learning	-	- +	- +	- +	- +	- +	- +	- +	-

Table 3.1 – Synthesis on *Lifelong Learning* approaches with respect to the thesis objectives. Legend: + positive match ; - + in development or relative match; - negative match.

Next chapter will finish this state of the art by focusing on learning techniques that implement introspection in the learning process.

4 Endogenous Learning

In this chapter, I extend the reflection on artificial learning by focusing on a more psychological view of learning. I present learning from the Constructivism point of view which gave birth to Developmental Learning. Then, I explore other learning paradigms that are related to Developmental Learning: Meta-Learning and Few Shot Learning. I introduce the Context Learning paradigm upon which the work of this thesis is based. The use of Adaptive Multi-Agent Systems is defended by presenting their properties which are in adequacy with the motivations of my thesis.

HOW do humans learn ? Infants learn the understanding of their environment and of their embodiment from their first experiences. This learning can be stimulated but it is undeniable that the human mind goes through first stages of learning before forging their adult mindset. These first stages are accomplished thanks to prior learning capabilities that each human possesses. Such capabilities are essential if one aspires to design an *AI* that is capable of learning from its environment and its interactions like humans do. Thus, one must try to design an infant like *AI* before even considering designing an adult-like *AI*.

> *Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain.* —Alan Turing, "Computing Machinery and Intelligence" [Turing, 1950]

From this statement, several works have been conducted on cognitive psychology to understand how do infants learn and how could these mechanisms could be replicated in *AI*. We will first present *Developmental Learning* that was born from *Constructivist Learning* inspirations. *Constructivist Learning* gives a theory on how knowledge is constructed and evolves. *Developmental Learning* is strongly related to *Developmental Robotics*. Robotics are partly the origin of the creation of *Developmental Learning* by expressing the need of cognitive development within robots. We will then introduce *Meta-Learning* and *Few Shot Learning*, learning paradigms that are related to learning to learn. We will finish by bringing to light *Context Learning*, a distributed learning approach also inspired from *Constructivist Learning*.

4.1 Developmental Learning

As we have seen, learning systems usually focus on a single-task or very limited number of tasks. Designing systems capable of learning throughout their lives, whatever the task they have to perform, is one of the directions *Developmental Learning*. This approach aims to reproduce the cognitive stages in infants in order to learn incrementally as humans do, and understand what motivates this learning. Directly inspired from *Constructivist Learning*, the approach of *Developmental Robotics* focuses on developing robots with infants mental abilities, assuming that we should first design an infant-like *AI* before trying an adult one [Weng et al., 2001, Zlatev and Balkenius, 2001, Cangelosi and Schlesinger, 2015].

Constructivism

The constructivist theory comes from Piaget's work on infant development [Piaget, 1977, Piaget, 1978]. According to this theory, knowledge is a construction, based on the observation of the subject's environment and the impact of its actions. The previously acquired concepts serve as a basis for assimilating and interpreting new experiences, and these old concepts themselves are restructured in the light of these observations. The basic unit of knowledge in this theory is the *schema*. This aggregates several perceptions and, in most cases, several actions [Guerin, 2011]. Firstly, proposed by Drescher [Drescher, 1991] and then formalized by Holmes [Holmes et al., 2005], it has been reused combining it to Self-Organizing Maps (SOM) [Chaput, 2004, Provost et al., 2006], model-based learning [Perotto, 2013] and Muti-Agent Systems (MAS) [Mazac et al., 2015, Guériau, 2016].

Principles

Developmental Robotics focus on developing robots autonomously. The robot acquires data as a child would. Indeed, we are already learning to develop our senses from the fetal phase [Asada et al., 2009]. This approach is intended for "virgin" systems that learn from scratch and thus create multiple levels of learning.

Developmental Learning has some basic principles: the principle of verification, incarnation, subjectivity, grounding and incremental development. The principle of verification is the fact that an intelligent agent can create or maintain knowledge, only to the extent that he can verify this knowledge himself. This saves programmers from coding all the possibilities and leaving the system to manage its knowledge on its own. The principle of incarnation says that physical structure is needed for an *AI* and not just code. This is in line with the thinking of [Pfeifer and Bongard, 2006] and [Beer, 2014] that states that embodiment is essential for cognition. The principle of subjectivity says that learning is experimental. Indeed, for the same task, 2 different robots can perform it in different ways due to their experiences. The purpose of the grounding principle is to determine what is a successful verification. Finally, the principle of incremental development is the fact that we learn something step by step. For example, we learn to walk before we can run [Stoytchev, 2009].

Learning Paradigms

Several *Machine Learning* paradigms intervene in this domain. Among them we find *Curriculum Learning*, *Transfer Learning*, *Multisensory Learning* and *Intrinsic Motivation*. *Curriculum Learning* is strongly related to *Lifelong Learning* but it focuses on progressively learning harder tasks. As the human mind is considered highly plastic in its early life, easy tasks should be learned with high plasticity. Then, this plasticity decreases as the learner ages and faces more complex tasks. As we have already discuss, *Transfer Learning* is an open challenge in *Machine Learning* and also a big concern of *Developmental Learning* where *Knowledge Generalization* is essential. *Multisensory Learning* is a key feature to design a coherent, robust, and efficient interaction with any noisy environment [Parisi et al., 2019].

Intrinsic Motivation is a major topic of *Developmental Learning* or *Developmental Robotics*. Beyond *Inverse Reinforcement Learning* and *Intention Learning*, in the fields of robotics, *Intrinsic Motivation* is a way to internally guide exploration by focusing on curiosity mechanisms to improve exploration in the learning process [Forestier and Oudeyer, 2016, Bondu and Lemaire, 2007]. *Intrinsic Motivation* promotes hierarchical learning of knowledge and the reuse of skills [Baldassarre, 2011]. We found two broad theoretical views [Mirolli and Baldassarre, 2013], the knowledge-based *Intrinsic Motivation* view and the competence-based *Intrinsic Motivation* view. Knowledge-based *Intrinsic Motivation* focuses on enabling the organism to detect novel or unexpected features while competence-based *Intrinsic Motivation* focuses on the particular abilities or skills the robot possesses.

Achievements

There are several robots designed for *Developmental Robotics*, such as example CB^2 (child robot with bio-mimetic body) [Minato et al., 2007] or ICub [Metta et al., 2010]. In the literature, there are mostly architectures that use different learning mechanisms. The Multilevel Darwinist Brain (MDB) focuses on online *Lifelong Learning* [Bellas et al., 2010]. The emergent cognitive architecture iCub [Vernon et al., 2011] includes the principles of self-organization, *Intrinsic Motivation*, social learning, embodied and active cognition, online and cumulative learning. In particular, this architecture develops short term and long-term memory mechanisms to deal with *Lifelong Learning*. One interesting approach is the Multilevel Darwinist Brain (MDB) [Bellas et al., 2010]. It is a cognitive architecture that follows an evolutionary approach to provide autonomous robots with lifelong adaptation and *Online Learning*. High computational cost induced by the evolutionary algorithms makes it unscalable. It enables minimal intervention of the designer and complex behaviors in dynamic environments.

Discussion

Developmental learning gives promising results because it focuses on several of the features we are interested such as *Agnosticity*, *Lifelong Learning*, *Online Learning*, and *Knowledge Generalization*. *Intrinsic Motivation* brings *Self-Observation* and *Endogenous Feedback* as it seeks for internally enhancing exploration to enhance the learning process. However, most existing *Developmental Robotics* models only focus on some of these principles

[Cangelosi and Schlesinger, 2015] and some show *Scalability* issues. *Developmental Robotics* suffer from the same *Explainability* issues than *Deep Learning* as neural networks constitute a major part of modern robotics. Only few methods so far enable to conceptualize neural networks [Weber and Wermter, 2020]. At this moment, there isn't any work about learning from *Any Data Amount* in the *Developmental Learning* field.

4.2 Meta-Learning

In the field of *AI*, learning aims to train a model based on parameters and data to solve a problem. Learning usually requires prior knowledge of the nature of the problem to set these parameters at the time of the design. It is the responsibility of the designer to determine the right knowledge to get the best possible model. This human intervention prevents from providing easy reuse of a model for a different problem or for a problem that is likely to change. One solution could be to permit models to "learn to learn", i.e. to automatically adapt its learning in order to be more efficient. This area is called *Meta-Learning*. Challenges in *Meta-Learning* are to collect meta-data about prior experiences, models and parameters, and to extract and transfer knowledge from it to guide the new learning tasks. According to [Lemke et al., 2015], a *Meta-Learning* system must include a learning subsystem that adapts through experience. This experience is represented by meta-knowledge extracted from previous learning and/or from different domains or problems.

Meta-Learning techniques can be categorized according to the type of meta-data they exploit, from the most general to the most task-specific [Vanschoren, 2019]. Techniques can be centered around model evaluations to recommend configurations. Other methods focus on explicit task similarity by learning meta-models representing the relationships. And some approaches concern the transfer of trained model parameters between similar tasks allowing *Transfer Learning* and *Few Shot Learning*. *Multi-Task Learning* and *Ensemble Learning* (building several models on the same tasks) are often combined with *Meta-Learning*.

Discussion

Usual *Meta-Learning* techniques work in an offline setting. [Finn et al., 2019] recently proposed an Online *Meta-Learning* setting which merges the ideas from both paradigms to better capture the spirit and practice of continual *Lifelong Learning*. This work appeared to be scalable as it outperformed traditional *Online Learning* approaches on large-scale problems. [Obamuyide and Vlachos, 2019] also introduced continuous setting in a *Meta-Learning* context. [Finn et al., 2017] proved that *Meta-Learning* can be model-agnostic by designing an algorithm that is applicable to a variety of different learning problems. [Gui et al., 2018] implemented proactive and adaptive *Meta-Learning* that frames *Few Shot Learning* by capturing common knowledge among a set of few-shot learning tasks.

According to *Endogenous Feedback* and *Self-Observation*, we sense the same motivations as ours to enhance the learning process by extracting additional information from the learning experiences. However, this approach lacks internal motivation and self-awareness. Even if the work concerning *Explainability* is still in its early stages, [Daglarli, 2021] argues that *Meta-*

Learning gives promising leads towards explainable *Deep Learning*.

4.3 Few Shot learning

Machine Learning techniques usually demand huge quantities of training data. However, humans learn from few examples and this is what *Few Shot Learning* (also called *One-Shot* or *Zero-Shot Learning*) is trying to reproduce. *Few Shot Learning* or *One-Shot Learning* are variants of *Transfer Learning*, where the learner tries to infer a required output based on just one or a few training examples using prior knowledge from a source task or domain [Wang et al., 2020]. *Zero-Shot Learning* is another extreme variant of *Transfer Learning*, which relies on no labeled examples to recognize a new perception. *Zero-Shot Learning* methods make clever adjustments during the training stage itself to exploit additional information to understand unseen data [Xian et al., 2017]. *Few Shot Learning* is strongly related to *Knowledge Generalization* as in *Few Shot Learning*, the problem is showing to the learner what matters for generalization to rapidly learn a new perception [Xian et al., 2017].

Few Shot Learning methods can be categorized into two classes: data augmentation and task-based *Meta-Learning* [Sun et al., 2019]. Data augmentation focuses on increasing the amount of available data by generating it [Khoreva et al., 2017]. Task-based *Meta-Learning* extracts experience from multiple experiences or models to perform faster and better on new situations. The use of prior knowledge is central in *Few Shot Learning*. A recent survey proposed another categorization based on data, model and algorithm depending on how is generated the prior knowledge [Wang et al., 2020]. The categorization concerning the data is the same as [Sun et al., 2019] but data is not only generated but also transformed.

Model-centered approaches use prior knowledge to constrain the complexity of the hypothesis space. Among these approaches, one finds *Multi-Task Learning*, *Embedding Learning*, *Learning with External Memory* and *Generative Modeling*. *Multi-Task Learning* allows *Few Shot Learning* by sharing or linking parameters between tasks. *Embedding Learning* embeds similar training data into lower dimensional spaces to better identify similarities and reduce the hypothesis space. Task-specific information can be used in addition to prior knowledge. *Learning with External Memory* extract refined knowledge in an external memory. This memory is then used to represent unseen situations in a reduce space. *Generative Modeling* estimates probability distribution from prior knowledge to constrain the hypothesis.

Algorithm centered approaches allow to parameterize the hypothesis for better initialization. Usual *Supervised Learning* approaches handle enough training samples and the use of cross validation allows to find the appropriate parameters. In *Few Shot Learning*, it is not the case, other mechanisms allow to set these parameters. In this case, the parameters are existing parameters obtained from prior knowledge that are refined or learned using the few training situations.

Discussion

Data augmentation usually suffers from domain-dependent issues. Model centered approaches cannot always be applied in different learning contexts and algorithm centered

approaches are strongly dependent from the prior knowledge to refine the parameters [Wang et al., 2020]. The needed prior knowledge is usually task-specific which can remove *Agnosticity*. One limitation of *Few Shot Learning* so far is that studies about this topic were mostly concentrated on image classification or object recognition [Fei-Fei et al., 2006, Lake et al., 2011, Wertheimer and Hariharan, 2019]. This gives poor proof of the *Agnosticity* of the implemented techniques. Yet, recent works achieved promising results using a *Meta-Learning* approach in a *Few Shot Learning* setting and applied it to sinusoidal regression, image classification, *Active Learning*, and *Reinforcement Learning* [Yoon et al., 2018]. [Duan et al., 2017] introduced *One-Shot Imitation Learning*, a combination of *Meta-Learning* and *Reinforcement Learning* to learn from few demonstrations in a robotic case scenario.

Online Learning seems to be poorly explored in the *Few Shot Learning* works. [Stewart et al., 2020] recently proposed online few-shot gesture learning using Spiking Neural Networks (SNN). SNNs incorporate the concept of time and use impulses to transmit information within the network. *Few-Shot Lifelong Learning* is also in its early work, [Wei et al., 2020] lately tackled it by outperforming classical *Few Shot Learning* on several benchmarks. Across the literature of *Few Shot Learning*, *Self-Observation*, as we intend it, is not a common concern. The generation of additional training data fits our view of *Endogenous Feedback* but not all approaches implement it. *Knowledge Generalization* is predominant in this field as it is essential to learn with few experiences thanks to prior knowledge and generalization. *Scalability* is also an issue in *Few Shot Learning* that is being explored in recent work [Li et al., 2019]. So far, there isn't any technique specialized in *Few Shot Learning Explainability* but [Sun et al., 2020] proved that by applying approaches intended for *Deep Learning*, few-shot classification models have been improved while providing intuitive and informative visualizations.

4.4 Context Learning

Context Learning comes from the *Multi-Agent System* paradigm combined with *Constructivist Learning*. This section introduces *Adaptive Multi-Agent Systems*: *Multi-Agent Systems* with enhanced cooperation properties. *Context Learning* formalism and functioning is then detailed.

Adaptive Multi-Agent Systems

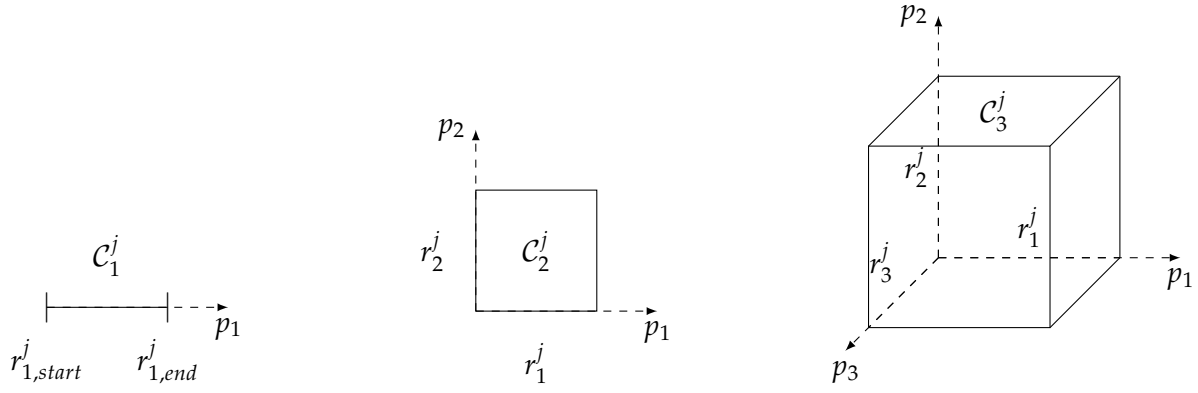
The *Multi-Agent System (MAS)* approach [Ferber, 1999], and in particular the *Adaptive Multi-Agent System (AMAS)* approach [Georgé et al., 2011], gives a system adaptive properties to deal with unexpected situations, which is appropriate for learning systems [Mazac et al., 2014, Guériau et al., 2016]. An *AMAS* is a complex artificial system composed of fine-grained agents promoting the emergence of expected global properties. It allows to cope with the complexity of the world (openness, non-linearity, dynamics, distributed information, noisy data and unpredictability) as defined by Ashby [Ashby and Goldstein, 2011]. Numerous experiments have shown such properties in areas such as the control of biological processes [Videau, 2011], the optimal control of motors [Boes, 2014] or robotics learning

[Verstaevel, 2016]. The late in line is *Agnostic MOdEl Builder by self Adaptation (AMOEBA)*, an agnostic model builder that brings distributed *Online Learning* and *Knowledge Generalization* in continuous environments [Nigon, 2017].

Context Learning Formalism

This section presents the functioning of the *AMOEBA* system [Nigon et al., 2017] where I also define a formalism for *Context Learning*. This formalism seeks to generalize the learning paradigm of *AMOEBA* by providing abstraction in relation to the learning models to converge toward absolute model independence.

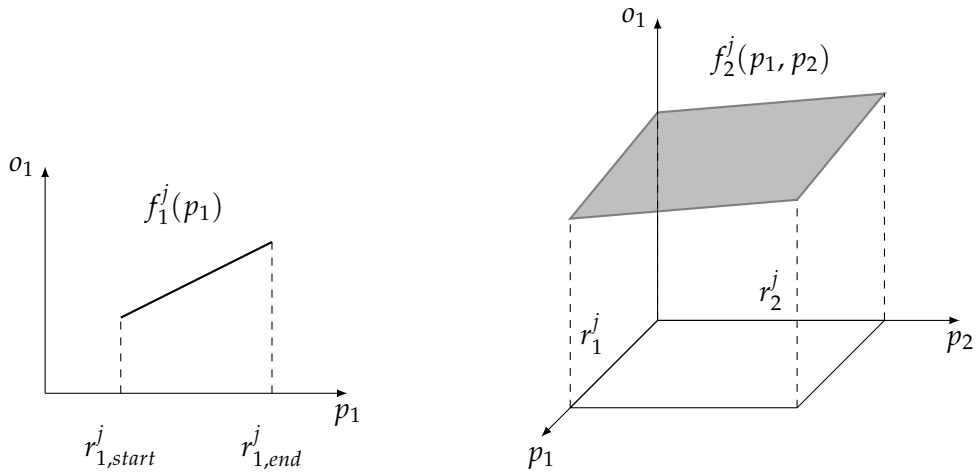
Context Learning is a problem of exploring a search space with n dimensions and estimating a *local model* based on any *Machine Learning* technique (neural networks, linear regression, SVMs, nearest neighbor, decision trees, k-means...). An instance of the learning system learns an output called ***prediction vector*** $\mathcal{O}'_m = [o'_1, \ldots, o'_k, \ldots, o'_m] \in \mathbb{R}^m$ according to a *hidden function* $\mathcal{F}(\mathcal{P}_n) = \mathcal{F}(p_1, \ldots, p_i, \ldots, p_n) = \mathcal{O}_m$ with $\mathcal{O}_m = [o_1, \ldots, o_k, \ldots, o_m] \in \mathbb{R}^m$ the desired predictions. $\mathcal{P}_n = [p_1, \ldots, p_i, \ldots, p_n] \in \mathbb{R}^n$ is the vector of inputs called the ***perceptions***. The vector $\mathcal{L}_{n,m} = [\mathcal{P}_n, \mathcal{O}_m]$, composed of *perceptions* associated with desired predictions, defines a ***learning situation***, which is similar to a *schema* in Piaget's theory. The learned models are managed by *Context Agents* $\mathcal{C}_n^j$ with j the jth pavement in dimension n which represents a part of the *schema*. A *Context Agent* is an intelligent autonomous agent that locally represents a part of the global function $\mathcal{F}$ with a *local model* $f_n^j(p_1, \ldots, p_i, \ldots, p_n) = \mathcal{O}_m^j$ with $\mathcal{O}_m^j = [o_1^j, \ldots, o_k^j, \ldots, o_m^j] \in \mathbb{R}^m$ the *local prediction vector* of the *Context Agent* $\mathcal{C}_n^j$. The *Context Agent* representation is a parallelotope of dimension n associated with a *Machine Learning* model. The parallelotope is defined by *validity ranges* $\mathcal{R}_n^j = [r_1^j, \ldots, r_i^j, \ldots, r_n^j]$ with $r_i^j = [r_{i,start}^j, r_{i,end}^j]$ which represents a validity interval on a perception p_i (Fig. 4.1).



(a) 1 Perception (b) 2 Perceptions (c) 3 Perceptions

Figure 4.1 – Context Agents *validity ranges*

The *Context Agents* have a *confidence* $c^j \in \mathbb{Z}$ to evaluate themselves in relation to others. A *Context Agent* is therefore defined by its *validity ranges*, its *local model* and its *confidence* $\mathcal{C}_n^j = \{\mathcal{R}_n^j, f_n^j, c^j\}$.



(a) 1 Perception (b) 2 Perceptions

Figure 4.2 – Context Agents linear regression models

Implemented Model

In the work of [Nigon et al., 2017], *Context Agents* $\mathcal{C}_n^j$ have local linear regression models as *local models* f_n^j (Fig. 4.2). For each vector of *perceptions* $\mathcal{P}_n$ in the *validity ranges*, a *Context Agent* provides a *local prediction vector* $\mathcal{O}_1^j \in \mathbb{R}$ calculated from the coefficients of its model $[a_0^j, \ldots, a_i^j, \ldots, a_n^j] \in \mathbb{R}^{n+1}$ and *perceptions* $\mathcal{P}_n$ according to the following equation:

$$\mathcal{O}_1^j = o_1^j = f_n^j(p_1, p_i, \cdots, p_n) = \sum_{i=1}^{n} a_i^j.p_i + a_0^j$$

In this case, the *prediction vectors* are real values $\mathcal{O}'_1$ and $\mathcal{O}_1$. A *learning situation* is then $\mathcal{L}_{n,1} = [\mathcal{P}_n, \mathcal{O}_1] = [\mathcal{P}_n, o_1]$. All along this work we will intend to disregard the implement model to seek absolute abstraction towards the learning process.

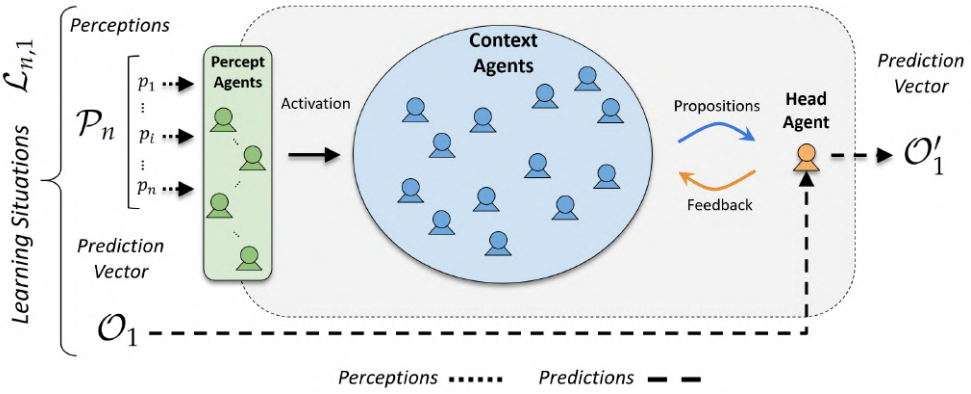


Figure 4.3 – *Context Learning* with *AMOEBA*

Exogenous Learning Rules

Learning with *Context Agents* in *AMOEBA* [Nigon et al., 2016] is based on a *Multi-Agent System* (Fig. 4.3) and several simple rules that are briefly presented in the following. The *Percept Agents* enable to activate the relevant *Context Agents* and the *Head Agent* is responsible for giving an output prediction and back-propagating the feedback from the oracle to the *Context Agents*. Each execution cycle is either a learning cycle or an exploitation cycle. For learning cycles, the input is a *learning situation* $\mathcal{L}_{n,1} = [\mathcal{P}_n, \mathcal{O}_1]$ and for exploitation cycles, the input is an *exploitation situation* $\mathcal{E}_n$ that is only *perceptions* $\mathcal{E}_n = [\mathcal{P}_n]$. The learning is characterized as passive and exogenous because all *learning situations* are exogenous and passive *learning situations*. They are *exogenous learning situations* because they are provided by an external entity (i.e. an oracle). They are *passive learning situations* because the learning mechanisms does not chooses the *learning situations*, it receives them in a passive way.

Learning Cycles

We define $\mathcal{L}_{n,1}^{c_l} = [\mathcal{P}_n^{c_l}, \mathcal{O}_1^{c_l}]$ the *learning situation* associated to the cycle c_l. During a learning cycle c_l, if the *perceptions* $\mathcal{P}_n^{c_l}$ belong to the *validity ranges* of a *Context Agent*, it is a *Valid Context Agent*. It proposes a *local prediction vector* with its *local model*. If there are several *Valid Context Agents*, the one with the closest prediction to the oracle's *prediction vector* is selected. It is called the *Best Context Agent* for the current execution cycle. If the *Best Context Agent* gives a good prediction, it increments its *confidence*. To know if the prediction of a *Context Agent* is good or bad, an *error margin* and an *inaccuracy margin* are used. They are given by the user of the learning mechanism. A prediction is good if the error with the oracle's prediction $\mathcal{O}_1^{c_l}$ is less than the *inaccuracy margin*. A prediction is bad if the error with the oracle's prediction is more than the *error margin*. Then, all *Valid Context Agents* reorganize themselves by following the adaptive behaviors below. The self-organization of *Valid Context Agents* corresponds to abnormal situations that require special treatments. These situations are called *Non Cooperative Situations (NCS)*, and provide the four following rules:

- ▷ **Bad Prediction *NCS*** The error between the prediction of a *Valid Context Agent* $\mathcal{O}_1^{j,c_l}$ and the oracle's prediction $\mathcal{O}_1^{c_l}$ is beyond the *error margin*. The *Context Agent* should not be valid for these *perceptions*. It modifies its *validity ranges* to exclude them and decrements its *confidence* c^j. See [Verstaevel et al., 2017] for further details on *validity ranges* modifications.
- ▷ **Inaccurate Prediction *NCS*** The error between the prediction of a *Valid Context Agent* $\mathcal{O}_1^{j,c_l}$ and the oracle's prediction $\mathcal{O}_1^{c_l}$ is greater than the *inaccuracy margin* but less than the *error margin*. The *Context Agent* adds to its *local model* the current *learning situation* $\mathcal{L}_{n,1}^{c_l}$ and it decrements its *confidence*.
- ▷ **Uselessness *NCS*** A *validity range* of a *Valid Context Agent* is less than a critical size in one of the *perceptions*. The *Context Agent* is considered unnecessary and is subtracted from the system.
- ▷ **Unproductivity *NCS*** There aren't any *Context Agents* to cover the current *perceptions* $\mathcal{P}_n^{c_l}$ The closest *Context Agent* that gives a good prediction extends its ranges towards the

perceptions. If it fails to include them or no such *Context Agent* exists, a new one is created. Euclidian distance is used to determine relative distances between the *perceptions* and a *Context Agent*.

Exploitation Cycles

During an exploitation cycle, if they are several *Valid Context Agents*, the one with the higher *confidence* is the *Best Context Agent*. It provides the prediction output $\mathcal{O}_1^{c_l\prime}$. If there aren't any *Valid Context Agents*, the closest *Context Agent* to the *perceptions* is designated as the *Best Context Agent*.

Discussion

Learning by *Endogenous Feedback* is not present in the *Context Learning* approach as *Context Agents* only learn from *exogenous learning situations*. *Context Learning* is fully agnostic as it is not specialized in any application domain. It also exhibits promising leads towards model independence as the learning rules do not depend on the underneath regression model.

Context Learning thank to its *AMAS* foundations is well suited for *Lifelong Learning* as it permits both exploitation and enrichment of a the learning mechanism. [Verstaevel et al., 2017] by introducing the Self-Adaptive Context Learning (SACL) Pattern proved that *Context Learning* is suitable for both *Supervised Learning* and *Reinforcement Learning* approaches. As agents in an *AMAS* remain active throughout their life, they confer intrinsic properties to *AMAS* to carry out *Lifelong Learning*.

Context Learning is on its way to handling *Knowledge Generalization* as it is defined in the *Machine Learning* community. Indeed, the self-adaptability of *AMAS* confers intrinsic dynamic properties to *Context Learning* allowing it to potentially reuse knowledge from a *Context Agent* to another.

The construction of *Context Learning* allows learning or exploitation anytime after the first *learning situation*. Thus, *Online Learning* is fully operational in this approach.

Self-Observation is an intrinsic property of *Context Learning* as *Multi-Agent Systems* handle multiple levels of granularity and distribute a problem into several entities. Each entity, by interacting with other ones, provides introspection concerning the collective of agents.

Context Learning enables model generalization in a continuous space but it has not been applied in a *Multi-Task Learning* setting. One important feature is that it can differentiate whether or not *perceptions* have already been experienced.

Thanks to a limited neighborhood of the agents, the complexity of *Multi-Agent Systems* can be linearly dependent on the number of agents which is a *Scalability* quality. However, constructivist approaches are well known for their *Scalability* issues.

As knowledge is represented locally, any *Context Agent* can provide a prediction for an unseen situation. Even with few learning experiences, the closest known knowledge (i.e. the closest *Context Agent*) can give predictions for unexplored situations.

According to recent studies, *MAS* offer promising properties for *Explainability* in *AI*

[Alzetta et al., 2020]. *Multi-Agent Systems* promote conceptual integrity in the engineering of complex software systems and serves the purpose of social Explainable *AI* (XAI).

4.5 Endogenous Learning Challenges

This section summaries the presented state of the art and defends the chosen learning approach in this thesis: *Context Learning*. In order to achieve *Endogenous Learning* with the *Context Learning* approach, its shortcomings are detailed.

4.5.1 Global Synthesis

In the chapter 1, we presented the set of fundamental features a learning system must have to face the properties of Sociotechnical Environments. The objectives that we proposed to focus on are *Endogenous Feedback*; *Agnosticity*; *Lifelong Learning*; *Online Learning*; *Self-Observation*; *Knowledge Generalization*; *Scalability*; learning from *Any Data Amount* and *Explainability*. We presented an overview of today's learning methods in *AI* and Robotics, highlighting their advantages among the exhibited characteristics.

We have seen in chapter 2 that classical learning approaches generally lack *Lifelong Learning*, *Online Learning* and *Knowledge Generalization*. These are the main challenges of *Lifelong Learning* in the *Machine Learning* community. *Lifelong Learning* research gave birth to *Transfer Learning*, *Online Learning* and *Multi-Task Learning* to focus on a new generation of learning where abstraction and generalization are central. Indeed, to achieve *Artificial General Intelligence* (*AGI*), connection between experiences and knowledge are an essential challenge that needs to be overcome. Then, we have seen in chapter 3 that the major missing topic in the *Lifelong Learning* approaches is the self-motivation of learning. The human behavior is impacted by its social interactions but also its internal curiosity and its capacity of self-teaching. Self-Learning is a key to flexible *AGI*. Self-Learning must not be mistaken with *Unsupervised Learning*. Self-learning has to be seen in a more general and broad sense. It should be understood as a system's ability to exploit or explore the training data and its environment by itself [Lyre, 2020]. It is in this way that *Developmental Learning* went by introducing *Intrinsic Motivation* as a way of providing internal motivation in learning mechanisms. *Developmental Learning* aims at developing infant-like intelligence to build adult-like intelligence as an emergence of incremental experiences and explorations. Related to this topic *Meta-Learning* and *Few Shot Learning* bring an abstract layer to learning mechanism to also converge toward human-like intelligence. Finally, *Context Learning* introduced an online agnostic distributed approach to learning that gives promising leads towards learning in complex environments.

Among all the presented learning paradigms, what is most missing is the endogenous view of learning and the introspection within learning mechanisms. *Context Learning* offers a malleable learning framework and many flexible opportunities towards the manipulation of knowledge fragments. It is this path that we will explore in this work. *Context Learning* and *AMAS* provide many characteristics to cope with the complexity of the world. The distribution of learning fragment is a key to adaptive learning. Systems that involve many components that adapt or learn as they interact are at the heart of important contemporary prob-

lems [Holland, 2006]. Knowledge in *AMAS* learning is divided in autonomous fragments of knowledge that cooperate together. This cooperative reasoning is thus a way for generating a self-enrichment of knowledge. The creation of internal requests, which are the *Endogenous Feedback*, can lead to *Active Learning* and *Self-Learning*. Taking inspiration from *Developmental Learning* to design learning in a self-motivated way is an important of our concerns. We take inspiration from *Developmental Learning* work but we are not yet experimenting on blank robotic systems. Self-organization in *AMAS* learning is a powerful tool to provide autonomy of knowledge fragments. Through the generation of *Endogenous Feedback*, our goal is to enhance exploration such as knowledge among *Context Agents*. It is seemingly more favorable to distribute smaller subsets of data to different learning units and then collaborate with direct neighbors to find the optimal solution [Sayed, 2014]. Our work focuses on developing new cooperation rules in *Adaptive Multi-Agent Systems* to enable self-enrichment of knowledge by generating *endogenous learning situations* and seeking learning insufficiencies.

	Endogenous Feedback	*Agnosticity*	*Lifelong Learning*	*Online Learning*	*Self-Observation*	*Knowledge Generalization*	*Scalability*	*Any Data Amount*	*Explainability*
Supervised Learning	-	+	- +	- +	-	- +			
Unsupervised Learning	- +	+	- +	-	- +	- +	+		+
Semi-Supervised Learning	-	+	-	- +	- +	-			
Active Learning	- +	+	-	- +	+	-			
Reinforcement Learning	-	-	- +	+	-	- +	- +	- +	- +
Inverse Reinforcement Learning	- +	+	- +	+	-	- +	- +		- +
Learning by Demonstrations	-	+	- +	-	-	- +			
Intention Learning	- +	+	- +	-	-	- +	-	- +	
Transfer Learning	- +	+	- +	- +	- +	+	-	- +	
Multi-Task Learning	-	+	- +	- +	-	+	- +		- +
Online Learning	- +	+	- +	+	- +	-	+	+	
Deep Learning	-	- +	- +	- +	- +	- +	- +	- +	-
Developmental Learning	+	+	- +	- +	+	- +	- +		-
Meta-Learning	- +	+	- +	- +	- +	- +	+	+	- +
Few Shot Learning	- +	- +	- +	- +	-	+	- +	+	- +
Context Learning	-	+	- +	+	+	- +	- +	+	+

Table 4.1 – Synthesis on literature learning approaches with respect to the thesis objectives. Legend: + positive match ; - + in development or relative match; - negative match.

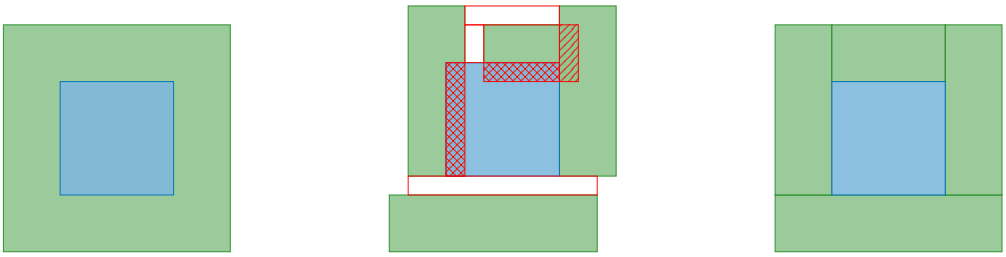
One important feature in *AI* nowadays is also *Explainability* (XAI). To verify a system, improve it, understand recommendations, make it learn from itself and for legislation reason *Explainability* is a growing issue in *AI* [Samek et al., 2017]. *Explainability* is being explored in many learning paradigms but it is yet an important challenge [Samek and Müller, 2019]. Designing mechanisms in order to internally enhance a learning mechanism requires a perfect understanding of its mechanics. *AMAS* possess powerful tools to explain *AI* with different degrees of granularity especially with the help of frameworks and user interfaces

[Drogoul et al., 2013, Perles et al., 2018].

Table 4.1 summaries the strengths and weaknesses of all the presented learning techniques according to our objectives.

4.5.2 Context Learning Shortcomings

On a simple 2D classification problem, with two linear models represented by two different colors (Fig. 4.4a), the exogenous learning rules do not allow the system to converge to an ideal representation (Fig. 4.4c). An ideal representation would be a learning with the minimum of *Context Agents*, without any overlap or gaps. We observe several types of unfulfilled situations which can be solved by endogenous reasoning processes. The exogenous rules do not protect against imprecise exploration (Fig. 4.4b). Incompetencies (white or blank areas) are not detected and are only partially filled in when the learning system encounters them. Conflicts and concurrencies (overlapping hatched *validity ranges*) are not taken into account when changing the *validity ranges* of the *Context Agents*. A mechanism using neighborhood properties is missing: for each *NCS* of the exogenous learning rules (section 4.4), all resolutions are made independently of the *Context Agents* neighbors if they are any.



(a) Models to learn (b) Example of imprecise learning (c) Example of ideal learning

Figure 4.4 – Simple learning problem of 2 linear models symbolized by different colors

The aim of this thesis work is to introduce additional mechanisms to internally deal with theses inconsistencies and provide *Context Learning* with better lifelong and generalization properties.

Lifelong Learning by Endogenous Feedback
Application to a Robotic System

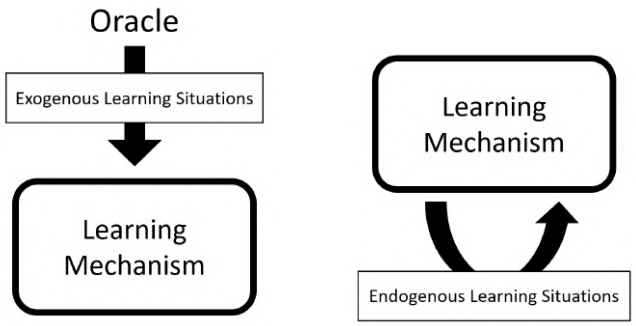
Contribution

5 Endogenous Learning Principles

In this chapter, I detail the construction of Endogenous Context Learning. It consists in the identification of learning hypotheses that allows to identify learning inaccuracies. The learning inaccuracies are learning defects that can be detected with local agents negotiations. To add local communication between the agents of a Multi-Agent based learning mechanism, the concept of neighborhood is introduced. Learning inaccuracies are characterized to determined how to detect and solve them. Their processing order is explained according to their priority. Cooperative Neighborhood Learning is presented. It is a mechanism allowing to generate internal learning situations to enhance the model of agents through local communication.

In recent years, *AI* challenges have evolved. With the ultimate goal of conceiving an *Artificial General Intelligence* (*AGI*), the work on artificial learning focuses on lifelong and multi-task problems. In robotics, the imitation of infants cognitive stages inspired incremental learning with internal motivation. Transfer of knowledge and self-taught learning are two main challenges in *AI*. In this thesis, the focus is made on self-learning that we call *Endogenous Learning*.

Endogenous Learning is when a learning mechanism teaches itself new *learning situations*. Its opposite is *Exogenous Learning* (Fig. 5.1). It is what we call *Supervised Learning* in the field of *Machine Learning* (section 2.1). When a mechanism uses *Exogenous Learning*, it learns from an external entity that we previously called the oracle. This oracle provides *exogenous learning situations*. When a mechanism uses *Endogenous Learning*, it learns from internal *learning situations* that we call *endogenous learning situations*.



(a) *Exogenous Learning* (b) *Endogenous Learning*

Figure 5.1 – *Exogenous Learning* VS *Endogenous Learning*

Previously presented, a *learning situation* $\mathcal{L}_{n,m}$ is composed of a vector of *perceptions* $\mathcal{P}_n$ and a *prediction vector* $\mathcal{O}_m$ such that $\mathcal{L}_{n,m} = [\mathcal{P}_n, \mathcal{O}_m]$. From now on, we differentiate *exogenous learning situations* $\mathcal{L}^{exo}_{n,m}$ and *endogenous learning situations* $\mathcal{L}^{endo}_{n,m}$. *Exogenous learning situations* are defined by an *exogenous prediction vector* $\mathcal{O}^{exo}_m$ provided by an oracle, an external entity, and any *perceptions* (*exogenous perceptions* $\mathcal{P}^{exo}_n$ or *endogenous perceptions* $\mathcal{P}^{endo}_n$): $\mathcal{L}^{exo}_{n,m} = [\mathcal{P}^{exo}_n / \mathcal{P}^{endo}_n, \mathcal{O}^{exo}_m]$. *Endogenous learning situations* are composed of an *endogenous prediction vector* $\mathcal{O}^{endo}_m$ and *endogenous perceptions* $\mathcal{P}^{endo}_n$ self-generated by the learning mechanism: $\mathcal{L}^{endo}_{n,m} = [\mathcal{P}^{endo}_n, \mathcal{O}^{endo}_m]$.

The generation of *endogenous learning situations* implies two questions:

- Where and when to generate the *endogenous perceptions* in the learning space ?
- What *endogenous prediction vectors* should be learned in the generated *endogenous perceptions* ?

This chapter presents the motivations and theoretical bases of *Endogenous Learning* and *Endogenous Context Learning*.

Endogenous Context Learning is an enhancement of *Context Learning*. Until now, *Context Learning* only used *exogenous learning situations* when *Endogenous Context Learning* combines *exogenous learning situations* and *endogenous learning situations*. The first question is answered by the identification of inconsistencies in *Context Learning*. A mechanism of *neighborhood* is added to enable internal communication among the knowledge agents that are the *Context Agents*. All possible learning inconsistencies are explored and are defined as *learning inaccuracies*. The resolutions of *learning inaccuracies* are presented in the *AMAS* framework as *Non Cooperation Situations* (*NCS*). The mechanisms are operational regardless of the number of *perceptions*. To simplify the illustrations, only two dimensions are represented. The priority of *learning inaccuracies* is introduced to select the strategy for the execution of the resolutions. Finally, a response to the second question of *Endogenous Learning* is answered by presenting *Cooperative Neighborhood Learning*, a way to internally generate knowledge by sharing *endogenous learning situations* between *Context Agents*.

5.1 Endogenous Learning Objectives

This section presents the high level learning objectives that lead to *Endogenous Context Learning*. These objectives come from the definition of an ideal learning inspired from some of the AI challenges that we have seen in chapters 3 and 4. An ideal or optimal learning is defined by the following hypotheses:

- ***Curiosity Hypothesis***: Curiosity is the source of learning [Forestier and Oudeyer, 2016]. A learner internally seeks novelty and explores new experiences. The learner also looks for contradictions to develop better understanding of its environment.
- ***Continuity Hypothesis***: The discreteness and continuity of our nature is an eternal question in physics theory [Hagar, 2014]. However, the world, at its macroscopic scale is continuous. It is humans, through their perception of it, that make it discrete [Holmgren, 2014]. Learning is seeking discontinuities in a continuum.
- ***Agnosticity Hypothesis***: Learning is agnostic. It adapts to new needs and it is always open for new experiences as different as they can be. It does not need any prior knowledge to gather experience.
- ***Generalization Hypothesis***: To learn is to generalize thanks to several experiences. Generalization is made across time and space as experiences can be related through history and context.

5.2 Endogenous Context Learning Objectives

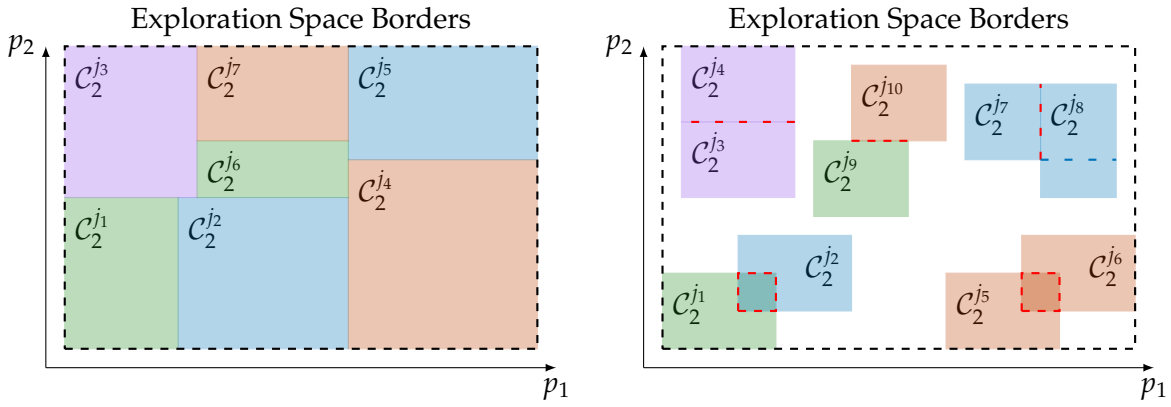
The learning hypotheses can be transformed using the vocabulary of the *Context Learning* framework that we have introduced section 4.4. The following hypotheses are from the point a view of learning with *Context Agents*, their *validity ranges* and their *local models*.

- ***Curiosity Hypothesis***: This hypothesis concerns the *validity ranges* of the *Context Agents*. *Context Agents* completely should fill the exploration space. There aren't any unknown areas (void areas). If it is the case, these areas should be discovered. There aren't any conflicts nor concurrencies between the *Context Agents validity ranges* (overlap areas). If so, these situations should be solved.
- ***Continuity Hypothesis***: This hypothesis concerns the *local models* and the *validity ranges* of the *Context Agents*. It is assumed that *Context Agents* should seek continuity with other adjacent *Context Agents*. However, they should also be able to detect cases of discontinuities between their models.
- ***Agnosticity Hypothesis***: This hypothesis only concerns the embedded learning models of the *Context Agents*. The learning is independent from the underlying models. All the implemented mechanisms to satisfy the learning hypotheses should not refer to any particular learning model.
- ***Generalization Hypothesis***: This hypothesis concerns the number of *Context Agents* that represent the learning. The exploration space should be represented by as few *Context Agents* as possible. As *Context Agents* generalize over continuous spaces, the bigger their

validity ranges are, the more *Context Agents* generalize. The generalization also concerns the models of *Context Agents*. When it is adequate, *Context Agents* should share their model to allow transfer of knowledge.

The learning objectives can be categorized into exploration objectives and model objectives. Exploration objectives concern the *validity ranges* of the *Context Agents* and model objectives concern the *local models* of the *Context Agents*.

5.2.1 Exploration Objectives



(a) Example of ideal exploration

(b) Example of imperfect exploration.

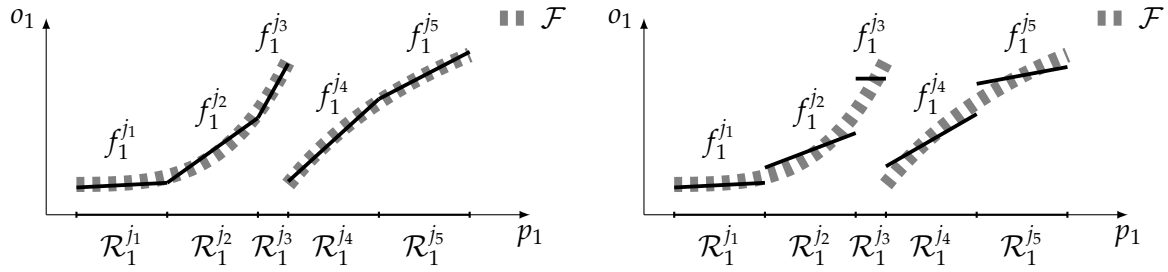
Figure 5.2 – Schemas of ideal and imperfect explorations. The imperfect exploration shows all possible exploration inaccuracies in *Context Learning*. Different colors represent different *local models*

From these hypotheses, in opposition to an ideal learning exploration (Fig. 5.2a), several exploration *learning inaccuracies* can be identified (Fig. 5.2b). They are represented with red dashed lines figure 5.2b. *Learning inaccuracies* can occur in the following examples.

- $\mathcal{C}_2^{j_1}$ and $\mathcal{C}_2^{j_2}$ are inaccurate because they pave a common zone with different models. This goes against the *Curiosity Hypothesis*.
- $\mathcal{C}_2^{j_5}$ and $\mathcal{C}_2^{j_6}$ are inaccurate because the pave a common zone with similar models. This goes against the *Curiosity Hypothesis*.
- $\mathcal{C}_2^{j_3}$ and $\mathcal{C}_2^{j_4}$ are inaccurate because they pave a zone that could be represented by only one *Context Agent*. This goes against the *Generalization Hypothesis*.
- $\mathcal{C}_2^{j_7}$ and $\mathcal{C}_2^{j_8}$ are inaccurate because $\mathcal{C}_2^{j_7}$ could pave a bigger zone than $\mathcal{C}_2^{j_8}$ by taking a part of $\mathcal{C}_2^{j_8}$ *validity ranges*. This goes against the *Generalization Hypothesis*.
- $\mathcal{C}_2^{j_9}$ and $\mathcal{C}_2^{j_{10}}$ are inaccurate because they are adjacent and their models are different. This is related to the *Continuity Hypothesis*. The frontier needs to be clarified as continuous or discontinuous.
- Finally, according to the *Curiosity Hypothesis*, the remaining inaccuracies are all the void areas that are not represented by *Context Agents* inside the explored space.

5.2.2 Model Objectives

The *Agnosticity Hypothesis* states that the learning must be independent of the learning models. However, for the illustration purposes of this work, models will always be represented by linear functions. According to the *Continuity Hypothesis*, the learned *hidden function* $\mathcal{F}$ is piecewise continuous (Fig. 5.3a). In the continuous parts, the *Context Agents* models should meet at their frontiers. As discontinuities can also exist in the *hidden function*, they should be detected so that the models can learn them. Figure 5.3b shows an example of non ideal learning where all *local model* frontiers are discontinuous and do not match the *hidden function* $\mathcal{F}$.



(a) Example of ideal learning

(b) Example of imprecise learning

Figure 5.3 – Schemas of ideal and imprecise learning for linear *local models*. The search space is supposed entirely explored and represented by *Context Agents validity ranges*. The gray thick dashes represent the *hidden function* $\mathcal{F}$, and the black line are the learned *local models* $f_1^{j_i}$.

5.3 Neighborhood Definitions

We have identified that the learning can be inaccurate in the *Context Learning* paradigm thanks to the learning hypotheses. In classical *Context Learning*, *Context Agents* only adapt to *learning situations* given by an oracle. *Context Agents* are not aware of other any neighboring *Context Agents*. To allow this awareness and permit communication between neighboring *Context Agents*, the concept of *neighborhood* is added to *Context Learning*.

What is invariant for both the learning and exploitation cycles is the vector of *perceptions*. For each *perceptions* $\mathcal{P}_n$, a set of *Neighbor Context Agents* is computed from an area around the *perceptions*. This ensures that the resolutions that are implemented are local as only a subset of all the *Context Agents* of the *MAS* is considered. The *Neighbor Context Agents* are the set of *Context Agents* that will communicate during a cycle to detect possible inaccuracies. The *neighborhood* adds local interactions between *Context Agents*. By allowing communication between *Context Agents* in a local part of the space of *perceptions*, we aim to control the complexity of these interactions for large numbers of *perceptions* and large quantities of *Context Agents*. The *neighborhood* area depends on a core value in the learning process which is defined as the *Context Agents range creation radius* $r_i^{creation}$ for a perception p_i.

The *range creation radius*

$$r_i^{creation} = (p_i^{max} - p_i^{min}).p^{\mathcal{R}} \tag{5.1}$$

with p_i^{max} and p_i^{min} the maximum and minimum experienced values by the learning mechanism on the perception p_i. The ***validity ranges precision*** $p^{\mathcal{R}}$ is a parameter chosen by the user of the mechanism. It represents the percentage of an explored perception p_i that can be used to represent the explored space at the creation of the *validity ranges* of a *Context Agent*. $p^{\mathcal{R}} \in]0, 1[$. 0 is excluded because it gives a null radius. 1 is excluded by convention, it represents the whole experienced perception space. The *validity ranges precision* allows to choose the sizes of *Context Agents* at their creation. From this, the following distances result.

The *neighborhood radius*

$$r_i^{\mathcal{N}} = \alpha^{\mathcal{N}}.(p_i^{max} - p_i^{min}).p^{\mathcal{R}} = \alpha^{\mathcal{N}}.r_i^{creation} \tag{5.2}$$

with $\alpha^{\mathcal{N}} \in \mathbb{R}$ the ***neighborhood radius coefficient***. A *Context Agent* $\mathcal{C}_n^j$ is considered as a neighbor of the *perceptions* if its *validity ranges* intersect a *neighborhood* area surrounding the current *perceptions*. For each perception p_i the following condition must be satisfied: $[p_i - r_i^{\mathcal{N}}; p_i + r_i^{\mathcal{N}}] \cap r_i^j \neq \varnothing$. The *neighborhood radius* being proportional to $r_i^{creation}$, it is directly influenced by the *validity ranges precision* $p^{\mathcal{R}}$ and by the size of the explored space in each perception p_i. The *validity ranges precision* enables to set the creation size of the *Context Agents validity ranges* and the *neighborhood* at the same time. The *neighborhood radius coefficient* enables to modify the width of the *neighborhood* independently of the *validity ranges precision*.

The *prediction neighborhood radius*

$$r_{o_k}^{\mathcal{N}} = \alpha^{\mathcal{N}}.(o_k^{max} - o_k^{min}).p^{\mathcal{R}} \tag{5.3}$$

with o_k^{max} and o_k^{min} the maximum and minimum experienced predictions by the learning mechanism on the prediction values o_k of the *exogenous prediction vectors* $\mathcal{O}_m^{exo}$. *Context Agents* $\mathcal{C}_n^{j_1}$ and $\mathcal{C}_n^{j_2}$ are considered as prediction neighbors if all the distances between their last predictions $\mathcal{O}_m^{j,last}$ are lesser than the *prediction neighborhood radius* ($|o_k^{j_1,last} - o_k^{j_2,last}| < r_{o_k}^{\mathcal{N}}$ for the prediction o_k). The use of the last *prediction vector* of a *Context Agent* $\mathcal{O}_m^{j,last}$ is not dependent on any particular model. The *prediction neighborhood radius* is set with *validity ranges precision* and the *neighborhood radius coefficient*, and it is proportional to the range of the experiences oracle values.

The *influence radius*

$$r_{j,i}^{\mathcal{I}} = (1 + \alpha^{\mathcal{I}}).r_i^j \tag{5.4}$$

with $\alpha^{\mathcal{I}} \in \mathbb{R}$ the ***influence radius coefficient***. A large *Context Agent* is an agent that has generalized lot of information. With the area-based *neighborhood*, such a *Context Agent* may not be considered as a *Neighbor Context Agent* when it should be. For this purpose, *Context Agents* have an *influence* zone outside their *validity range* that is proportional to their *validity ranges*. If the current *perceptions* are within this zone of influence, the *Context Agent* $\mathcal{C}_n^j$ is also in the *neighborhood* ($r_{i,center}^j \in [p_i - r_{j,i}^{\mathcal{I}}; p_i + r_{j,i}^{\mathcal{I}}]$ for one perception p_i with $r_{i,center}^j$ the center of a *validity range*).

Table 5.1 is a summary of the parameters involved in the configuration of the *neighborhood* mechanisms.

	Name	Notation	Constrains	Domain
User Parameter	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$
User Distance	*range creation radius*	$r_i^{creation}$	($p^{\mathcal{R}}$)-dependent	$\mathbb{R}$
Designer Paramaters	*neighborhood radius coefficient*	$\alpha^{\mathcal{N}}$	> 0	$\mathbb{R}$
	influence radius coefficient	$\alpha^{\mathcal{I}}$	> 0	$\mathbb{R}$
Designer Distances	*neighborhood radius*	$r_i^{\mathcal{N}}$	($p^{\mathcal{R}}$,$\alpha^{\mathcal{N}}$)-dependent	$\mathbb{R}$
	prediction neighborhood radius	$r_{o_k}^{\mathcal{N}}$	($p^{\mathcal{R}}$,$\alpha^{\mathcal{N}}$)-dependent	$\mathbb{R}$
	influence radius	$r_{j,i}^{\mathcal{I}}$	($\alpha^{\mathcal{I}}$)-dependent	$\mathbb{R}$

Table 5.1 – Table of user and designer parameters and distances for the configuration of the *neighborhood* and the *prediction neighborhood*.

Figure 5.4 shows an example of *neighborhood* and *influence* areas with the associated *Neighbor Context Agents*. ${}^{\mathcal{N}}\mathcal{C}_2^{j_1}$ is a *Neighbor Context Agent* because its *validity ranges* intersect the *neighborhood* area. ${}^{\mathcal{N}}\mathcal{C}_2^{j_2}$ is a *Neighbor Context Agent* because its *influence* zone contains the current *perceptions*. $\mathcal{C}_2^{j_3}$ is not a *Neighbor Context Agents* because its *validity ranges* don't intersect the *neighborhood* area and its *influence* area does not intersect the current *perceptions* $p_1^{c_l}$ and $p_2^{c_l}$.

Figure 5.5 shows an example of *prediction neighborhood* with linear models, for one prediction and one perception. The *Context Agent* $\mathcal{C}_1^{j_1}$ associated with the model $f_1^{j_1}$ is a prediction neighbor of the *Context Agent* $\mathcal{C}_1^{j_2}$ associated with the model $f_1^{j_2}$ because the distance between their last predictions is lesser than the *prediction neighborhood radius* $r_{o_1}^{\mathcal{N}}$. On the contrary, the

Context Agent $\mathcal{C}_1^{j_3}$ associated with the model $f_1^{j_3}$ is not a prediction neighbor because the distance between their last predictions is greater than the *prediction neighborhood radius* $r_{o_1}^{\mathcal{N}}$.

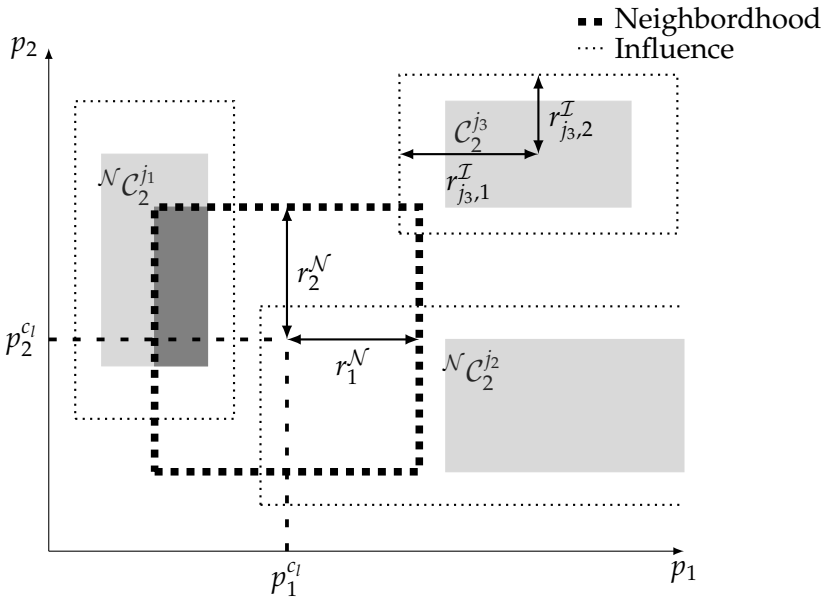


Figure 5.4 – Schema of the *neighborhood* of the *perceptions* $\mathcal{P}_2^{c_l}$ and the *Context Agents influences* for 2 perceptions p_1 and p_2.The *neighborhood* is represented with thick dashes and *influences* are represented with thin dashes.

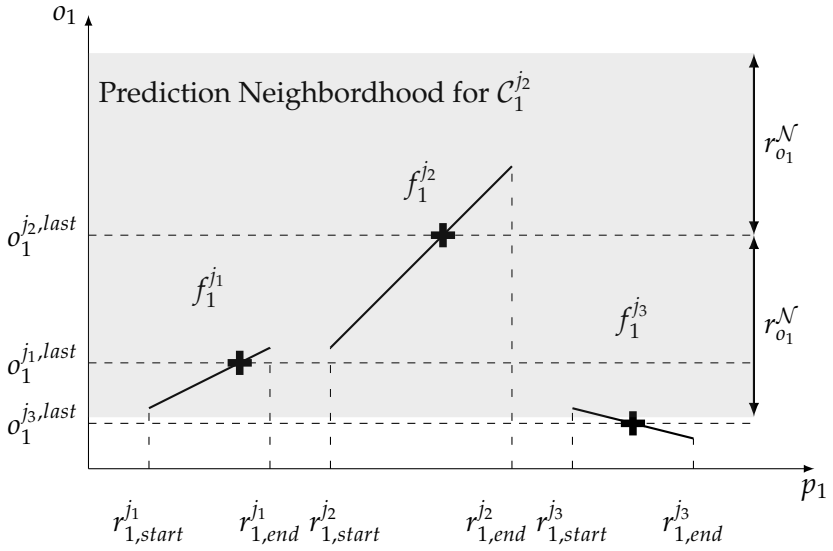


Figure 5.5 – Schema of *Prediction Neighbor Context Agents* for one perception p_1 and one prediction output o_1. $o_1^{j_1,last}$, $o_1^{j_2,last}$ and $o_1^{j_3,last}$ are the last predictions of the *Context Agents* $\mathcal{C}_1^{j_1}$, $\mathcal{C}_1^{j_2}$ and $\mathcal{C}_1^{j_3}$ represented by the local models $f_1^{j_1}$, $f_1^{j_2}$ and $f_1^{j_3}$. $\mathcal{C}_1^{j_1}$ and $\mathcal{C}_1^{j_2}$ are prediction neighbors whereas $\mathcal{C}_1^{j_3}$ is not.

5.4 Learning Inaccuracies

We have introduced a mechanism that allows *Context Agents* to locally communicate and interact. Thanks to this additional awareness between *Context Agents*, it is now possible for them to communicate and share information about their *validity ranges* and *local models* without scanning the whole collective of agents. In this section, the detection and resolution of *learning inaccuracies* are presented.

In *AMAS*, agents are designed thanks to two behaviors: the nominal behavior and the cooperative behavior. The nominal behavior gathers the actions to be done when an agent is in a cooperative state. The cooperative behavior defines the actions to be carried out in order to return to the cooperative state if the agent has deviated from it. The situations that enable to determine if an agent is in the cooperative behavior are called *Non Cooperative Situations* (*NCS*) [Georgé et al., 2011]. *NCS* are declined in different sub-categories that will be used to categorized the *learning inaccuracies*.

5.4.1 Communication in Context Agent Pairs

We have seen previously that for each cycle, a set of *Neighbor Context Agents* is computed. Each *Context Agent* of the *Neighbor Context Agents* communicates with the other neighbors to seek for *learning inaccuracies*. A *Context Agent* cannot know its relative position compared to another *Context Agent* until they have mutually shared their *validity ranges*. There is no representation of space in the list of *Neighbor Context Agents*. Let n be the number *Neighbor Context Agents* and k a subset of *Neighbor Context Agents* that shares their characteristics. Finding all the subsets is equivalent to finding all the $k-$combinations C_k^n in a set of n elements.

Only $2-$combinations and not superior combinations are considered because C_2^n is between linear dependency and degree 2 polynomial dependency on the number of elements. Beyond $2-$combinations C_k^n is above degree 2 polynomial dependency.

$$C_2^n = \binom{n}{2} = \frac{n!}{2!(n-2)!} \tag{5.5}$$

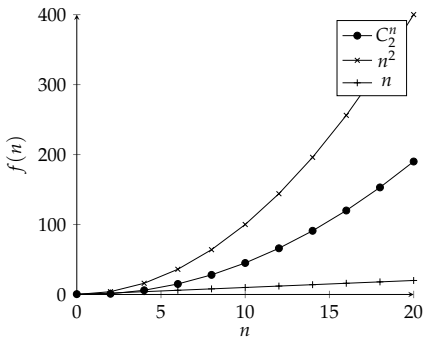


Figure 5.6 – Behavior comparison of $2-$combinations C_2^n with linear and degree 2 polynomial functions

This means that *learning inaccuracies* that involve more than two *Context Agents* will not

be solved in one resolution step. It will need several steps of two *Context Agents* resolutions. Thus, in the following, the definition of *learning inaccuracies* will only involve two *Context Agents* at a time.

5.4.2 Exploration Inaccuracies Detection

In this section, the characterization of exploration *learning inaccuracies* and their detection are detailed. In the first place, a formalism for *validity range* comparison is introduced and the used distances for the *learning inaccuracies* detection are defined. To formalize intersections between *validity ranges*, Allen's interval algebra is used. This algebra was initially made to formalize all the 13 possible relations between time intervals [Allen, 1983]. It is used for the relations between one dimension *validity ranges* (Tab. 5.2).

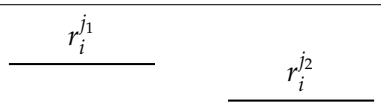
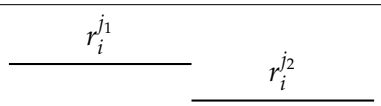
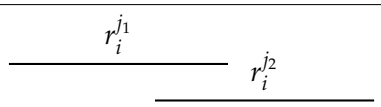
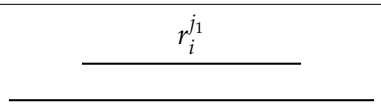
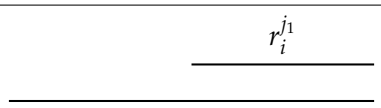
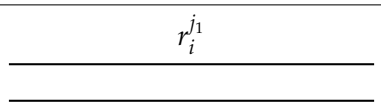
Relation	Illustration	Interpretation
$r_i^{j_1} < r_i^{j_2} \quad r_i^{j_2} > r_i^{j_1}$		$r_i^{j_1}$ and $r_i^{j_2}$ are disjointed
$r_i^{j_1}$ m $r_i^{j_2} \quad r_i^{j_2}$ mi $r_i^{j_1}$		$r_i^{j_1}$ meets $r_i^{j_2}$ (*i* stands for *inverse*)
$r_i^{j_1}$ o $r_i^{j_2} \quad r_i^{j_2}$ oi $r_i^{j_1}$		$r_i^{j_1}$ overlaps with $r_i^{j_2}$
$r_i^{j_1}$ s $r_i^{j_2} \quad r_i^{j_2}$ si $r_i^{j_1}$		$r_i^{j_1}$ starts $r_i^{j_2}$
$r_i^{j_1}$ d $r_i^{j_2} \quad r_i^{j_2}$ di $r_i^{j_1}$		$r_i^{j_1}$ is included by $r_i^{j_2}$
$r_i^{j_1}$ f $r_i^{j_2} \quad r_i^{j_2}$ fi $r_i^{j_1}$		$r_i^{j_1}$ ends $r_i^{j_2}$
$r_i^{j_1} = r_i^{j_2}$		$r_i^{j_1}$ is equal to $r_i^{j_2}$

Table 5.2 – 13 possible relations between one dimension *validity ranges*

All relations can be represented by intervals that we note $\overline{r_i^{j_1} \bullet r_i^{j_2}}$ with $\bullet$ any relation. For the relations $>$ and $<$, the interval is between the ranges $r_i^{j_1}$ and $r_i^{j_2}$. For the following,

the center of an interval will be written $\overline{r_i^{j_1} \bullet r_i^{j_2}}^{C}$.

5.4.2.1 Detection Distances

For the detection of exploration *learning inaccuracies*, the following distances need to be defined.

The *range similarity distance*

$$d_i^{\mathcal{R}_{sim}} = \alpha^{\mathcal{R}_{sim}}.r_i^{creation} \tag{5.6}$$

with $\alpha^{\mathcal{R}_{sim}} \in]0,1[$ the *range similarity coefficient*. This distance is a relaxation distance that enables to determine *validity ranges* similarities in the continuous space of *perceptions*.

The *minimum range distance*

$$d_i^{\mathcal{R}_{min}} = \alpha^{\mathcal{R}_{min}}.r_i^{creation} \tag{5.7}$$

with the *minimum range coefficient* $\alpha^{\mathcal{R}_{min}} \in]0,1[$ and $\alpha^{\mathcal{R}_{min}} < \alpha^{\mathcal{R}_{sim}}$. This distance is the exploration precision limit below which the mechanism no longer considers *validity ranges* for the detection of *learning inaccuracies*. This distance represents the utility limit in the space of *perceptions* for each perception p_i.

The *model similarity distance*

$$d_{sim}^{f} \tag{5.8}$$

The *model similarity distance* is a metric that defines the similarity of two models. This distance is model-dependent and needs to be defined by the user according to the learning models he uses.

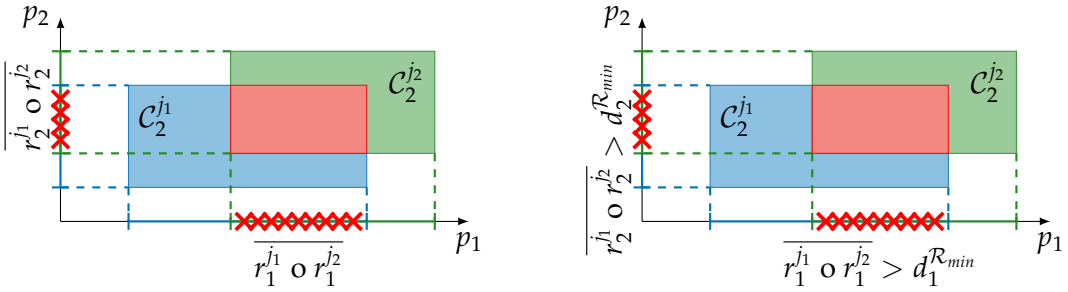
Table 5.3 is a summary of the parameters involved in the *learning inaccuracies* detection mechanisms.

	Name	Notation	Constrains	Domain
User Parameter	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$
User Distance	*range creation radius*	$r_i^{creation}$	$(p^{\mathcal{R}})$-dependent	$\mathbb{R}$
User Metric	*model similarity distance*	d_{sim}^{f}	model-dependent	$\mathbb{R}$
Designer Paramaters	*range similarity coefficient*	$\alpha^{\mathcal{R}_{sim}}$	$]0,1[$	$\mathbb{R}$
	minimum range coefficient	$\alpha^{\mathcal{R}_{min}}$	$]0,1[, \alpha^{\mathcal{R}_{min}} < \alpha^{\mathcal{R}_{sim}}$	$\mathbb{R}$
Designer Distances	*range similarity distance*	$d_i^{\mathcal{R}_{sim}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{R}_{sim}})$-dependent	$\mathbb{R}$
	minimum range distance	$d_i^{\mathcal{R}_{min}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{R}_{min}})$-dependent	$\mathbb{R}$

Table 5.3 – Table of parameters, distances and metrics for the *learning inaccuracies* detection mechanisms.

5.4.2.2 Conflict NCS Detection

A *Conflict NCS* is a *validity ranges* overlap between *Context Agents* that have different models according to the *model similarity distance*. For a learning problem with *n perceptions*, a theoretical overlap between two *Context Agents* is defined by *n validity ranges* overlaps $r_i^{j_1}$ o $r_i^{j_2}$ (Fig. 5.7a). In practice, an overlap occurs if the intersections lengths between *Context Agents validity ranges* are greater than the *minimum range distance* $d_i^{\mathcal{R}min}$ (Fig. 5.7b). $\overline{r_i^{j_1} \text{ o } r_i^{j_2}}$ represents the overlap distance between $r_i^{j_1}$ and $r_i^{j_2}$.

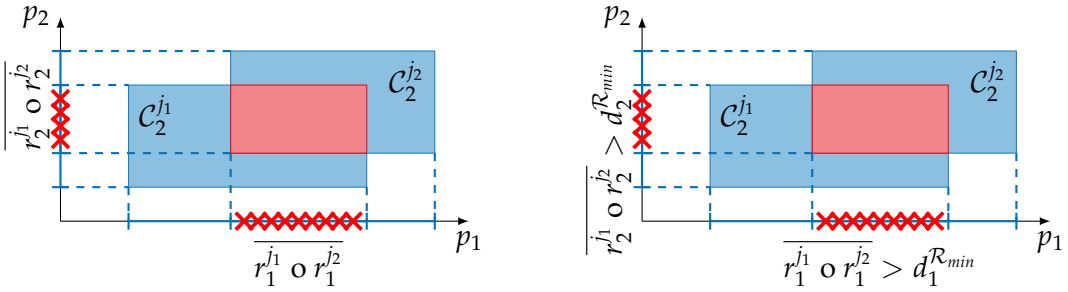


(a) Theoretical (b) Practical

Figure 5.7 – *Conflict NCS* Detection.

5.4.2.3 Concurrency NCS Detection

Concurrency NCS is a *validity ranges* overlap between two *Context Agents* that have similar models according to the *model similarity distance* d_{sim}^f. The geometry detection is the same as the *Conflict NCS* Detection. Figure 5.8a represents a theoretical *Concurrency NCS* and figure 5.8b represent a practical *Concurrency NCS*.



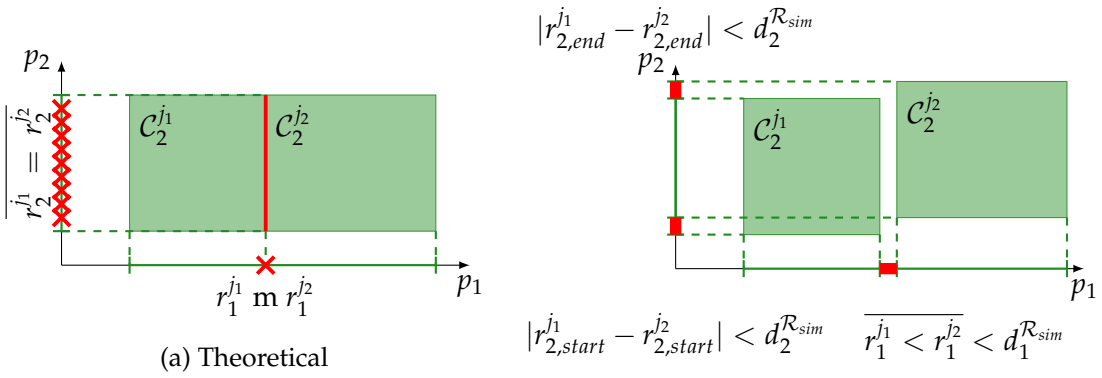
(a) Theoretical (b) Practical

Figure 5.8 – *Concurrency NCS* Detection.

5.4.2.4 Complete Redundancy NCS Detection

A *Complete Redundancy NCS* is detected when two adjacent *Context Agents* with similar models can merge their common boundary without altering their other ranges. The *model*

similarity distance d^{f}_{sim} is used as in the *Concurrency NCS* detection. An ideal *Complete Redundancy NCS* is defined by $n-1$ equal *validity ranges* ($r_i^{j_1} = r_i^{j_2}$) and a meeting intersection ($r_i^{j_1}$ m $r_i^{j_2}$) (Fig. 5.9a). In a continuous space, it is almost impossible to match these conditions. So the *range similarity distance* $d_i^{\mathcal{R}_{sim}}$ is used as a threshold when establishing the meeting condition. In this case, two *Context Agents* are adjacent if the distances between their boundaries are less than $d_i^{\mathcal{R}_{sim}}$. The *range similarity distance* $d_i^{\mathcal{R}_{sim}}$ is also used for the alignment of the other *validity ranges* (Fig. 5.9b).



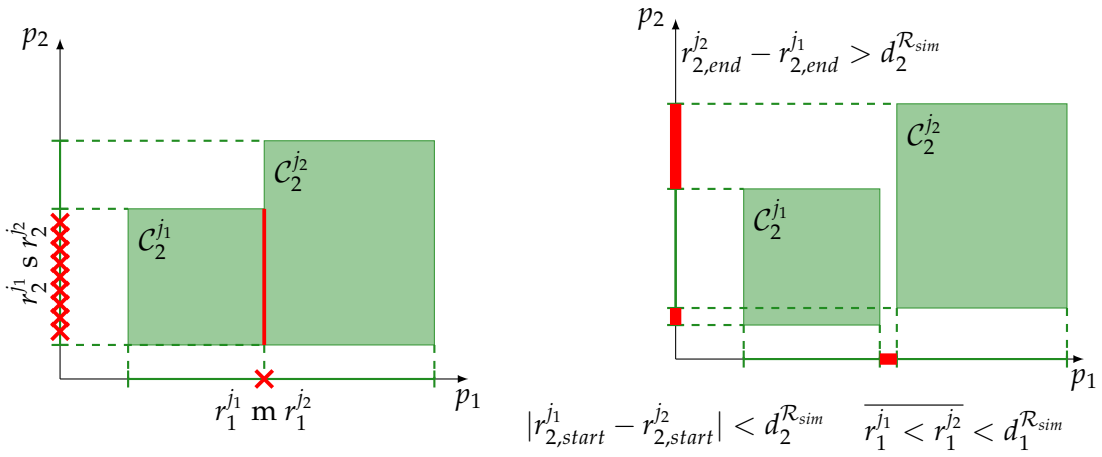
(a) Theoretical

(b) Practical

Figure 5.9 – *Complete Redundancy NCS* Detection.

5.4.2.5 Partial Redundancy NCS Detection

A *Partial Redundancy NCS* is when two adjacent *Context Agents* have similar models and can restructure one of their *validity ranges* to maximize the volume of one of the *Context Agents*. The model similarity is also detected using the *model similarity distance* d^{f}_{sim}. An ideal *Partial Redundancy NCS* is defined by $n-2$ equal ranges ($r_i^{j_1} = r_i^{j_2}$), a starting ($r_i^{j_1}$ s $r_i^{j_2}$) or a finishing ($r_i^{j_1}$ f $r_i^{j_2}$) intersection and a meeting intersection ($r_i^{j_1}$ m $r_i^{j_2}$) (Fig 5.10a).



(a) Theoretical

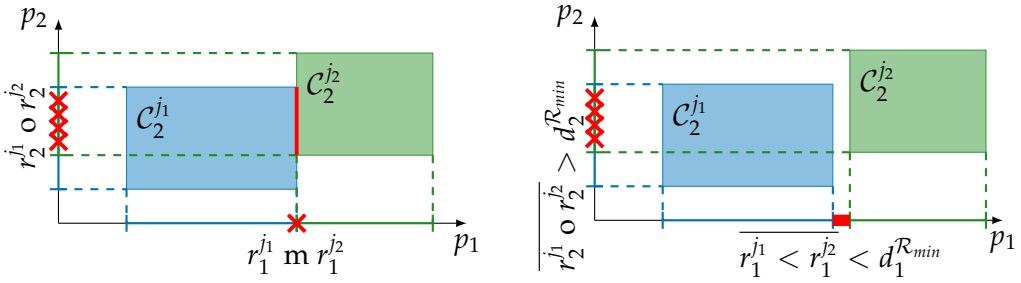
(b) Practical

Figure 5.10 – *Partial Redundancy NCS* Detection.

By representing two *perceptions*, only the two last conditions can be present in the figure 5.10. As in the previous *NCS*, the *range similarity distance* $d_i^{\mathcal{R}_{sim}}$ is used. Two agents are therefore adjacent if the distance between one their boundaries are less than the *range similarity distance* $d_i^{\mathcal{R}_{sim}}$. A *validity range* starts or finishes another one if respectively $|r^{j_1}_{2,start} - r^{j_2}_{2,start}| < d_i^{\mathcal{R}_{sim}}$ and $r^{j_2}_{2,end} - r^{j_1}_{2,end} > d_i^{\mathcal{R}_{sim}}$ or $r^{j_1}_{2,start} - r^{j_2}_{2,start} > d_i^{\mathcal{R}_{sim}}$ and $|r^{j_1}_{2,end} - r^{j_2}_{2,end}| < d_i^{\mathcal{R}_{sim}}$. The figure 5.10b shows an example of practical *Partial Redundancy NCS*.

5.4.2.6 Range Ambiguity NCS Detection

A *Range Ambiguity NCS* is a difference of model between two adjacent *Context Agents*. Model similarity is once again computed with the *model similarity distance* d^f_{sim}. An ideal *Range Ambiguity NCS* is defined by $n-1$ non-empty intersections (overlaps, beginnings, ends, inclusions and equalities) and a meeting intersection (Fig. 5.11a). In practice, for two agents to be adjacent according to the perception p_i, the distance between their boundaries must be less than the *minimum range distance* $d_i^{\mathcal{R}_{min}}$ and the distance of the non-empty intersection must be greater than the *minimum range distance* $d_i^{\mathcal{R}_{min}}$ (Fig. 5.11b).

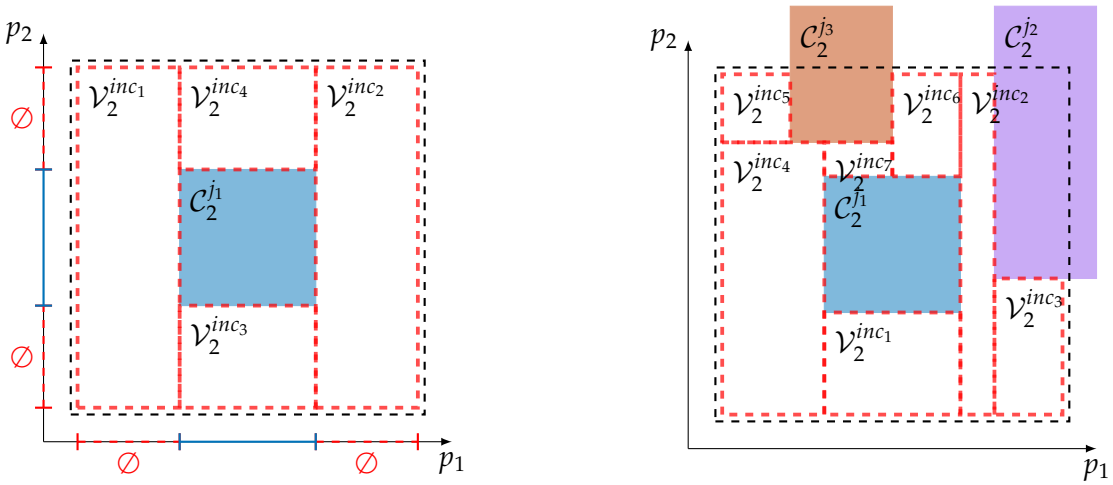


(a) Theoretical (b) Practical

Figure 5.11 – *Range Ambiguity NCS* Detection.

5.4.2.7 Incompetence NCS Detection

An *Incompetence NCS* is an emptiness inside the *neighborhood* of the current *perceptions*. For a gap to be retained, the minimal length of the ranges that defines it must be greater than the *minimum range distance* $d_i^{\mathcal{R}_{min}}$. The *Incompetence NCS* is used to detect empty areas alias *incompetent volumes* $\mathcal{V}_n^{inc}$ within the *neighborhood* of the current *perceptions*. An area is empty if it has an empty intersection $\emptyset$ with the *validity range* of a *Context Agent* on at least one perception p_i. As few *incompetent volumes* as possible are constructed. At 2 dimensions and with only one *Context Agent* in the middle of the *neighborhood*, the minimum number of *incompetent volumes* to be filled is 4 (Fig. 5.20a). This strategy for seeking incompetent zones may be computationally expensive for high numbers of *perceptions* and large quantities of *Neighbor Context Agents*. This matter will be addressed in the *Scalability* experiments in section 7.8. The figure 5.12b shows an example of *incompetent volume* detection with several *Context Agents* in a *neighborhood*.



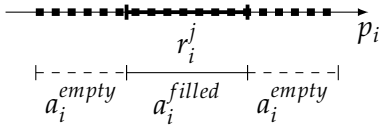
(a) *Incompetence NCS* with one *Neighbor Context Agent*

(b) *Incompetence NCS* with several *Neighbor Context Agents*

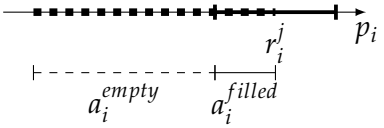
Figure 5.12 – *Incompetence NCS* Detection.

Recursive Incompetent Volume Generation: *Incompetent volumes* are computed across all *Neighbor Context Agents*. Initially, there is one *empty area* : the *neighborhood* area (Fig. 5.14a). The *neighborhood* area is compared to the *validity ranges* of the first *Neighbor Context Agent*. From this comparison, the initial empty area is deleted and new ones are created (Figures 5.14b, 5.14c and 5.14d). The new empty areas are then compared to the next *Neighbor Context Agent* and so on until all *Neighbor Context Agents* are processed. Once all *Neighbor Context Agents* are processed, the left empty areas are defined as *incompetent volumes*. Figure 5.14d shows an example of completed recursive *incompetent volume* generation with only one *Neighbor Context Agent*.

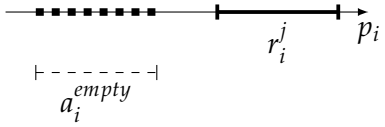
When a *Context Agent* is compared to an *empty area* A_n^{empty}, each perception p_i is recursively computed to obtain *one-dimensional filled areas* a_i^{filled} and *one-dimensional empty areas* a_i^{empty} (Fig. 5.13). If they are *one-dimensional empty areas*, they are completed with the other dimensions of the initial *empty area* A_n^{empty} to create a definitive *incompetent volume* $\mathcal{V}_n^{inc}$. This corresponds to the *incompetent volumes* $\mathcal{V}_2^{inc_1}$ and $\mathcal{V}_2^{inc_2}$ in the figure 5.14b. If there is a *one-dimensional filled area*, the area is completed with the other dimensions of the initial *empty area* A_n^{empty} to create a new *empty area* $A_n^{empty'}$. This corresponds to the *empty area* $A_2^{empty'}$ in the figure 5.14b. The current perception that gave the *one-dimensional filled area* is considered computed and the *Context Agent validity ranges* are then compared to $A_n^{empty'}$ across the remaining perceptions as is has been done with A_n^{empty} (Fig. 5.14c). Figure 5.14d shows the completed search of *incompetent volumes* for two perceptions and one *Neighbor Context Agent* in the center of the *neighborhood*.



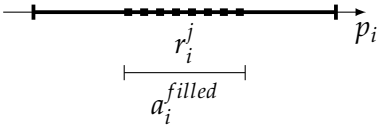
(a) Creation of two *empty areas* and one *filled area*



(b) Creation of one *empty area* and one *filled area*

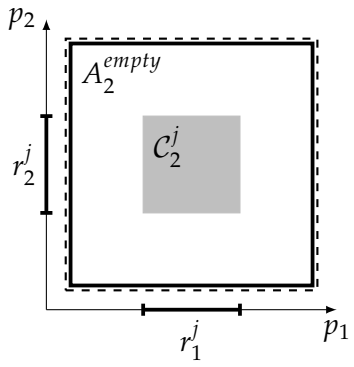


(c) Creation of one *empty area*

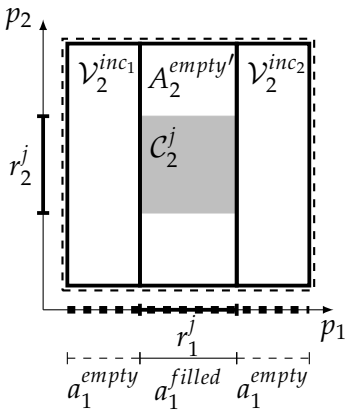


(d) Creation of one *filled area*

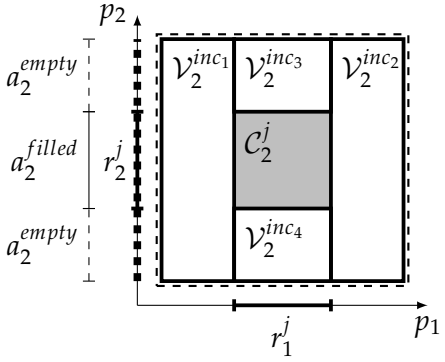
Figure 5.13 – Possible configurations when comparing one-dimensional *empty areas* (dotted line) with one-dimensional *validity ranges*.



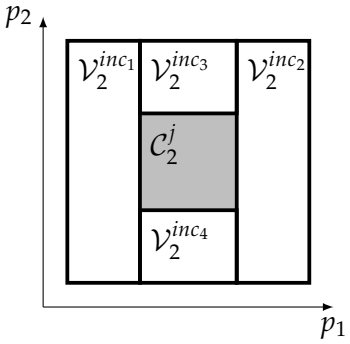
(a) The *neighborhood* is A_2^{empty} and corresponds to the initial *empty area*.



(b) A_2^{empty} and $\mathcal{C}_2^j$ are compared on p_1. It gives two *incompetent volumes* $\mathcal{V}_2^{inc_1}$ and $\mathcal{V}_2^{inc_2}$; and one *empty area* $A_2^{empty'}$.



(c) $A_2^{empty'}$ and $\mathcal{C}_2^j$ are compared on p_2. It gives two additional *incompetent volumes* $\mathcal{V}_2^{inc_3}$ and $\mathcal{V}_2^{inc_4}$.



(d) All perception have been processed. Four *incompetent volumes* have been created.

Figure 5.14 – Recursive *incompetent volume* generation for two perceptions and one *Neighbor Context Agent*.

5.4.3 Model Inaccuracies Detection

In this section, the characterization of model *learning inaccuracies* and their detection are presented. The mechanisms are independent from the underlying models. For presentation purposes, linear regressions are used.

5.4.3.1 Model Ambiguity NCS Detection

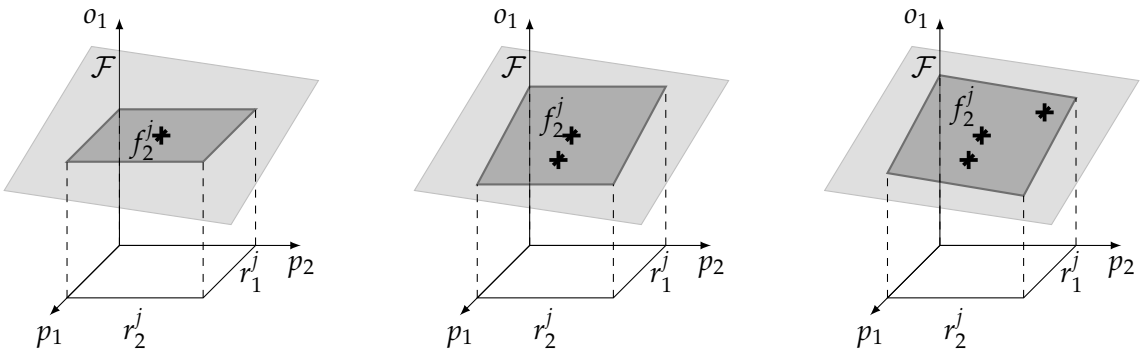
Learning models in *Machine Learning* can require certain quantities of labeled data to provide reliable predictions. When the volume or quality of data is not adequate for the learning, this *NCS* is detected. A *Model Ambiguity NCS* is when a model needs additional *learning situations* to consider itself mature or reliable. A model without enough *learning situations* is thus called a immature model or an unreliable model.

In the case of linear regression with n perceptions and one prediction, $n+1$ coefficients are needed to complete the linear model. Thus $n+1$ *learning situations* are needed to fulfill the model needs. The figure 5.15 shows an example of linear regression for one perception. To match the *hidden function*, at least two *learning situations* are needed. The figure 5.16 shows an example of linear regression with two perceptions. To match the *hidden function*, at least three *learning situations* are needed.



(a) Immature model with 1 *learning situation* (b) Mature model with 2 *learning situations*

Figure 5.15 – *Model Ambiguity NCS* detection for 1 perception.



(a) Immature model with 1 *learning situation*

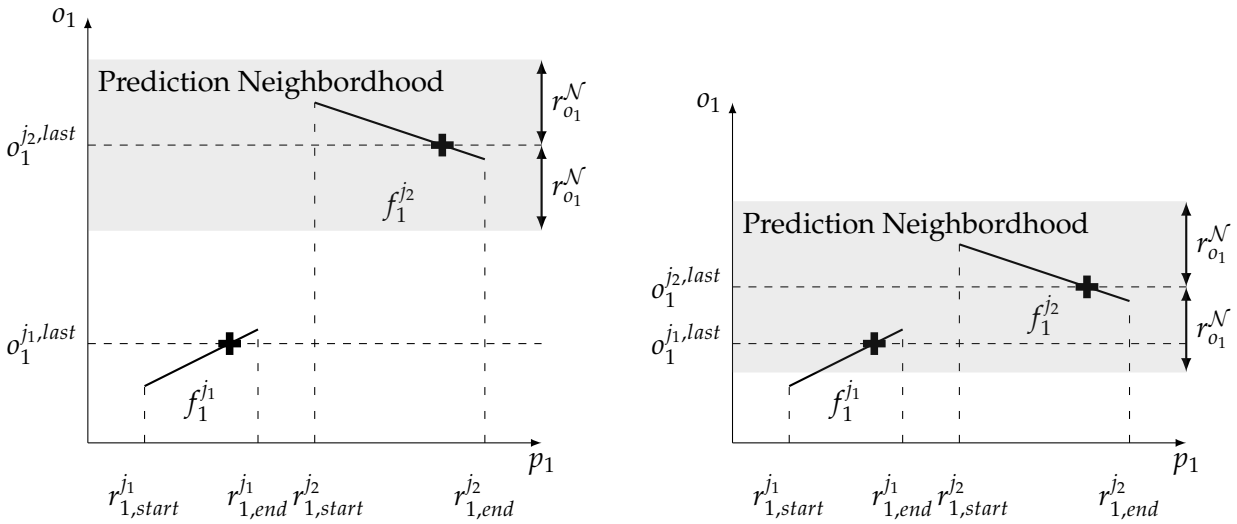
(b) Immature model with 2 *learning situations*

(c) Mature model with 3 *learning situations*

Figure 5.16 – *Model Ambiguity NCS* Detection for 2 perceptions.

5.4.3.2 Model Discontinuity NCS Detection

As said in sections 5.1 and 5.2, the *Continuity Hypothesis* states that *Context Agents* must differentiate continuity and discontinuity. A *Model Discontinuity NCS* is detected when two *Context Agents* are *Neighbor Context Agents* but not *Prediction Neighbor Context Agents*. The distance between their last predictions is greater than the *prediction neighborhood radius* $r^{\mathcal{N}}_{o_k}$. Otherwise, among the *Prediction Neighbor Context Agents*, continuity is assumed. Continuity is when the distance between the last predictions of two *Context Agents* among the *Neighbor Context Agents* is less than the *prediction neighborhood radius* $r^{\mathcal{N}}_{o_k}$. Figure 5.17 shows the cases of assumed discontinuity and continuity.



(a) Discontinuity is assumed

(b) Continuity is assumed

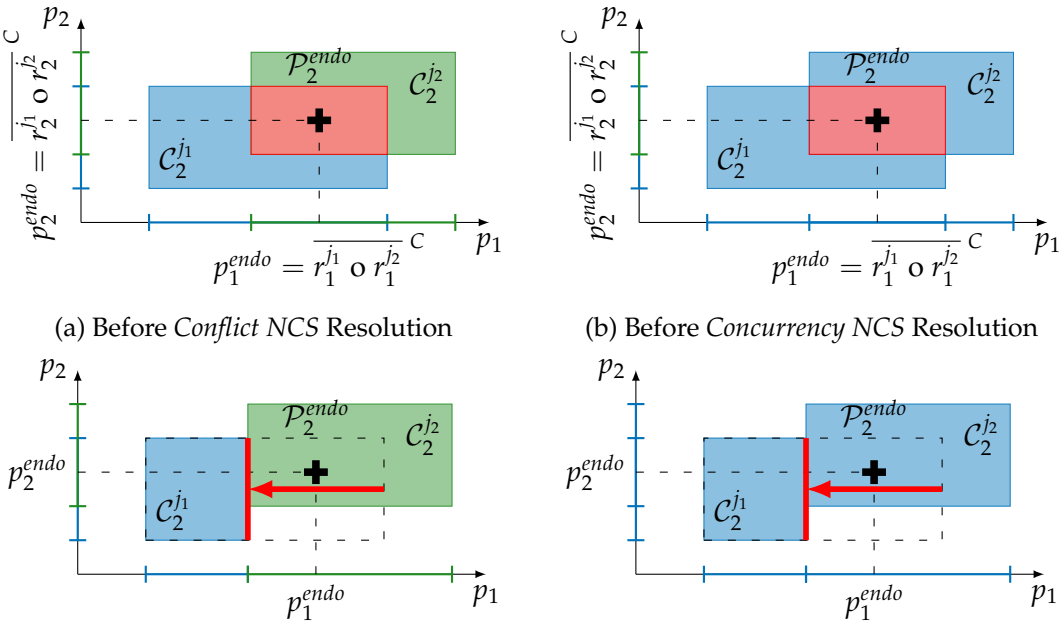
Figure 5.17 – Continuity and Discontinuity among *Neighbor Context Agents*.

5.4.4 Exploration Inaccuracies Resolution

I present here the resolution of the *learning inaccuracies*. Most of the *learning inaccuracies* resolutions are started with the generation of *endogenous perceptions* $\mathcal{P}_n^{endo}$ that situate the *learning inaccuracies*. The resolutions that require *endogenous perceptions* are presented as if they were instantly processed. In practice, it will be presented in the section 5.5 that the *learning inaccuracies* associated with *endogenous perceptions* are stored to be solved during a future cycle. The nature of this future cycle will depend on the learning strategy that is used: the *Active Learning Strategy* or the *Self-Learning Strategy*. These strategies will be presented in section 6.4. Their use depends on the availability of additional specific *learning situations*. The resolutions that do not require any *endogenous perceptions* can be processed instantaneously. Each *NCS* is then treated differently.

5.4.4.1 Conflict and Concurrency NCS Resolution

For the *Conflict NCS* and *Concurrency NCS*, the *endogenous perceptions* are generated in the middle of the overlapping volume $p_i^{endo} = \overline{r_i^{j_1} \text{ o } r_i^{j_2}}^{C}$. Depending on the learning strategies that will be presented section 6.4, the two *Context Agents* negotiate which one should retract itself to suppress the overlapping area. The looser of the negotiation updates one of its *validity range* in order to eliminate the overlap. The *validity range* that is chosen is the one that least affects the volume of the shrunk *Context Agent*. In other words, it is the *validity range* that minimizes the volume loss while resolving the overlap. The figures 5.18a and 5.18b show the position of the *endogenous perceptions*. The figures 5.18c and 5.18d represent examples of resolutions where $\mathcal{C}_2^{j_1}$ is the *Context Agent* that retracts itself and p_1 is the optimal perception for the resolution.



(a) Before *Conflict NCS* Resolution

(b) Before *Concurrency NCS* Resolution

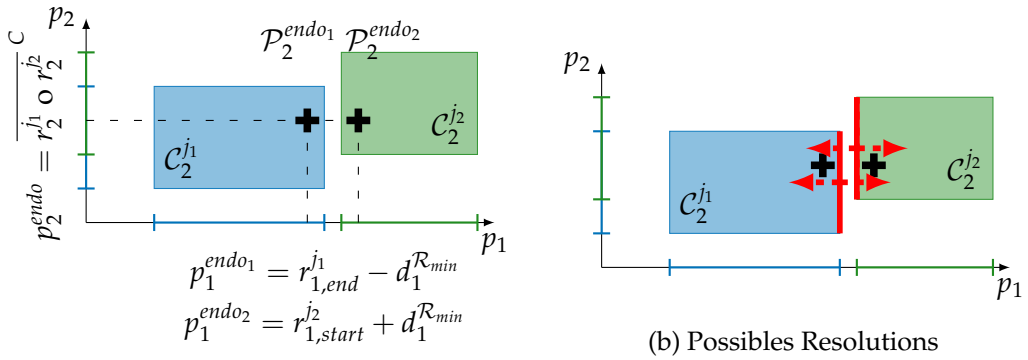
(c) Resolution Example After *Conflict NCS*

(d) Resolution Example After *Concurrency NCS*

Figure 5.18 – *Conflict NCS* and *Concurrency NCS* Resolutions.

5.4.4.2 Range Ambiguity and Model Discontinuity NCS Resolution

Range Ambiguity NCS and *Model Discontinuity NCS* are coupled *NCS* because a discontinuity necessarily concerns two *Context Agents* that are adjacent and that possess different models according to the *model similarity distance* d^{f}_{sim}. If both *NCS* are detected between two *Context Agents*, the frontier between them can be clarified. To do so, two *endogenous perceptions* $\mathcal{P}_i^{endo_1}$ and $\mathcal{P}_i^{endo_2}$ are generated. For the adjacent perception, $p_i^{endo_1}$ and $p_i^{endo_2}$ are at a distance of $d_i^{\mathcal{R}min}$ of the adjacent border $r^j_{i,end} - d_i^{\mathcal{R}min}$ or $r^j_{i,start} + d_i^{\mathcal{R}min}$. The adjacent perception is p_1 in the figure 5.19a. For all the other perceptions, $p_i^{endo_1}$ and $p_i^{endo_2}$ are both at the center of the overlapping ranges $\overline{r_i^j \text{ o } r_i^j}^{\,C}$. In the figure 5.19a the other perception is only p_2. Figure 5.19b shows the possibles future modifications of the considered ambiguous *Context Agents* borders. A border may not change if it is correct. As many of these situations can occur and to avoid targeting infinitely the same discontinuity, the user parameter *discontinuity detection probability* pb^{disc} is used. It enables to select the importance of discontinuities detections.



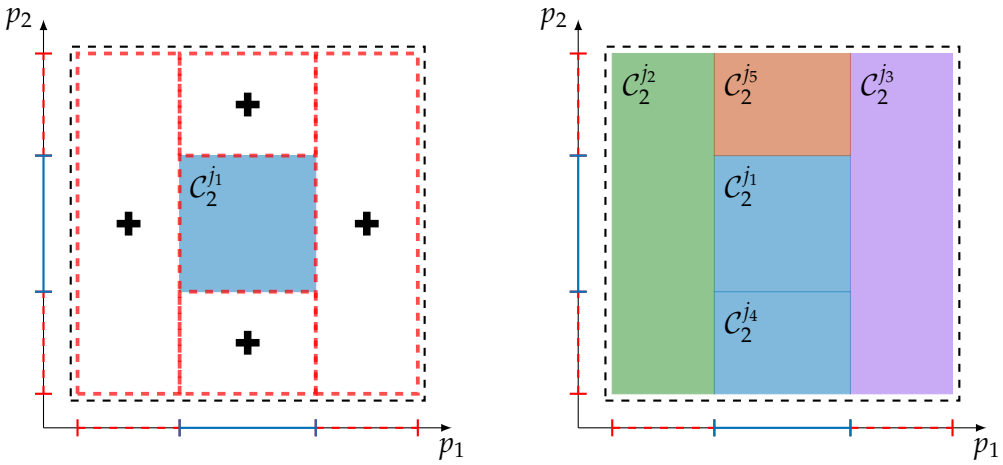
(a) Before Resolution

(b) Possibles Resolutions

Figure 5.19 – *Range Ambiguity NCS* and *Model Discontinuity NCS* Resolution.

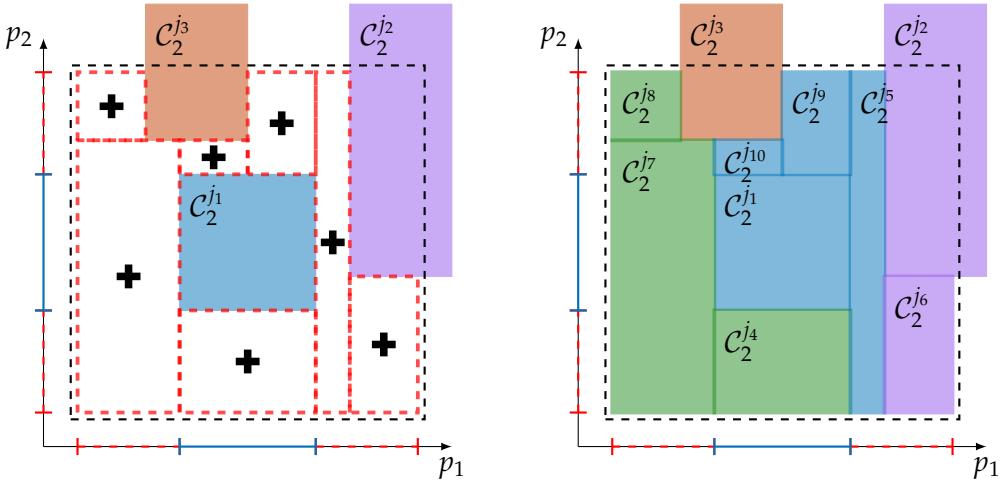
5.4.4.3 Incompetence NCS Resolution

For the *Incompetence NCS*, *endogenous perceptions* are generated in the center of the *incompetent volumes* introduced during the detection of the *Incompetence NCS* (section 5.4.2.7). Each *endogenous perceptions* associated to their *incompetent volume* will define a different *learning inaccuracy* that will need a different cycle to be processed. When the *endogenous perceptions* are used to create a new *Context Agent*, the dimensions of the *incompetent volume* are used to initialize the *validity ranges* of the created *Context Agent*. Figure 5.20 shows the creation of *Context Agents* in a partially empty *neighborhood* with one *Neighbor Context Agent* and with several *Neighbor Context Agents*.



(a) Before resolution with one *Neighbor Context Agent*

(b) After resolution with one *Neighbor Context Agent*



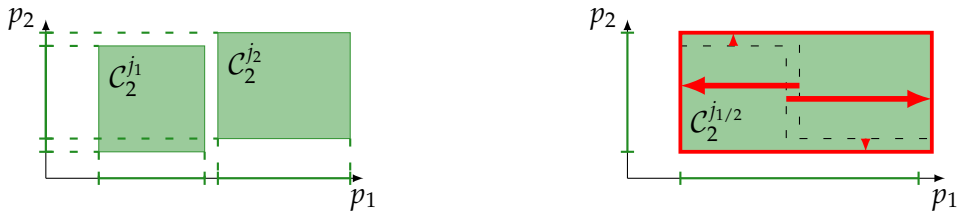
(c) Before resolution with several *Neighbor Context Agents*

(d) After resolution with several *Neighbor Context Agents*

Figure 5.20 – *Incompetence NCS* Resolution.

5.4.4.4 Complete Redundancy NCS Resolution

The *Complete Redundancy NCS* resolution is made instantaneously as no additional information is needed and the generalization of two *Context Agents* into one is possible. One of the two *Context Agents* extends its *validity ranges* to the maximum of both *Context Agents validity ranges* to cover the two. The updated *Context Agent* sets its *confidence* to the sum of both *Context Agents* confidences. The other *Context Agent* that didn't make any adaptation is subtracted from the collective. Figure 5.21 represents a *Complete Redundancy NCS* resolution with two *perceptions*.



(a) Before Resolution (b) After Resolution

Figure 5.21 – *Complete Redundancy NCS* Resolution.

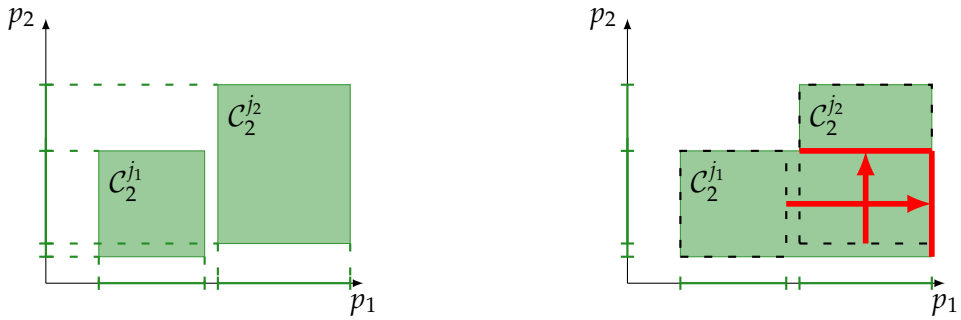
5.4.4.5 Partial Redundancy NCS Resolution

The *Partial Redundancy NCS* resolution is also made instantaneously. The *Context Agent* that can maximize its volume extends its *validity range* on the adjacent perception. It is the growing *Context Agent*. In the figure 5.22b, the adjacent perception is p_1 and the growing *Context Agent* is $\mathcal{C}_2^{j_1}$. The other *Context Agent* updates the *validity range* on the perception that is finishing or starting one of the *validity ranges* of the growing *Context Agent*. It is the shrinking *Context Agent*.

On the figure 5.22b, the shrinking perception is p_2 because $r_{2,start}^{j_1}$ is similar to $r_{2,start}^{j_2}$ The shrinking *Context Agent* is $\mathcal{C}_2^{j_2}$. The *confidence* of the shrinking *Context Agent* is shared between itself and the growing *Context Agent*. Still using the example of the figure 5.22b, let c^{j_1}, c^{j_2}, $\mathcal{V}_2^{j_1}$ and $\mathcal{V}_2^{j_2}$ the *confidences* and volumes of the *Context Agents* $\mathcal{C}_2^{j_1}$ and $\mathcal{C}_2^{j_2}$ before the resolutions. After the resolution, the calculation of the new *confidences* $c_{new}^{j_1}$, $c_{new}^{j_2}$ is done as follows.

$$\begin{aligned} c_{new}^{j_1} &= c^{j_1} + c^{j_2}.\left(1 - \frac{\mathcal{V}_{2,new}^{j_2}}{\mathcal{V}_2^{j_2}}\right) \\ c_{new}^{j_2} &= c^{j_2}.\frac{\mathcal{V}_{2,new}^{j_2}}{\mathcal{V}_2^{j_2}} \end{aligned} \tag{5.9}$$

With $\mathcal{V}_{2,new}^{j_2}$ the new volume of the shrunk *Context Agent*. The *confidence* of the *Context Agent* losing volume is given to the other *Context Agent* in proportion to the volume transferred.



(a) Before Resolution (b) After Resolution

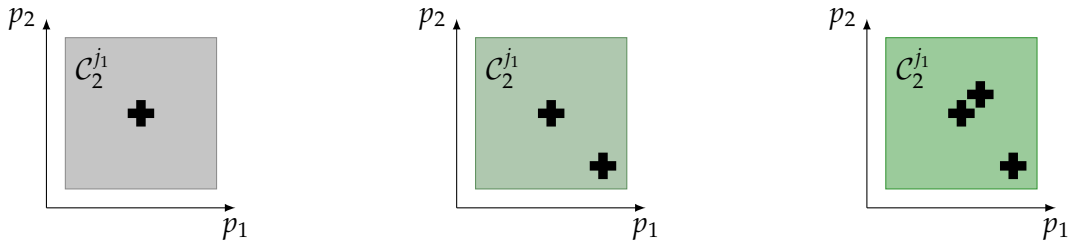
Figure 5.22 – *Partial Redundancy NCS* Resolution.

5.4.5 Model Inaccuracies Resolution

The only remaining model inaccuracy is the *Model Ambiguity NCS* as the *Model Discontinuity NCS* is merged with the *Range Ambiguity NCS*.

5.4.5.1 Model Ambiguity NCS Resolution

As we have seen, a model can require additional *learning situations*. The *Model Ambiguity NCS* resolution is made by generating *endogenous perceptions* where new *learning situations* will be integrated in future cycles. Each *endogenous perceptions* are randomly created uniformly inside the *validity ranges* of the considered *Context Agent*. Figure 5.23 shows an example of *Model Ambiguity NCS* resolution. Each new random *endogenous perceptions* transformed into a *learning situation* enables the *Context Agent* to converge towards a complete or mature model. Maturity being represented by the shade of color.



(a) Immature *Context Agent* with 1 *learning situation*

(b) Immature *Context Agent* with 2 *learning situations*

(c) Mature *Context Agent* with 3 *learning situations*

Figure 5.23 – *Model Ambiguity NCS* Resolution for 2 perceptions.

5.5 Learning Inaccuracies Computation

Now that all the *learning inaccuracies* have been presented by defining them as *NCS*, this section will detail the order in which they are processed. Except for the *Complete Redundancy NCS* and the *Partial Redundancy NCS* which trigger an instantaneous *validity ranges* adaptations, the *NCS* detection are stacked in a pile waiting to be called for their resolutions.

Once detected, *learning inaccuracies* (other than *Complete Redundancy NCS* and *Partial Redundancy NCS*) are associated with a certain priority to be addressed. The order of priority from the highest to the lowest is given in the table 5.4.

Model Ambiguity NCS
Conflict NCS
Concurrency NCS
Incompetence NCS
Range Ambiguity NCS

Table 5.4 – *NCS* priorities from the highest to the lowest.

The most important goal of a *Context Agent* is to complete its model because the rest of the *learning inaccuracies* rely on the completeness of the models. Then, the most critical *learning inaccuracies* is the *Conflict NCS* because, in addition to bad *validity range*, it can lead to bad prediction. After that, there is the *Concurrency NCS*, whose manifestation is the same but without the risk of a bad prediction. When there are not any overlaps between the *Context Agents*, the priority is to fill in the gaps and seek for unexplored areas by detecting *Incompetence NCS*. Finally, the last possible improvement for a complete exploration is to refine the discontinuous boundaries with the *Range Ambiguity NCS*.

Each *learning inaccuracy* $\mathcal{I}_n^l$ is defined by $\{\mathcal{R}_n^l, \mathcal{P}_n^{endo}, \mathcal{M}^l, \mathcal{T}^l, p\}$, with:

- **Identification,** l: It is the concatenation of the *Context Agents* names that raised the *learning inaccuracies*. For *Incompetence NCS*, all the *Neighbor Context Agents* are involved in the detection of the *learning inaccuracy*. For all other *NCS*, only two *Context Agents* are concerned.
- **Ranges,** $\mathcal{R}_n^l$: they define the area of the overlap in the case of a *Conflict NCS* or *Concurrency NCS*. In the case of an *Incompetence NCS*, the ranges define the *incompetent volume* that will be used for the creation of the *Context Agent*.
- **Endogenous perceptions,** $\mathcal{P}_n^{endo}$: it is the vector of *perceptions* that is generated in order to solve the *learning inaccuracy*.
- **Memory,** $\mathcal{M}^l$: it is the list of *Context Agents* that detected the *learning inaccuracy*. This list is used to ensure that two *Context Agents* don't detect the same *NCS* twice.
- **Type,** $\mathcal{T}^l$: it is the type of *learning inaccuracy* (*Model Ambiguity NCS*, *Conflict NCS*, *Concurrency NCS*, *Incompetence NCS* or *Range Ambiguity NCS*).
- **Priority,** p: it determines the urgency with which the inaccuracy should be addressed.

To ensure that *Incompetence NCS* are not detected several times, the stack of *learning inaccuracies* must be emptied before storing new ones. Thus, an empty *neighborhood* is fully explored before seeking more *Incompetence NCS*.

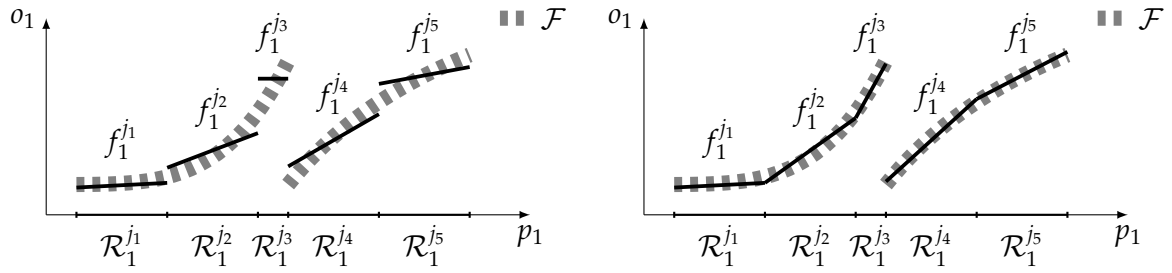
As for *Incompetence NCS*, *Range Ambiguity NCS* being the last learning improvement, the stack of *learning inaccuracies* must be empty before adding any *Range Ambiguity NCS*. This ensures that *Range Ambiguity NCS* does not impinge on the space exploration.

5.6 Cooperative Neighborhood Learning

As we have seen, according to the *Continuity Hypothesis*, in this context, learning is seeking continuity and discontinuity. In this section, a new mechanism is presented to satisfy the *Continuity Hypothesis* with the objective of reducing the amount of *learning situations* from an oracle: *Cooperative Neighborhood Learning*.

The aim of this mechanism is to enhance the *Context Agents local models* by internally generating new *learning situations*: *endogenous learning situations*. *Cooperative Neighborhood Learning* enables *Endogenous Learning*. As we have seen, *Endogenous Learning* is when a learning mechanism teaches itself *learning situations*.

We have seen that *Context Agents* locally map a *hidden function* with a *local model*. As continuity is assumed among all *Prediction Neighbor Context Agents*, *Context Agents* should seek to match their border predictions with their neighbors. This can be seen as a smoothing of the *Context Agents* models between them. We call this mechanism *Cooperative Neighborhood Learning (CNL)*. Figure 5.24a shows an example of a learning state where the models have not matched the *hidden function* and are not meeting the prediction of their neighbors. Figure 5.24b shows the objectives of *CNL* which are to smooth the models but using only *Endogenous Learning*.



(a) Before learning with *Prediction Neighbor Context Agents*

(b) After learning with *Prediction Neighbor Context Agents*

Figure 5.24 – Schema of *Cooperative Neighborhood Learning* aims. The search space is supposed entirely explored and represented by *Context Agents validity ranges*. The gray thick dashes represent the *hidden function* $\mathcal{F}$, and the black lines are the learned *local models* $f_1^{j_i}$.

Only one *local model* is enhanced by learning or exploitation cycle (learning and exploitation cycles will be detailed section 6.1). It is the model of the *Best Context Agent*. It is reminded that the *Best Context Agent* is the *Context Agent* for which its *local model* is considered the best during a learning or an exploitation cycle. A *Best Context Agent* communicates with its *Prediction Neighbor Context Agents* to ask them for *endogenous learning situations*. An *endogenous learning situation* is defined by *endogenous perceptions* $\mathcal{P}_n^{endo}$ and an *endogenous prediction vector* $\mathcal{O}_m^{endo}$ such as $\mathcal{L}_{n,m}^{endo} = [\mathcal{P}_n^{endo}, \mathcal{O}_m^{endo}]$.

The *endogenous perceptions* $\mathcal{P}_n^{endo}$ of the *endogenous learning situations* are chosen randomly in the intersection of the *neighborhood* and the neighbor's *validity ranges*. For influential *Neighbor Context Agents*, *endogenous learning situations* are chosen randomly in the intersection of the *neighborhood* and the *influence* zone of the *Context Agents* (fig. 5.25).

The *endogenous prediction vector* $\mathcal{O}_m^{endo}$ is asked to the model of the neighbors. As only *Prediction Neighbor Context Agents* are considered, the endogenous predictions are in the *prediction neighborhood*. *Neighbor Context Agents* that are not in the *prediction neighborhood* are not allowed to share *endogenous prediction vectors*. They represent a discontinuity with the *Best Context Agent* because they are in the spatial *neighborhoods* but not in the *prediction neighborhood*.

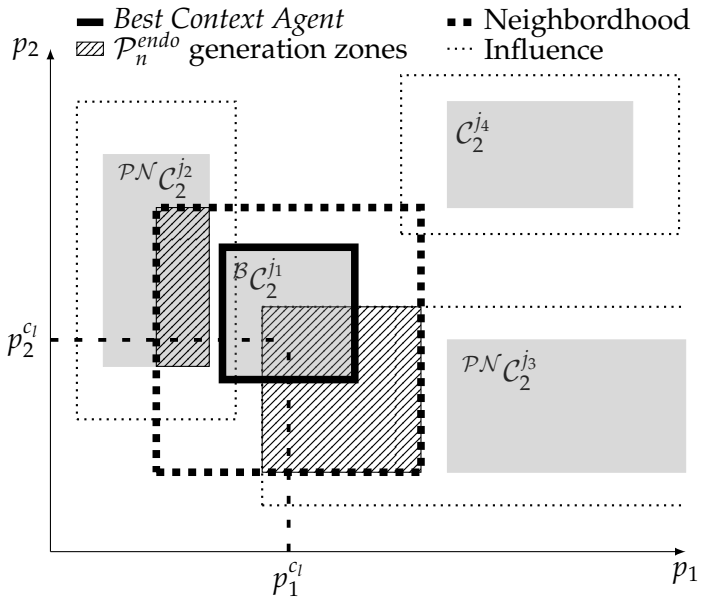


Figure 5.25 – *Cooperative Neighborhood Learning*. The *endogenous learning situations* are generated in the hatched zones. The *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_2^{j_1}$ is receiving the *endogenous learning situations*. The *Prediction Neighbor Context Agents* ${}^{\mathcal{PN}}\mathcal{C}_2^{j_2}$ and ${}^{\mathcal{PN}}\mathcal{C}_2^{j_3}$ are each providing an *endogenous learning situation*.

The set of all retained *endogenous learning situations* is used to update the *Best Context Agent* model with a weight w_{lrn}^{endo} called the *endogenous learning weight*. To satisfy this weight, artificial *learning situations* are generated for a lifelong update of *Context Agents local models*. They are distributed on the *Best Context Agent local model* according to a normal law centered in its *validity ranges* center. The normal law standard deviation is set to ensure that the artificial *learning situations* are contained close to the center of the *local model*. Section 6.3 will further detail the implemented mechanisms for updating *local models* in a *Lifelong Learning* setting.

The *endogenous learning weight* w_{lrn}^{endo} is a designer parameter to set the importance of *endogenous learning situations* (Table 5.5).

	Name	Notation	Constrains	Domain
Designer Parameter	*endogenous learning weight*	w_{lrn}^{endo}	$]0,1[$	$\mathbb{R}$

Table 5.5 – Designer parameter for the *Cooperative Neighborhood Learning* mechanism.

5.7 Synthesis

In this chapter, we presented the learning hypotheses upon which relies the introduction of *learning inaccuracies*. In order to detect them, the concept of *neighborhood* was added to the *Context Learning* paradigm. The characterization and the resolution of the *learning inaccuracies* were detailed. A prioritized treatment strategy was developed. Finally, *Cooperative Neighborhood Learning* was defined as a mechanism that self-generates new *learning situations* that are *endogenous learning situations*. It enables to enhance the learning models by locally sharing predictions. Table 5.6 provides a summary of the parameters involved in the presented mechanisms.

	Name	Notation	Constrains	Domain
User Parameters	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$
	endogenous learning weight	w_{lrn}^{endo}	$]0;1]$	$\mathbb{R}$
	discontinuity detection probability	pb^{disc}	$]0;1[$	$\mathbb{R}$
User Distance	*range creation radius*	$r_i^{creation}$	$(p^{\mathcal{R}})$-dependent	$\mathbb{R}$
User Metric	*model similarity distance*	d_{sim}^{f}	model-dependent	$\mathbb{R}$
Designer Paramaters	*neighborhood radius coefficient*	$\alpha^{\mathcal{N}}$	> 0	$\mathbb{R}$
	influence radius coefficient	$\alpha^{\mathcal{I}}$	> 0	$\mathbb{R}$
	range similarity coefficient	$\alpha^{\mathcal{R}_{sim}}$	$]0,1[$	$\mathbb{R}$
	minimum range coefficient	$\alpha^{\mathcal{R}_{min}}$	$]0,1[, \alpha^{\mathcal{R}_{min}} < \alpha^{\mathcal{R}_{sim}}$	$\mathbb{R}$
Designer Distances	*neighborhood radius*	$r_i^{\mathcal{N}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{N}})$-dependent	$\mathbb{R}$
	prediction neighborhood radius	$r_{o_k}^{\mathcal{N}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{N}})$-dependent	$\mathbb{R}$
	influence radius	$r_{j,i}^{\mathcal{I}}$	$(\alpha^{\mathcal{I}})$-dependent	$\mathbb{R}$
	range similarity distance	$d_i^{\mathcal{R}_{sim}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{R}_{sim}})$-dependent	$\mathbb{R}$
	minimum range distance	$d_i^{\mathcal{R}_{min}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{R}_{min}})$-dependent	$\mathbb{R}$

Table 5.6 – Table of user and designer parameters, distances and metrics for the *Endogenous Context Learning* principles.

The next chapter will focus on the update of *Context Learning* rules taking advantage of the additional information that the *neighborhood* brings. Two learning strategies that differ in the availability of *learning situations* will be proposed and other mechanisms involved in *Endogenous Context Learning* will be detailed.

6 Endogenous Lifelong Learner by Self-Adaptation ELLSA

In this chapter, the functioning of the designed learning mechanism is presented: ELLSA for Endogenous Lifelong Learner by Self-Adaptation. The optimization of agents activation is detailed and the processes of learning and exploitation cycles are described. The learning criticality and the exploitation criticality are introduced, they enable to include performance, generalization and experience in the learning and exploitation processes. A mechanism for dynamically updating learning models in a lifelong setting is proposed. The Active Learning Strategy and the Self-Learning Strategy are introduced. Their use depends on the availability of learning data. The chapter is ended by an analysis and a positioning about AI challenges.

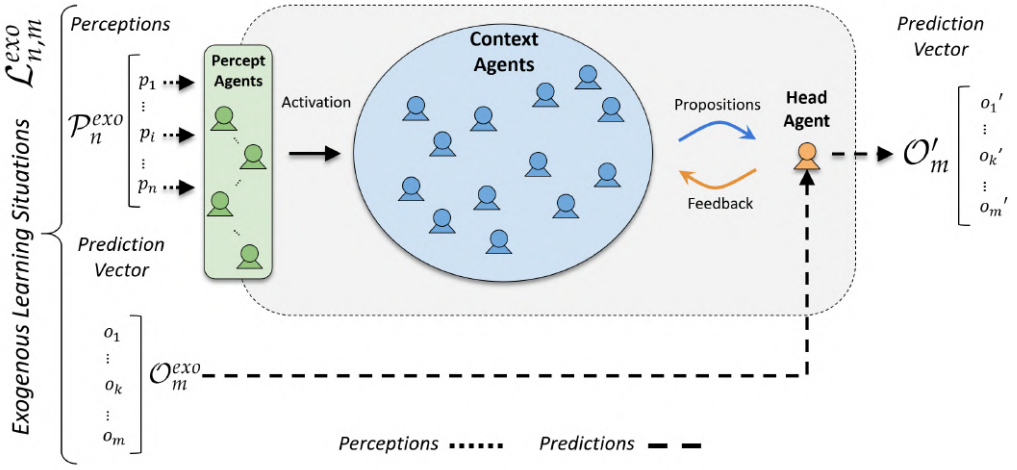
THE previous chapter focused on the *Context Agents* mechanisms and how their communication can enhance their collective representation of learning. We introduced the concept of *Endogenous Learning* by presenting the detection of *learning inaccuracies* and their resolutions using *endogenous perceptions*. Finally, a way of generating complete *endogenous learning situations* was detailed.

However, in *Context Learning*, other agents intervene in the process of learning. They define a larger Multi-Agent System (*MAS*) that is called *Endogenous Lifelong Learner by Self-Adaptation (ELLSA)*. Based on the learning mechanism *AMOEBA* [Nigon, 2017], figure 6.1a shows a global view of the agents involved in classical *Context Learning*. As presented in section 4.4, the learning is made with *learning situations* that are *exogenous learning situations*. The *exogenous learning situations* $\mathcal{L}^{exo}_{n,m}$ are composed of *exogenous perceptions* $\mathcal{P}^{exo}_{n}$ and an *exogenous prediction vector* $\mathcal{O}^{exo}_{m}$ provided by an external entity called the oracle.

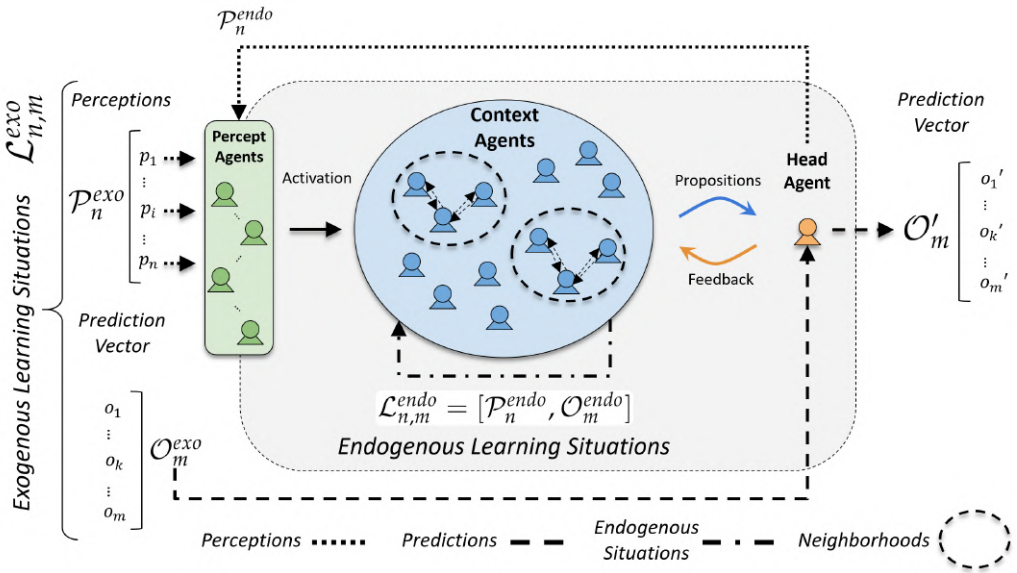
The other agents involved are the *Percept Agents* and the *Head Agent*. The role of the *Percept Agents* is to activate the *Context Agents*. Activated *Context Agents* are called *Valid Context Agents*. There are as many *Percept Agents* as *perceptions*. The section 6.1.1 will detail how the activation of *Context Agents* is optimized thanks to the *Percept Agents*. The *Head Agent* communicates with the *Valid Context Agents* to retrieve their model propositions according to the *perceptions*. When the *Context Agents* are learning, the *Head Agent* provides them the *exogenous prediction vector* $\mathcal{O}^{exo}_{m}$. Indeed, *Context Learning* uses *Supervised Learning* and thus *exogenous learning situations*.

Figure 6.1b shows where the mechanisms presented in the previous chapter appear. The *neighborhoods* enable local communications between *Context Agents*. These communications

are at the origin of the *endogenous perceptions* generation that can replace the *exogenous perceptions* during the resolution of *learning inaccuracies* (section 5.4). Still with the addition of *neighborhoods*, *endogenous learning situations* are generated by and for the *Context Agents* thanks to the *Cooperative Neighborhood Learning* mechanism (section 5.6).



(a) Exogenous classical *Context Learning* with *AMOEBA*



(b) *Exogenous* and *Endogenous Context Learning* with *ELLSA*

Figure 6.1 – *Context Learning* VS *Endogenous Context Learning*

In this chapter, we present the learning mechanism *ELLSA*. The intervention of endogenous mechanisms at different points in the learning process is explained. To do so, we present the learning and exploitation rules of *ELLSA* that rely on *Endogenous Context Learning*. For each cycle, the optimization of *Context Agents* activation with *Percept Agents* is detailed. The learning and exploitation process rely on criticality measures that are explained.

Then, we look deeper into the *Context Agents* to outline a mechanism for *Lifelong Learning* that we call *Lifelong Context Learning*. *Lifelong Context Learning* enables to update any learning

model taking into account its past experiences without memorizing all of them.

According to the availability of learning data, we detail two learning strategies that are the *Active Learning Strategy* and the *Self-Learning Strategy*. The *Active Learning Strategy* is based on the generation of *endogenous perceptions* to actively resolve the *learning inaccuracies* and accelerate the learning process with specific *exogenous prediction vectors*. The *Self-Learning Strategy* is based on the generation of *endogenous perceptions* to resolve the *learning inaccuracies* without *exogenous prediction vectors*.

This chapter is ended with reflections about *Transfer Learning* and *Reinforcement Learning* in the *Context Learning* framework.

6.1 Endogenous Context Learning

In this section, learning cycles and exploitation cycles are detailed. The introduction of the *neighborhood* during a cycle brings additional information that can be helpful during the construction of the *Context Agents validity ranges* and *local models*.

A cycle c_l of the learning mechanism *ELLSA* is defined by a common input which is the vector of *perceptions* $\mathcal{P}_n$. A vector of *perceptions* alone defines an *exploitation situation* $\mathcal{E}_n = [\mathcal{P}_n]$ and thus an exploitation cycle. *Perceptions* associated with an *exogenous prediction vector* $\mathcal{O}_m^{exo}$ define a learning cycle with an *exogenous learning situation* as input $\mathcal{L}_{n,m}^{exo} = [\mathcal{P}_n, \mathcal{O}_m^{exo}]$. During a learning cycle, *Context Agents* update their models with the *exogenous learning situation*.

$\mathcal{P}_n^{c_l}$, $\mathcal{O}_m^{exo,c_l}$, $\mathcal{O}_m^{j,c_l}$ and $\mathcal{O}_m^{c_l\prime}$ respectively represent *perceptions*, an *exogenous prediction vector*, a *local prediction vector* from a *Context Agent* $\mathcal{C}_n^j$ and an *output prediction vector* for a cycle c_l.

Each cycle results in the operations of the algorithm 6.1. The *Percept Agents* use the input $\mathcal{P}_n^{c_l}$ to activate the *Valid Context Agents*. The activated *Valid Context Agents* and *Neighbor Context Agents* notify to the *Head Agent* that they have been activated. The *Head Agent* selects the *Best Context Agent* to set the *output prediction vector* $\mathcal{O}_m^{c_l\prime}$. Depending on the availability of an *exogenous prediction vector* $\mathcal{O}_m^{exo,c_l}$, the *Head Agent* provides feedback to the *Valid Context Agents* according to their respective *local models*. It then processes different *Non Cooperation Situations* (*NCS*) that will be detailed in the following. They enable the construction of the *Context Agents* and the processing of *learning inaccuracies* during the learning and the exploitation cycles.

The table 6.1 recalls a summary of the different *Context Agents* types during *ELLSA* cycles.

The activation of *Valid Context Agents* and their *neighborhood* during the *Percept Agents* cycles is an optimized process that is firstly presented in the following. The *Context Agents* cycles are straight forward as *Valid Context Agents* and *Neighbor Context Agents* only notify their activation. The difference of execution between exploitation and learning cycles appears in the *Head Agent* cycle only. Further explanations will concern the *Head Agent* cycle by detailing learning cycles and exploitation cycles.

Algorithm 6.1: Execution cycle c_l of *ELLSA*

Input : $\mathcal{P}_n^{c_l}$ or $\mathcal{L}_{n,m}^{c_l} = [\mathcal{P}_n^{c_l}, \mathcal{O}_m^{exo,c_l}]$
Output: $\mathcal{O}_m^{c_l\prime}$

Percept Agents Cycles
Percept Agents use the *perceptions* input $\mathcal{P}_n^{c_l}$ to activate the *Valid Context Agents* ${}^{\mathcal{V}}\mathcal{C}_n$ and their *neighborhood* the *Neighbor Context Agents* ${}^{\mathcal{N}}\mathcal{C}_n$;
Context Agents Cycles
Valid Context Agents ${}^{\mathcal{V}}\mathcal{C}_n$ and *Neighbor Context Agents* ${}^{\mathcal{N}}\mathcal{C}_n$ notify their activation to the *Head Agent*;
Head Agent Cycle
The *Head Agent* selects the *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_n$ for the final *output prediction vector* $\mathcal{O}_m^{c_l\prime}$;
if *exogenous prediction vector* $\mathcal{O}_m^{exo,c_l}$ *is available* **then**
    Valid Context Agents ${}^{\mathcal{V}}\mathcal{C}_n$ receive feedback about their *local model* f_n^j;
    NCS are processed using $\mathcal{O}_m^{exo,c_l}$;
else
    NCS are processed without $\mathcal{O}_m^{exo,c_l}$;
end

Context Agents Types	Notations	Description
Context Agents	$\mathcal{C}_n$	They represent the accumulated learning with *validity ranges* $\mathcal{R}_n^j$ associated to *local models* f_n^j.
Valid Context Agents	${}^{\mathcal{V}}\mathcal{C}_n$	They are *Context Agents* that are activated by the *Percept Agents* using the *perceptions* $\mathcal{P}_n$. They provide *local prediction vectors* $\mathcal{O}_m^j$.
Neighbor Context Agents	${}^{\mathcal{N}}\mathcal{C}_n$	They are *Context Agents* that are activated by the *Percept Agents* using the *neighborhood* of the *perceptions* $\mathcal{P}_n$ and the *influences* of *Context Agents*.
Prediction Neighbor Context Agents	${}^{\mathcal{PN}}\mathcal{C}_n$	They are *Neighbor Context Agents* which *local prediction vectors* are in *prediction neighborhoods*.
Best Context Agent	${}^{\mathcal{B}}\mathcal{C}_n$	It is the *Context Agent* for which the *local model* is considered the best among the collective according to the *perceptions* and the *exogenous prediction vector* if available.

Table 6.1 – Table different types of *Context Agents* during learning and exploitation cycles.

6.1.1 Percept Agents Cycles

The *Percept Agents* are responsible for the activation of the *Valid Context Agents* and the *Neighbor Context Agents*. There is one *Percept Agent* $\mathcal{P}ct_i$ by perception p_i.

A *Percept Agent* starts by updating the minimal and maximal experienced values p_i^{min} and p_i^{max} for its perception p_i. Each *Percept Agent* computes the *Valid Context Agents* and *Neighbor Context Agents* projections ${}^{\mathcal{V}}\mathcal{C}_{p_i}$ and ${}^{\mathcal{N}}\mathcal{C}_{p_i}$ according to its perception p_i. A *Context Agent* $\mathcal{C}_n^j$ is valid for the perception p_i if p_i is contained in the *validity range* r_i^j. A *Context Agent* is a *Neighbor Context Agent* for the perception p_i if its *validity range* r_i^j intersects the *neighborhood* or if its *influence* contains p_i (section 5.3).

${}^{\mathcal{V}}\mathcal{C}_n$ and ${}^{\mathcal{N}}\mathcal{C}_n$ represent the *Valid Context Agents* and the *Neighbor Context Agents* once all *Percept Agents* have been processed. At the end of their cycles, *Percept Agents* intersect the projected *Valid Context Agents* ${}^{\mathcal{V}}\mathcal{C}_{p_i}$ and the projected *Neighbor Context Agents* ${}^{\mathcal{N}}\mathcal{C}_{p_i}$ with ${}^{\mathcal{V}}\mathcal{C}_n$ and ${}^{\mathcal{N}}\mathcal{C}_n$. When a *Context Agent* is neither a *Valid Context Agent* nor a *Neighbor Context Agent* in a perception p_i, it is eliminated from the potential *Valid Context Agents* and *Neighbor Context Agents* of other *perceptions* to accelerate the processing of other *Percept Agents*. The algorithm 6.2 shows the main operations of a *Percept Agent*. ${}^{\mathcal{V}}\mathcal{C}_n$ and ${}^{\mathcal{N}}\mathcal{C}_n$ are shared variables so that each *Percept Agent* can be parallelized in a thread.

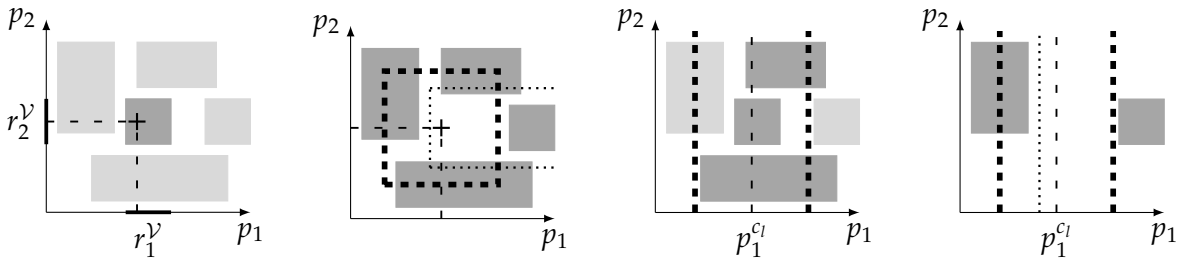
Algorithm 6.2: Execution cycle c_l of a *Percept Agent*

Input : p_i
Output : ${}^{\mathcal{V}}\mathcal{C}_{p_i}$ and ${}^{\mathcal{N}}\mathcal{C}_{p_i}$
Shared Variables: ${}^{\mathcal{V}}\mathcal{C}_n$ and ${}^{\mathcal{N}}\mathcal{C}_n$

1 Adjust p_i^{min} and p_i^{max};
2 ***Compute Valid Context Agents*** ${}^{\mathcal{V}}\mathcal{C}_{p_i}$ ***and Neighbor Context Agents*** ${}^{\mathcal{N}}\mathcal{C}_{p_i}$ ***projections on the perception*** p_i
3 ${}^{\mathcal{V}}\mathcal{C}_{p_i} \leftarrow$ all ${}^{\mathcal{V}}\mathcal{C}_{p_i}^j$ with $p_i \in r_i^j$;
4 ${}^{\mathcal{N}}\mathcal{C}_{p_i} \leftarrow$ all ${}^{\mathcal{N}}\mathcal{C}_{p_i}^j$ with $([p_i - r_i^{\mathcal{N}}; p_i + r_i^{\mathcal{N}}] \cap r_i^j \neq \varnothing)$ or $(r_{i,center}^j \in [p_i - r_{j,i}^{\mathcal{I}}; p_i + r_{j,i}^{\mathcal{I}}])$;
5 Process ${}^{\mathcal{V}}\mathcal{C}_{p_i} \cap {}^{\mathcal{V}}\mathcal{C}_n$ and ${}^{\mathcal{N}}\mathcal{C}_{p_i} \cap {}^{\mathcal{N}}\mathcal{C}_n$;

The activation of *Context Agents* can also be done without all *perceptions*, we call it activation with sub-*perceptions*. Figure 6.2 shows the different cases of *Context Agents* activation with two *perceptions* and thus two *Percept Agents*. Figure 6.2a shows a *Valid Context Agent* (the darker one) activated by its *validity ranges* $r_1^{\mathcal{V}}$ and $r_2^{\mathcal{V}}$. Figure 6.2b shows the activation of the *Neighbor Context Agents*. When only a sub-set of *perceptions* is considered for the activation of the *Valid Context Agents* and the *Neighbor Context Agents*, the other *perceptions* are ignored. Figure 6.2c, only p_1 is used to detect the *Valid Context Agents* (the darker ones). The activation is processed with only the *validity ranges* on the perception p_1 and the current perception $p_1^{c_l}$. It is the same for the *Neighbor Context Agents* (Fig. 6.2d).

The *neighborhood radius* $r_i^{\mathcal{N}}$, introduced section 5.3, enables to detect *Neighbor Context Agents* but also *learning inaccuracies* by exploring the surroundings of the *perceptions*. It is dependent from the minimal and maximal experienced *perceptions* p_i^{min} and p_i^{max}. It is then



(a) *Valid Context Agent* with p_1 and p_2

(b) *Neighbor Context Agents* with p_1 and p_2

(c) *Valid Context Agents* with p_1

(d) *Neighbor Context Agents* with p_1

Figure 6.2 – *Valid Context Agents* and *Neighbor Context Agents* activation with and without all *perceptions*. Dashes represent the *neighborhood* area and dots represent the influence zones of *Context Agents*.

dependent of the *perceptions* space exploration. If the space is little explored, the *neighborhood* can remain limited and so the future exploration. If the space is entirely explored, the *neighborhood* reaches its largest possible size and the exploration is optimal. To accelerate p_i^{min} and p_i^{max} discovery during the first learning cycles, a *bootstrap cycles number* c_{boot} is introduced. During the first cycles of learning, no *Context Agent* is generated in order to reach a sufficient degree of *perceptions* exploration. Past the *bootstrap cycles*, the creation of *Context Agents* can start with optimal sizes. The *bootstrap cycles number* is a user parameter since the user of the learning mechanism should have an idea of the space to be explored during the training. The *bootstrap cycles number* must be must be at least n to let the learning mechanism initialize all p_i^{min} and p_i^{max}.

Table 6.2 reminds the user and designer parameters involved in the activation of *Valid Context Agents* and *Neighbor Context Agents*. The *validity ranges precision* is the user parameter that sets the mapping precision of the *Context Agents* along with the size of the *neighborhood*. The *bootstrap cycles number* is indirectly related to the space exploration as it enables the *Percept Agents* to work before starting to learning with the *Context Agents*. The *neighborhood radius coefficient* and the *influence radius coefficient* are designer coefficients to configure and evaluate different *neighborhood* and *influence* zones.

	Name	Notation	Constrains	Domain
User Parameter	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$
	bootstrap cycles number	c_{boot}	$\geqslant n$	$\mathbb{N}$
Designer Paramaters	*neighborhood radius coefficient*	$\alpha^{\mathcal{N}}$	> 0	$\mathbb{R}$
	influence radius coefficient	$\alpha^{\mathcal{I}}$	> 0	$\mathbb{R}$
Designer Distances	*neighborhood radius*	$r_i^{\mathcal{N}}$	$(p^{\mathcal{R}},\alpha^{\mathcal{N}})$-dependent	$\mathbb{R}$
	influence radius	$r_{j,i}^{\mathcal{I}}$	$(\alpha^{\mathcal{I}})$-dependent	$\mathbb{R}$

Table 6.2 – Table of user and designer parameters and distances involved in the activation of *Valid Context Agents* and *Neighbor Context Agents*.

6.1.2 Learning Cycles

A learning cycle is defined by a *learning situation* $\mathcal{L}_{n,m}$ as input of the learning mechanism. It is the availability of *perceptions* $\mathcal{P}_n$ associated with an *exogenous prediction vector* $\mathcal{O}_m^{exo}$. To take into account *learning inaccuracies* during the learning process, it is necessary to modify the rules presented in section 4.4. The algorithm 6.3 sums up the main mechanisms of a learning cycle which are detailed below. Once that the *Valid Context Agents* and *Neighbor Context Agents* have been activated, the *Head Agent* finishes the learning execution cycle.

6.1.2.1 Algorithm Description

Initialization

The minimum and maximum experienced exogenous predictions o_k^{min} and o_k^{max} are updated on the vectors $\mathcal{O}_m^{min}$ and $\mathcal{O}_m^{max}$. The last *local prediction vectors* $\mathcal{O}_m^{j,last}$ of the *Neighbor Context Agents* are updated with the *perceptions* $\mathcal{P}_n^{c_l}$. This is necessary to set the *prediction neighborhoods* for the *Cooperative Neighborhood Learning* mechanism.

***Best Context Agent* Selection**

The *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_n$ is selected using the smallest *learning criticality* $Crit^{lrn}$ that will be detailed section 6.2.1. The *Best Context Agent* is sought among the *Valid Context Agents*. If there aren't any, it is sought among the *Neighbor Context Agents*. If there are no *Valid Context Agents* nor *Neighbor Context Agents*, the *Best Context Agent* can be searched among all the *Context Agents*.

The *Best Context Agent* provides the final *output prediction vector* $\mathcal{O}_m^{c_l\prime}$. At initialization, when no *Context Agent* have been created, the minimal and maximal prediction vectors $\mathcal{O}_m^{min}$ and $\mathcal{O}_m^{max}$ are used to provide a mean *output prediction vector*. One of our goals is to always provide an *output prediction vector* even with very few learning cycles.

***Valid Context Agents* Feedback**

Each *Valid Context Agents* then receives feedback about its *local model*. If the *local model* is incomplete (see sections 5.4.3.1 and 5.4.5.1), it is updated with the *learning situation* $\mathcal{L}_{n,m}^{c_l} = [\mathcal{P}_n^{c_l}, \mathcal{O}_m^{exo,c_l}]$. If the *local model* is complete and reliable, the *model prediction distance* $d_{\mathcal{L}_{n,m}}^{f_j}$ is compared to the *model error margin* m_{err}^f. The *inaccuracy margin* (section 4.4) is removed so there is only one error margin to define the prediction accuracy: the *model error margin* m_{err}^f.

The *model error margin* is a user parameter that represents the desired precision on the different values of the *output prediction vector*. The *model prediction distance* $d_{\mathcal{L}_{n,m}}^{f_j}$ is the distance between a *local model* f_n^j and a *learning situations* $\mathcal{L}_{n,m}$. It is a model-dependent distance that must be specified by the designer when he selects the underlying learning models. This distance enables to assess the relevance of a model according to a *learning situation*. The *model prediction distance* used in this work will be presented in the section 6.2.3.3.

If the prediction of a *Valid Context Agents* is correct ($d_{\mathcal{L}_{n,m}}^{f_j} < m_{err}^f$), it becomes a *Good*

Context Agent. It is updated with the lifelong updating mechanisms that will be presented section 6.3. Always updating *Good Context Agents* enables to be robust to noise in the *exogenous learning situations* by accumulating more *learning situations* than necessary to complete a model. Each *Good Context Agent* receives a *confidence* bonus. If a *Valid Context Agent* is not good, it triggers a *Bad Prediction NCS* and its *confidence* decreases (section 6.2.3.2 will detail the dynamics of *confidence* bonuses and penalties).

Algorithm 6.3: Learning cycle c_l for the *Head Agent*

Input : $\mathcal{L}^{c_l}_{n,m} = [\mathcal{P}^{c_l}_n, \mathcal{O}^{exo,c_l}_m]$
Output: $\mathcal{O}^{c_l\prime}_m$

Update all $\mathcal{O}^{min}_m$ and $\mathcal{O}^{max}_m$;
Update *Neighbor Context Agents* ${}^{\mathcal{N}}\mathcal{C}_n$ last *local prediction vectors* $\mathcal{O}^{j,last}_m$;
Select the *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_n$ with the lowest *learning criticality* $Crit^{lrn}$;
Set the *output prediction vector* $\mathcal{O}^{c_l\prime}_m$;
Send feedback to Valid Context Agents
  Foreach *Valid Context Agents* ${}^{\mathcal{V}}\mathcal{C}^j_n$
    if *local model* f^j_n *incomplete* **then**
      Update *Valid Context Agent* models f^j_n ;
    else if $d^{f_j}_{\mathcal{L}_{n,m}} < m^f_{err}$ **then**
      Lifelong update of *Valid Context Agent* models f^j_n ;
      c^j ++ ;
    else
      Bad Prediction NCS ;
      c^j − = 2 ;
    end
if $c_l > c_{boot}$ **then**
  Uselessness NCS ;
  Conflict NCS and *Concurrency NCS* resolution;
  Unproductivity NCS ;
  Complete Redundancy NCS and *Partial Redundancy NCS* resolutions;
  Learning inaccuracies NCS detection ;
end

Other NCS

If the cycle of execution is higher than the *bootstrap cycles number*, all other *NCS* can be treated (*Uselessness NCS*, *Conflict NCS*, *Concurrency NCS*, *Unproductivity NCS*, *Complete Redundancy NCS* and *Partial Redundancy NCS*). A learning cycle is ended by seeking the *learning inaccuracies NCS* presented section 5.4. Each detected *learning inaccuracy NCS* generates *endogenous perceptions* $\mathcal{P}^{endo}_n$ that situates the *learning inaccuracy*. $\mathcal{P}^{endo}_n$ represents a proposition for a future *learning situation*. When the $\mathcal{P}^{endo}_n$ are used by the oracle to provide a new *learning situation*, it corresponds to the *Active Learning Strategy* that will be detailed section 6.4.1. When the $\mathcal{P}^{endo}_n$ are self-requested as *endogenous exploitation situations*, it corresponds to the *Self-Learning Strategy*. Both strategies will be described in section 6.4.2.

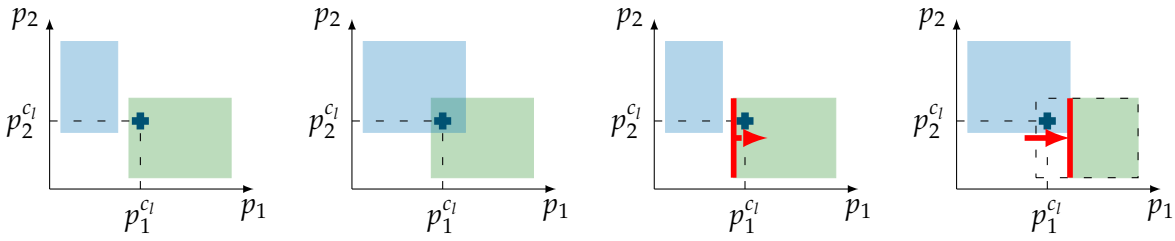
6.1.2.2 Learning NCS

The Non Cooperative Situations (*NCS*) that intervene in a learning cycle are presented here. Some of them are inspired from classical *Context Learning* (section 4.4) [Nigon, 2017]. The main contribution here is the addition of the *Neighbor Context Agents* to resolve the *NCS*.

Bad Prediction NCS

Detection. The *Bad Prediction NCS* is triggered when a *Valid Context Agent* has a bad model: the *model prediction distance* is higher than the *model error margin* ($d^{f_j}_{\mathcal{L}_{n,m}} > m^f_{err}$). The *Valid Context Agent* is not good for the *exogenous learning situation*, its model is not close enough given the expected precision.

Resolution. To resolve this situation, the *Context Agent* moves one of its *validity ranges* r^j_i to exclude the current *perceptions*. At the same time, it communicates with the other *Valid Context Agents* to destroy any possible overlap by choosing the largest if there are several of them. The overlap detection is made as described in the sections 5.4.2.2 and 5.4.2.3. The *confidence* is decreased depending on the presence of an overlap (section 6.2.3.2). The *validity range* which is chosen for the modification is the one that least affects the volume of the *Context Agent* that is bad for current *perceptions* (p_1 in the figures 6.3a and 6.3b). This ensures that as little knowledge and experience as possible is lost for the *Context Agent*. To find the perception that will least affect the *Context Agent* volume, potentially lost volumes are calculated. The perception corresponding to the smallest is selected.



(a) Before Resolution Without Overlap

(b) Before Resolution With Overlap

(c) Resolution Without Overlap

(d) Resolution With Overlap

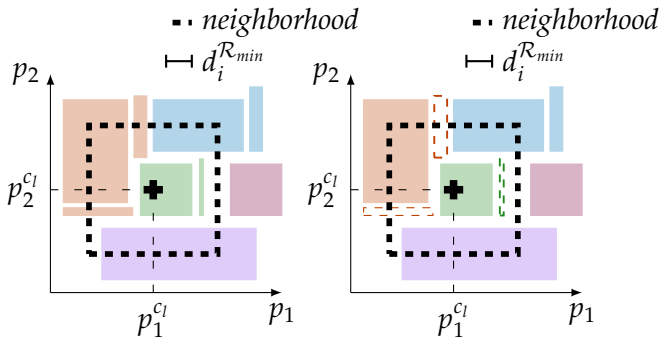
Figure 6.3 – *Bad Prediction NCS* Resolutions.

Figure 6.3 shows the possibles resolutions with and without any overlaps. When there isn't any overlap, the *validity range* moves inside its closest border with an *AVT* increment. *AVT* stands for *Adaptive Value Tracker*. It is a tool to discover a real dynamic value with successive returns [Lemouzy, 2011]. In our case, it rules the *validity ranges* modification increments and will not be detail any further. When there is an overlap, the *validity range* modifies the closest border in order to suppress the conflicting zone. The *Bad Prediction NCS* enables to process *Conflict NCS* when one of the conflicting *Context Agents* has a bad model.

Uselessness NCS

Detection. The *Uselessness NCS* has the role of eliminating the useless *Context Agents* of the collective. A *Context Agent* is considered useless when the length of one of its *validity ranges* r_i^j is lesser than the *minimum range distance* $d_i^{\mathcal{R}_{min}}$. The *minimum range distance* role is indeed to set the minimal exploration distance. Any *Context Agent* with any *validity range* below this distance should not be considered.

Resolution. Such *Context Agents* disappear and no longer represent a portion of the explored space. To be effective, the detection is made on all the *Neighbor Context Agents*. Figure 6.4 shows an example of resolution in a *neighborhood*.



(a) Before Resolution (b) After Resolution

Figure 6.4 – *Uselessness NCS* Resolution.

Conflict NCS and *Concurrency NCS*

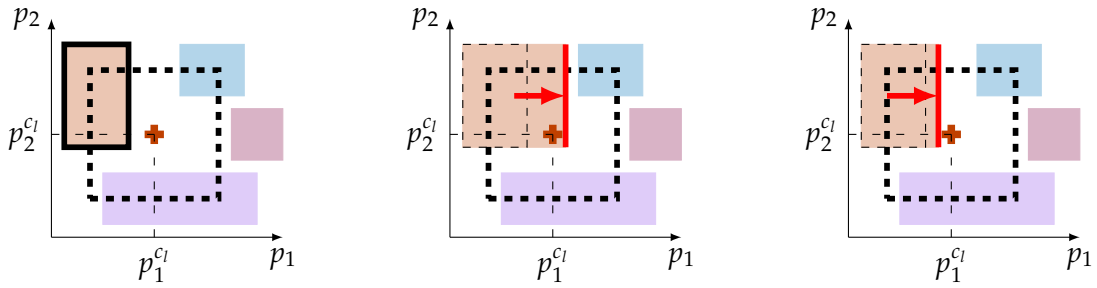
Detection. The *Bad Prediction NCS* deals with some of the *Conflict NCS* detections but to ensure that all *Conflict NCS* and *Concurrency NCS* detections are taking into account, the *Conflict NCS* and *Concurrency NCS* resolutions are called here between the *Valid Context Agents*. As presented in sections 5.4.2.2 and 5.4.2.3, the *model similarity distance* d_{sim}^f is used to determine if *local models* are similar or not. The overlap presence is due to the existence of several *Valid Context Agents*. If several *Context Agents* are valid, overlaps are necessarily present. Highlighting the *perceptions* where *Conflict NCS* and *Concurrency NCS* are is done during the detection of *learning inaccuracies* with the generation of *endogenous perceptions* $\mathcal{P}_n^{endo}$ at the end of the learning cycle.

Resolution. The *Conflict NCS* and *Concurrency NCS* resolutions are processed only if the *Best Context Agent* is also a *Good Context Agent* (i.e. its *model prediction distance* $d_{\mathcal{L}_{n,m}}^{f_j}$ is lower than *model error margin* m_{err}^f). The spatial resolutions are as presented in section 5.4.4.1. The *Context Agent* that wins the resolution is the *Best Context Agent* selected earlier. When a *Context Agent* looses a *Conflict NCS* or a *Concurrency NCS*, its *confidence* is decreased. The dynamic of the *confidence* will be detailed section 6.2.3.2.

Unproductivity NCS

Detection. The *Unproductivity NCS* is triggered when there aren't any *Valid Context Agents*. It means that the current *perceptions* $\mathcal{P}_n^{c_l}$ are not represented by any *Context Agent* (Figures 6.5a, 6.6a, 6.6c and 6.6e).

Resolution. The resolution of such situations can involve two steps. First, the closest *Good Context Agent*, if there is one, tries to include the new *perceptions* $\mathcal{P}_n^{c_l}$ by expanding its *validity ranges*. In a second step, if the attempt to cover the new *perceptions* fails (Fig. 6.5c), a new *Context Agent* is created in addition to the expansion. In the following, the *Good Context Agent* expansion and the *Context Agent* creation mechanisms are detailed.



(a) Before expansion of the *Good Context Agent* in the *neighborhood*

(b) Successful expansion of the *Good Context Agent*

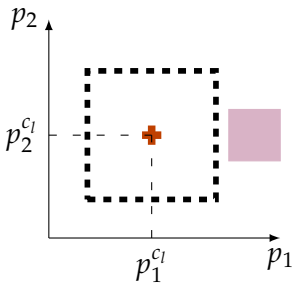
(c) Failed expansion of the *Good Context Agent*

Figure 6.5 – *Unproductivity NCS* expansion resolutions. The *neighborhood* is represented by dashes. The *Good Context Agent* is boxed. For the illustrations, influences are not used.

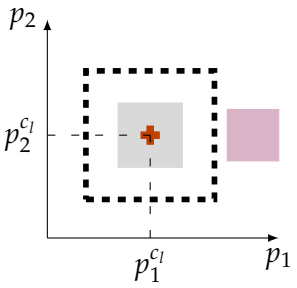
▷ ***Good Context Agent* Expansion**. The closest *Good Context Agents* is sought among the *Neighbor Context Agents*. The distance used to determine proximity is the *range perceptions proximity distance*. It is the distance between the *perceptions* and the *validity ranges* of a *Context Agent* (section 6.2.3.4). If such *Context Agent* exists, it tries to expand its *validity ranges* in order to include the new *perceptions*. The *validity ranges* expansions are also ruled by *AVT* dynamic increment. Figure 6.5a shows a *Good Context Agent* among the *neighborhood* and figure 6.5b shows a successful expansion of the *Good Context Agent* which succeeds to contain the new *perceptions* $\mathcal{P}_n^{c_l}$. There are two cases where the new *perceptions* are not included (Fig 6.5c). The dynamic increment is not large enough or the *validity range* is not allowed to grow anymore. To detect if a *validity range* can no longer grow, the *maximum range radius* r_i^{max} is used.

$$r_i^{max} = \alpha^{\mathcal{R}_{max}} . r_i^{creation} \tag{6.1}$$

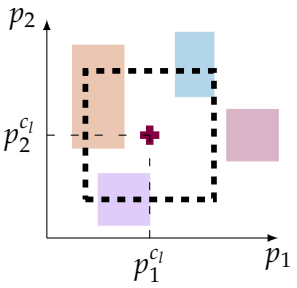
with $\alpha^{\mathcal{R}_{max}}$ the *maximum range radius coefficient*, a designer parameter. And $r_i^{creation}$ the *range creation radius* introduced section 5.3. Forbidding a *Context Agent* to grow is essential to control the space exploration by the *Context Agents*. Thus, a large *Context Agent* cannot cover a zone explored by smaller *Context Agents*. The creation of a new *Context Agents* is then compulsory.



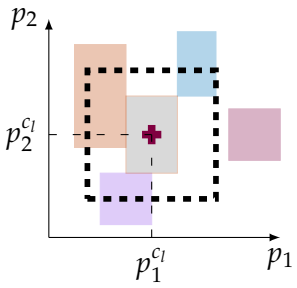
(a) Before creation without *Neighbor Context Agents*



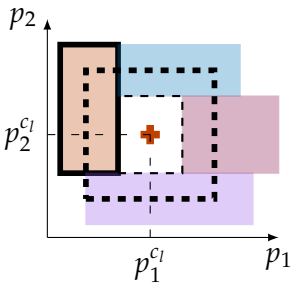
(b) After creation of a new *Context Agent* without *Neighbor Context Agents*



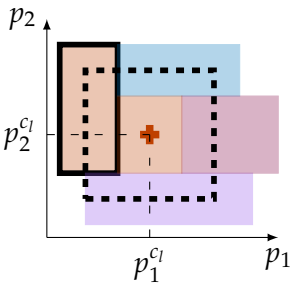
(c) Before creation with *Neighbor Context Agents*



(d) After creation of a new *Context Agent* with *Neighbor Context Agents*



(e) Before creation with *incompetent volume* and *Good Context Agent*



(f) After creation of a new *Context Agent* with *incompetent volume* and *Good Context Agent*

Figure 6.6 – *Unproductivity NCS* creation resolutions. The *neighborhood* is represented by dashes. The *Good Context Agents* are boxed. For the illustrations, influences are not used.

▷ ***Context Agent* creation**. If the *Good Context Agent* expansion fails, the second step is engaged, a new *Context Agent* is created. The creation of a *Context Agent* involves the initialization of the *confidence*, the *local model* and *validity ranges*. The section 6.2.3.2 will detail the initialization of the *confidence*. The initialization of the *validity ranges* depends on the availability of *Neighbor Context Agents* or the availability of a previous *Incompetence NCS* detection with its associated *incompetent volume*. Each case is listed in the following.

— ***Validity ranges* initialization without *Neighbor Context Agents***. In the case that there isn't any *Neighbor Context Agents* (Fig. 6.6a), the new *validity ranges* are initialized with the *range creation radius* $r_i^{creation}$ (Fig. 6.6b).

— ***Validity ranges* initialization with *Neighbor Context Agents***. When there are *Neighbor Context Agents* (Fig. 6.6c), knowing the mean *validity ranges* size gives information about the spatial granularity of the *neighborhood*. The new *validity ranges* are then initialize with the *neighborhood* mean *validity ranges* sizes (Fig. 6.6d).

— ***Validity ranges* initialization with an *incompetent volume***. If a previous *Incompetence NCS* detection occurred, an *incompetent volume* to fill is provided (Fig. 6.6e). The new *validity ranges* are then initialized with the ranges of the *incompetent volume* (Fig. 6.6f).

After initializing the *validity ranges*, the *local model* must be set. The initialization of the

local model depends on the availability of a *Good Context Agent*. A *Good Context Agent* is a *Context Agent* with a good *local model* (i.e. $d^{f_j}_{\mathcal{L}_{n,m}} < m^f_{err}$). The two cases are presented as follows.

— ***Local model* Initialization without *Good Context Agent***. The *local model* is initialized with only one *learning situation*, the *exogenous learning situation* $\mathcal{L}^{exo}_{n,m}$ of the current learning cycle (Figures 6.6b and 6.6d).

— ***Local model* Initialization with *Good Context Agent***. When a *Good Context Agent* is available, its *local model* can be shared to create the new one and thus accelerate the learning process. The *Good Context Agent* sponsors the *Context Agent* creation. The new *local model* is initialized with the *local model* of the sponsor *Context Agent* (Fig. 6.6f). The current *exogenous learning situation* $\mathcal{L}^{exo}_{n,m}$ is used to update the newly created model with the lifelong learning mechanism (section 6.3).

Complete Redundancy NCS* and *Partial Redundancy NCS

We have seen that past certain *validity ranges* sizes, *Context Agents* can no longer grow and generalize. Indeed, growing is a synonym of generalizing in the *Context Learning* paradigm. This is why the *Complete Redundancy NCS* and *Partial Redundancy NCS* were introduced. Detailed in the sections 5.4.2.4, 5.4.2.5, 5.4.4.4 and 5.4.4.5, the *Complete Redundancy NCS* and *Partial Redundancy NCS* are *learning inaccuracies* that can be resolved instantly without generating any *endogenous perceptions* $\mathcal{P}^{endo}_n$. They enable to merge or restructure *Context Agents validity ranges* to accelerate the generalization process and make it continue when the *Context Agents* can no longer do it by themselves. Each *Valid Context Agent* with a complete *local model* seeks with which *Neighbor Context Agent* is could resolve a *Complete Redundancy NCS* or *Partial Redundancy NCS*. Incomplete *local models* are not concerned as their models are not reliable.

The *learning inaccuracies NCS* detections are common to the learning and exploitation cycles. They will be detailed in the section 6.4.

6.1.2.3 Learning Parameters

Table 6.3 reminds the added parameters for the learning cycles of *ELLSA*. One important measure introduced with learning cycle is the *learning criticality*. The *learning criticality* determines the learning degree of confidence of a *Context Agent* during learning. This measure will be detailed in section 6.2. A user parameter that was added in this section is the *model error margin*, it enables to specify the desired prediction performances depending on the *model prediction distance*. The *model prediction distance* is a metric that represents the affinity of a *local model* towards a *learning situation*. This distance is model-dependent and needs to be specified along with the selection of the learning models. The *range perceptions proximity distance* is a metric that measures the proximity of *Context Agents validity ranges* from the *perceptions* input. The last added parameter is the *maximum range radius* that constrains the expansion of the *Context Agents validity ranges*.

	Name	Notation	Constrains	Domain
User Parameter	*model error margin*	m_{err}^{f}	> 0	$\mathbb{R}$
User Metric	*model prediction distance*	$d_{\mathcal{L}_{n,m}}^{f_j}$	model-dependent	$\mathbb{R}$
Designer Paramaters	*maximum range radius coefficient*	$\alpha^{\mathcal{R}_{max}}$	> 1	$\mathbb{R}$
Designer Distance	*maximum range radius*	r_i^{max}	$(p^{\mathcal{R}}, \alpha^{\mathcal{R}_{max}})$-dependent	$\mathbb{R}$
Designer Metric	*range perceptions proximity distance*	$d_{\mathcal{P}_n}^{\mathcal{R}_n^j}$	≥ 0	$\mathbb{R}$
Designer Criticality	*learning criticality*	$Crit^{lrn}$	> 0	$\mathbb{R}$

Table 6.3 – Table of user and designer parameters, distances, and criticalities involved in learning cycles.

6.1.3 Exploitation Cycles

An exploitation cycle is defined by an *exploitation situation* $\mathcal{E}_n$ as input of the learning mechanism. *Exploitation situations* are only *perceptions*: $\mathcal{E}_n = [\mathcal{P}_n]$. Its is a request to the learning mechanism in order to obtain an *output prediction vector* $\mathcal{O}'_m$ for specific *perceptions*. Usually, when a learning mechanism is exploited, it no longer learns. One of the goals of this thesis is to make *ELLSA* learn even during exploitation. Thus *learning inaccuracies* can be processed during an exploitation cycle. The algorithm 6.4 presents the main mechanisms of an exploitation cycle.

6.1.3.1 Algorithm Description

The *Valid Context Agents* and *Neighbor Context Agents* are still activated by the *Percept Agents* before the *Head Agent* exploitation cycle. The last *local prediction vectors* $\mathcal{O}_m^{j,last}$ of the *Neighbor Context Agents* are also updated with the *perceptions* $\mathcal{P}_n^{c_l}$. The *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_n$ is selected among the *Valid Context Agents* using the smallest *exploitation criticality* $Crit^{expl}$ that will be detailed in section 6.2.1. Like in learning cycles, the search is expanded to the *Neighbor Context Agents* and to all the *Context Agents* if there isn't any *Valid Context Agents* nor *Neighbor Context Agents*. The *Best Context Agent* provides the *output prediction vector* $\mathcal{O}_m^{c_l\prime}$. As in learning cycles, if no *Best Context Agent* is found, the last *Best Context Agent* or the minimal and maximal experienced *prediction vectors* $\mathcal{O}_m^{min}$ and $\mathcal{O}_m^{max}$ are used.

Algorithm 6.4: Exploitation cycle c_l for the *Head Agent*

Input : $\mathcal{P}_n^{c_l}$
Output: $\mathcal{O}_m^{c_l\prime}$

1 Update *Neighbor Context Agents* ${}^{\mathcal{N}}\mathcal{C}_n$ last *local prediction vectors* $\mathcal{O}_m^{j,last}$;
2 Select the *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_n$ with the *exploitation criticality* $Crit^{expl}$;
3 Set the *output prediction vector* $\mathcal{O}_m^{c_l\prime}$;
4 **if** $c_l > c_{boot}$ **then**
5 | *Uselessness NCS* ;
6 | *Conflict NCS* and *Concurrency NCS* resolution;
7 | *Unproductivity NCS* ;
8 | *Complete Redundancy NCS* and *Partial Redundancy NCS* resolutions;
9 | *Learning inaccuracies NCS* detection ;
10 **end**

If the cycle of execution is higher than the *bootstrap cycles number*, as in learning cycles, all the *NCS* can be treated and the *learning inaccuracies* can be detected.

6.1.3.2 Exploitation NCS

The *Uselessness NCS*, the *Complete Redundancy NCS* and the *Partial Redundancy NCS* are unchanged compared to the learning *NCS*. There are no *Bad Prediction NCS* during exploitation cycles because the *local models* cannot be compared to any oracle's *prediction vectors*. All

Conflict NCS and *Concurrency NCS* detections are treated by the *Conflict NCS* and *Concurrency NCS* resolutions. The difference with the learning cycles is that the *Best Context Agent* is selected with the *exploitation criticality*. As there isn't any *exogenous prediction vector*, it cannot be verified that the *Best Context Agent* is a *Good Context Agent*. The *confidences* of the loosing *Context Agents* are affected as during the learning cycles.

The *Unproductivity NCS* behaves differently as the *local models* of the *Context Agents* cannot be verified. In the case that there are *Neighbor Context Agents*, a new *Context Agents* can be created in some cases. The local predictions of the *Neighbor Context Agents* are use to create an *endogenous prediction vector* $\mathcal{O}_m^{endo}$. Each endogenous prediction o_k^{endo} is the mean of the *Neighbor Context Agents* predictions weighted by the inverse *exploitation criticality*.

$$o_k^{endo} = \frac{\sum_{\mathcal{N}\mathcal{C}_n^j} Crit_j^{expl^{-1}}.o_k^j}{\sum_{\mathcal{N}\mathcal{C}_n^j} Crit_j^{expl^{-1}}} \tag{6.2}$$

The *local model* of the created *Context Agent* is then initialized with the *learning situation* $\mathcal{L}_{n,m} = [\mathcal{P}_n^{c_l}, \mathcal{O}_m^{endo}]$ Depending on the *Neighbor Context Agents* or the *Incompetence NCS* detection, the *validity ranges* of the new *Context Agent* are initialized as in the learning cycles. The *confidence* initialization is detailed in section 6.2.3.2. However, to ensure that the weighted endogenous prediction are reliable, the number of *Neighbor Context Agents* must be greater that the *creation neighbors number* $n^{creation}$. $n^{creation}$ is a designer parameter.

6.1.3.3 Exploitation with Sub-Perceptions

In the case of an exploitation with sub-*perceptions*, the given subset of *perceptions* is used to define the set of *Valid Context Agents* (section 6.1.1). The *Best Context Agent* is chosen using the *exploitation criticality*. The unspecified *perceptions* are set by default to the centers of the *Best Context Agent validity ranges*.

6.1.3.4 Exploitation Parameters

Table 6.4 reminds the added parameters for the exploitation cycles of *ELLSA*. The *exploitation criticality* represents the exploitation confidence of a *Context Agent*. It will be detailed section 6.2. $n^{creation}$ is the minimal number of *Neighbor Context Agents* that are necessary to create a new *Context Agent* without an *exogenous learning situation*.

	Name	Notation	Constrains	Domain
Designer Paramaters	*creation neighbors number*	$n^{creation}$	> 0	$\mathbb{N}$
Designer Criticality	*exploitation criticality*	$Crit^{expl}$	> 0	$\mathbb{R}$

Table 6.4 – Table of user and designer parameters involved in exploitation cycles.

6.2 Criticalities

The *learning criticality* and the *exploitation criticality* have been introduced in learning and exploitation cycles to represent high level learning and exploitation confidences. The lower the criticality is, the higher the confidence is. They enable to select the *Best Context Agent* during the learning and exploitation cycles. The criticalities can gather the affinity with *learning situations* (i.e. the *model prediction distance*), the *confidences* of *Context Agents*, their volume and the distances to the *perceptions* (i.e. the *range perceptions proximity distance*). In the following the *learning criticality*, the *exploitation criticality* and their metrics are presented.

6.2.1 Learning Criticality

The *learning criticality* $Crit^{lrn}$ gathers the *model prediction distance* $d^{f_j}_{\mathcal{L}_{n,m}}$ with a *learning situation*, the *normalized confidence* $c^j_{0,1}$ of a *Context Agent* and its *validity ranges* volume $\mathcal{V}^j_n$.

$$Crit^{lrn} = \frac{w^{lrn}_{f_n}.d^{f_j}_{\mathcal{L}_{n,m}} + w^{lrn}_{c_{0,1}}.{c^j_{0,1}}^{-1} + w^{lrn}_{\mathcal{R}_n}.{\mathcal{V}^j_n}^{-1}}{w^{lrn}_{f_n} + w^{lrn}_{c_{0,1}} + w^{lrn}_{\mathcal{R}_n}} \tag{6.3}$$

- $w^{lrn}_{f_n}$ is the *accuracy learning weight*. It represents the weight given to *model prediction distance* $d^{f_j}_{\mathcal{L}_{n,m}}$. It is the weight given to the affinity of a *local model* with an *exogenous learning situation* $\mathcal{L}^{exo}_{n,m}$. The *model prediction distance* measures the accuracy of a *local model*.
- $w^{lrn}_{c_{0,1}}$ is the *experience learning weight*. It represents the weight given to the *normalized confidence* $c^j_{0,1}$. The *confidence* is an image of a *Context Agent* history, its past experiences. It is the result of all the past successes and failures of a *Context Agent*. The *confidence* needs to be normalized to be used in the *learning criticality* (section 6.2.3.2).
- $w^{lrn}_{\mathcal{R}_n}$ is the *generalization learning weight*. It represents the weight given to the *validity ranges* volume of a *Context Agent*. The volume represents the generalization of a *Context Agent* on the continuous space of *perceptions*.

The three learning weights enable to dose the importance of the model accuracy, the experience history and the generalization of *Context Agents* during the selection of the *Best Context Agent* for the learning cycles. They are three important facets of learning.

6.2.2 Exploitation Criticality

The *exploitation criticality* $Crit^{expl}$ gathers the *range perceptions proximity distance* $d^{\mathcal{R}^j_n}_{\mathcal{P}_n}$ with *perceptions*, the *normalized confidence* $c^j_{0,1}$ of a *Context Agent* and its *validity ranges* volume $\mathcal{V}^j_n$. As during an exploitation cycle an *exogenous prediction vector* is not available, the *model prediction distance* is replaced by the *range perceptions proximity distance*.

$$Crit^{expl} = \frac{w^{expl}_{\mathcal{P}_n}.d^{\mathcal{R}^j_n}_{\mathcal{P}_n} + w^{expl}_{c_{0,1}}.{c^j_{0,1}}^{-1} + w^{expl}_{\mathcal{R}_n}.{\mathcal{V}^j_n}^{-1}}{w^{expl}_{\mathcal{P}_n} + w^{expl}_{c_{0,1}} + w^{expl}_{\mathcal{R}_n}} \tag{6.4}$$

- $w^{expl}_{\mathcal{P}_n}$ is the *proximity exploitation weight*. It represents the weight given to *range perceptions proximity distance* $d^{\mathcal{R}^j_n}_{\mathcal{P}_n}$. It is the weight given to the proximity of *validity ranges* to input *perceptions*.
- $w^{expl}_{c_{0,1}}$ is the *experience exploitation weight*. It represents the weight given to the *normalized confidence* $c^j_{0,1}$ during exploitation cycles.
- $w^{expl}_{\mathcal{R}_n}$ is the *generalization exploitation weight*. It represents the weight given to the *validity ranges* volume of a *Context Agent* during exploitation cycles.

The three exploitation weights enable to dose the importance of the proximity, the experience history and the generalization of *Context Agents* during the selection of the *Best Context Agent* for the exploitation cycles.

6.2.3 Metrics

The learning and exploitation criticalities and the previously presented mechanisms use several metrics that are detailed in the following. The metrics that are described in the following enable to measure *Context Agents* generalization and experience, *local models* affinity and similarity, and *validity ranges* proximity in the *perceptions* space.

6.2.3.1 Generalization

The generalization is measured with the volumes of the *Context Agents validity ranges*. The larger the *validity ranges* are, the more a *Context Agent* generalizes over a large part of the continuous exploration space. The volume of a *Context Agent* $\mathcal{V}^j_n$ is the simple product of all the *validity ranges* r^j_i lengths.

$$\mathcal{V}^j_n = \prod_i \overline{r^j_i} = \prod_i (r^j_{i,end} - r^j_{i,start}) \tag{6.5}$$

6.2.3.2 Experience

Experience is measured with the *confidence* of the *Context Agents*. *Confidence* is initialized at the creation of *Context Agents*. It then evolves though different experiences that can be either successes or failures. The impact of positive and negative events on the *Context Agents* are firstly detailed. Then, to compare the *confidence* to the other metrics, a normalization mechanic is described.

Confidence Dynamics

Table 6.5 reminds all the *confidences* initializations and variations during learning and exploitation cycles. The reference initialization *confidence* is 0.5. It corresponds to the creation of a *Context Agent* with one *exogenous learning situation*. The initialization *confidence* is higher

when a *Neighbor Context Agent* is sponsoring the creation. The creation with an *endogenous learning situation* provides a much lower *confidence* as the *endogenous prediction vector* cannot be verified.

The positive experiences that are defined as successes are the good *local model* feedback, the volume gains and additional *endogenous learning situation* generated with the *CNL* mechanism. A good *local model* feedback is a good affinity of *local model* with a *learning situation*. It increases the *confidence* by 1. The volume gains or losses represent exchanges of explored areas. *Confidences* varies in proportion to the exchanged volumes. As in the creation of a *Context Agent*, additional *endogenous learning situations* provide little bonus of *confidence*.

The negative experiences designated as failures are the bad *local model* feeback, *Conflict NCS*, *Concurrency NCS* and volume losses. A bad feedback is more punished than a good one. A *Conflict NCS* loss has the same degree of severity of a bad feedback as the winner of the *Conflict NCS* possesses the lowest *learning criticality* or *exploitation criticality*. The *Concurrency NCS* loss is little punished because it is just a spatial inaccuracy. The *local model* is similar to the one of *Best Context Agent*.

Events	Experiences	Confidence	
		Initialization	Variation
Creations	*Exogenous learning situation* $\mathcal{L}_{n,m}^{exo}$	0.5	
	Sponsor good *local model* and $\mathcal{L}_{n,m}^{exo}$	1	
	Endogenous learning situation $\mathcal{L}_{n,m}^{endo}$	0.01	
Successes	Good *local model* feedback		$+1$
	Merge with a *Context Agent* $\mathcal{C}_n^j$		$+c^j$
	Volume gain from a *Context Agent* $\mathcal{C}_n^j$		$+(1-\alpha^{loss})c^j$
	nb^{endo} *endogenous learning situations* $\mathcal{L}_{n,m}^{endo}$		$+0.01nb^{endo}$
Failures	Bad *local model* feedback		-2
	Conflict NCS loss		-2
	Concurrency NCS loss		-0.5
	Volume loss α^{loss}		$-(1-\alpha^{loss})c^j$

Table 6.5 – Table of *confidence* dynamics through the different experiences of *Context Agents*

Confidence Normalization

The *confidence* c^j of a *Context Agent* $\mathcal{C}_n^j$ is a real number. The other metrics of the *learning criticality* and *exploitation criticality* are positive real values (*model similarity distance*, *validity ranges* volumes, *range perceptions proximity distance*). To make the *confidence* comparable to other learning metrics, it is modified with a Sigmoid normalization function. $c_{0,1}^j$ is the normalized confidence given by the equation 6.6.

$$c_{0,1}^j(c^j) = \frac{1}{1+\mathrm{e}^{-\frac{c^j-p_{ctr}}{p_{disp}}}} \tag{6.6}$$

Let c_{min} and c_{max} the minimal and maximal *confidences* across all the *Context Agents*. To

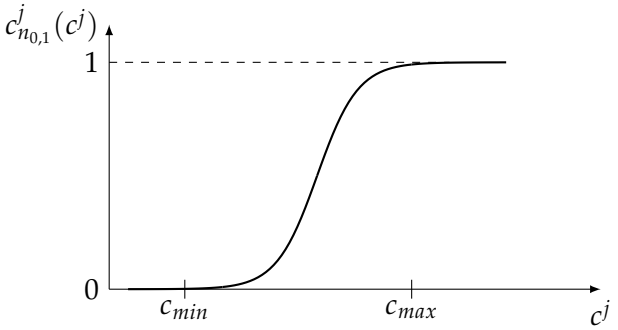


Figure 6.7 – Normalized *confidence* between 0 and 1.

adjust the slope between c_{min} and c_{max}, the systems of equations 6.7 is fixed.

$$\begin{aligned} c^j_{n_{0,1}}(c_{min}) &= C^{min}_{n_{0,1}} \\ c^j_{n_{0,1}}(c_{max}) &= C^{max}_{n_{0,1}} \end{aligned} \tag{6.7}$$

$C^{min}_{n_{0,1}}$ and $C^{max}_{n_{0,1}}$ are fixed such as $C^{min}_{n_{0,1}} = 0.01$ and $C^{max}_{n_{0,1}} = 0.99$. The resolution of the equations gives the values of p_{ctr} and p_{disp} that enable to configure the normalization depending on c_{min}, c_{max}, $c^{min}_{n_{0,1}}$ and $c^{max}_{n_{0,1}}$.

$$\begin{aligned} p_{ctr} &= \frac{c_{max} - c_{min}.CN^{max}_{min}}{1 - CN^{max}_{min}} \\ & \qquad \text{with} \quad CN^{max}_{min} = \frac{\ln\left(\frac{1}{C^{max}_{n_{0,1}}} - 1\right)}{\ln\left(\frac{1}{C^{min}_{n_{0,1}}} - 1\right)} \\ p_{disp} &= \frac{p_{ctr} - c_{max}}{\ln\left(\frac{1}{C^{max}_{n_{0,1}}} - 1\right)} \end{aligned} \tag{6.8}$$

The normalization could be done with a simple linear function but this function offers more possibilities for distributing the *confidences* values between 0 and 1. It also enables to not always update c_{min} and c_{max} as the function is strictly constrained in the interval $]0, 1[$.

6.2.3.3 Model

This section presents the model-dependent distances of the learning mechanism *ELLSA*. The *model prediction distance* $d^{f_j}_{\mathcal{L}_{n,m}}$ and the *model similarity distance* d^{f}_{sim}.

Local Model Affinity

The *model prediction distance* $d^{f_j}_{\mathcal{L}_{n,m}}$ measures the affinity of *learning situations* with a *local model*. In this thesis, we illustrated the use of *local models* with linear regressions. *Context Agents* $\mathcal{C}^j_n$ have local linear regression models as *local models* f^j_n. For each vector of *perceptions* $\mathcal{P}_n$ in the *validity ranges*, a *Context Agent* provides a *local prediction vector* $\mathcal{O}^j_1 \in \mathbb{R}$ calculated from the coefficients of its model $[a^j_0, \ldots, a^j_i, \ldots, a^j_n] \in \mathbb{R}^{n+1}$ and *perceptions* $\mathcal{P}_n$ according to the following equation:

$$\mathcal{O}_1^j = o_1^j = f_n^j(p_1, p_i, \cdots, p_n) = \sum_{i=1}^{n} a_i^j.p_i + a_0^j \tag{6.9}$$

The *prediction vectors* are real values $\mathcal{O}_1'$ and $\mathcal{O}_1$. A *learning situation* is $\mathcal{L}_{n,1} = [\mathcal{P}_n, \mathcal{O}_1] = [\mathcal{P}_n, o_1]$. In this context, the *model prediction distance* $d^{f_j}_{\mathcal{L}_{n,1}}$ is the euclidian distance between a *learning situation* $\mathcal{L}_{n,1}$ and the local linear model. It is the distance between a point with $n+1$ dimensions which is the situation $\mathcal{L}_{n,1} = [\mathcal{P}_n, \mathcal{O}_1]$, and the hyperplane $\mathcal{H}$ representing the linear model f_n^j. The $n+1^{th}$ dimension is the prediction dimension.

$$d^{f_j}_{\mathcal{L}_{n,1}} = \frac{|\overrightarrow{\mathcal{L}_{n,1}\mathcal{H}}.\vec{n}|}{\| \vec{n} \|} = \frac{|\mathcal{O}_1.1 - a_1^j.p_1 - \ldots - a_n^j.p_n - a_0|}{\| 1^2 + {a_1^j}^2 + \ldots + {a_n^j}^2 \|}$$

with $-a_1^j.x_1 - \ldots - a_n^j.x_n + 1.o_1 - a_0 = 0$ the cartesian equation of the plane and $\vec{n} = [-a_1^j, \ldots, -a_n^j, 1]$ a normal vector to the hyperplane $\mathcal{H}$, $\mathcal{O}_1$ the oracle's prediction, $[a_0^j, a_1^j, \ldots, a_n^j]$ the model coefficients and $[p_1, \ldots, p_n]$ the *perceptions* $\mathcal{P}_n$. For other learning models, the distance to a *learning situation* is different.

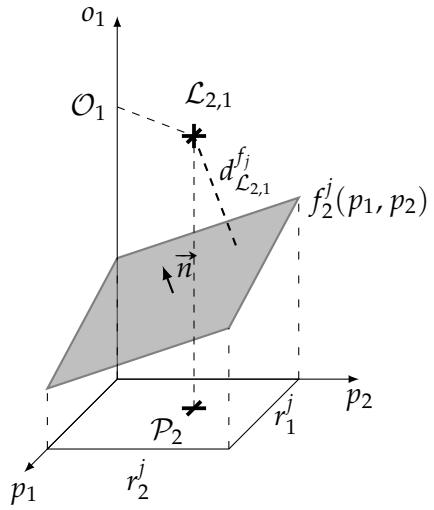


Figure 6.8 – Distance to a *learning situation* with 2 perceptions.

Demonstration. A linear regression of dimension n estimates the coefficients a_i to define the following function depending on the variables p_i. In our case, the result of the function is the prediction value o_1.

$$f(p_1, \ldots, p_i, \ldots, p_n) = a_1.\mathbf{p_1} + \ldots + a_i.\mathbf{p_i} + \ldots + a_n.\mathbf{p_n} + a_0 = \mathbf{o_1} \tag{6.10}$$

The corresponding hyper-plane equation of this function is given by the equation 6.11.

$$\begin{gathered} a_1.\mathbf{p_1} + \ldots + a_i.\mathbf{p_i} + \ldots + a_n.\mathbf{p_n} + a_0 = \mathbf{o_1} \\ -a_1.\mathbf{p_1} - \ldots - a_i.\mathbf{p_i} - \ldots - a_n.\mathbf{p_n} + 1.\mathbf{o_1} - a_0 = 0 \end{gathered} \tag{6.11}$$

with $(-a_1, \ldots, -a_i, \ldots, -a_n, 1)$ a normal vector.

Considering a *learning situation* $\mathcal{L}_{n,1} = [\mathcal{P}_n, \mathcal{O}_1]$ with coordinates $[p_1, \ldots, p_i, \ldots, p_n, \mathcal{O}_1]$ in the frame $\{\mathbf{p_1}, \ldots, \mathbf{p_i}, \ldots, \mathbf{p_n}, \mathbf{o_1}\}$, the distance between the hyper-plane of the regression and the *learning situation* $\mathcal{L}_{n,1}$ is given by the equation 6.12.

$$d_{\mathcal{L}_{n,1},\mathcal{H}} = \frac{|\overrightarrow{\mathcal{L}_{n,1}\mathcal{H}}.\vec{n}|}{\| \vec{n} \|} = \frac{| - a_1.p_1 - \ldots - a_n.p_n + \mathcal{O}_1.1 - a_0|}{\| a_1^{j\,2} + \ldots + a_n^{j\,2} + 1^2 \|} \tag{6.12}$$

Local Model Similarity

The *model similarity distance* d^f_{sim} is another model-dependent distance. It measures the similarity between two *local models*. When the *local models* are linear models, $f_n^{j_1}$ and $f_n^{j_2}$, the coefficients on each dimension $a_i^{j_1}$ and $a_i^{j_2}$ of the regressions are compared to evaluate the similarity. The maximum difference between the coefficient is retained as the *model similarity distance* with linear models $d^{f_{lr}}_{sim}$.

$$d^{f_{lr}}_{sim}(f_n^{j_1}, f_n^{j_2}) = \max_{i=0}^{n} \frac{|a_i^{j_1} - a_i^{j_2}|}{a^{max}} \tag{6.13}$$

With a^{max} the maximum regression coefficient across *Context Agents*. Two local linear models are similar when the *model similarity distance* is lesser than the *model similarity threshold* t^f_{sim}.

$$d^{f_{lr}}_{sim}(f_n^{j_1}, f_n^{j_2}) \; < \; t^f_{sim} \tag{6.14}$$

6.2.3.4 Proximity

The proximity of a *Context Agent* is measured by the *range perceptions proximity distance* $d^{\mathcal{R}^j_n}_{\mathcal{P}_n}$. It is an external Manhattan distance. It represents the distance between the *perceptions* and the *validity ranges* of a *Neighbor Context Agents*. It is the sum of the external distances between each perception p_i and each *validity range* r_i^j.

$$d^{\mathcal{R}^j_n}_{\mathcal{P}_n} = \sum_i \overline{p_i\, r_i^j} \tag{6.15}$$

with $\overline{p_i\, r_i^j} = \min(|p_i - r^j_{i,start}|, |p_i - r^j_{i,end}|)$ if the perception p_i is not included in the *validity range*. Is the perception p_i is included in the *validity range*, the distance $\overline{p_i\, r_i^j}$ is zero.

6.2.4 Synthesis

This section has presented the *learning criticality* and the *exploitation criticality*. These metrics enable to chose the importance of model accuracy or proximity, past experiences and generalization during learning and exploitation cycles. The metrics for all these facets of learning where also detailed. Table 6.6 reminds the parameters involved. The two model-dependent distances which are the *model prediction distance* and the *model similarity distance* were presented in the context of linear regression which is the used learning model of this work.

	Name	Notation	Constrains	Domain
User Metrics	*model prediction distance*	$d^{f_j}_{\mathcal{L}_{n,1}}$	linear regression	$\mathbb{R}$
	model similarity distance	$d^{f_{lr}}_{sim}$	linear regression	$\mathbb{R}$
Designer Paramaters	*accuracy learning weight*	$w^{lrn}_{f_n}$	≥ 0	$\mathbb{R}$
	experience learning weight	$w^{lrn}_{c_{0,1}}$	≥ 0	$\mathbb{R}$
	generalization learning weight	$w^{lrn}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$
	proximity exploitation weight	$w^{expl}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$
	experience exploitation weight	$w^{expl}_{c_{0,1}}$	≥ 0	$\mathbb{R}$
	generalization exploitation weight	$w^{expl}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$
	model similarity threshold	t^{f}_{sim}	> 0	$\mathbb{R}$
Designer Metrics	*normalized confidence*	$c^{j}_{0,1}$	$]0;1[$	$\mathbb{R}$
	validity ranges volume	$\mathcal{V}^{j}_{n}$	> 0	$\mathbb{R}$
	range perceptions proximity distance	$d^{\mathcal{R}^j_n}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$
Designer Criticalities	*learning criticality*	$Crit^{lrn}$	$(w^{lrn}_{f_n}, w^{lrn}_{c_{0,1}}, w^{lrn}_{\mathcal{R}_n})$-dependent	$\mathbb{R}$
	exploitation criticality	$Crit^{expl}$	$(w^{expl}_{\mathcal{P}_n}, w^{expl}_{c_{0,1}}, w^{expl}_{\mathcal{R}_n})$-dependent	$\mathbb{R}$

Table 6.6 – Table of designer parameters, metrics and criticalities involved in learning and exploitation criticalities.

6.3 Lifelong Context Learning

In this section, the updating of the *local model* with *learning situations* is detailed. A lifelong updating mechanism is proposed for updating any underlying learning model embedded in the *Context Agents*. This mechanisms was designed to enable learning models to memorize their past *learning situations* without storing all of them. The main idea is to ask the *local model* for a set of *prediction vectors* that will represent its *local model* so far. These predictions are then mixed with any new *learning situations* so that *local models* learn from past and new *learning situations*. This mechanism is used for learning exogenous and endogenous situations. First, the motivations of this mechanisms are presented. Then the principle of weighted model update is described followed by the necessary generation of artificial *learning situations*. The presentation of the two possible update scenarios finish this contribution.

6.3.1 Motivations

Learning models can require to memorize a large number of *learning situations* if all the history of situations must have an impact on the learning. This brings up different different issues. In a context of *Lifelong Learning*, all *learning situations* can not be stored to avoid memory and time saturation. The more *learning situations* they are, the more the update of a learning model takes time. Old *learning situations* can also be obsolete if the *validity ranges* of a *Context Agent* have changed. Keeping a subset of *learning situations* requires a non-obvious selection strategy. For all of these reasons, updating *local models* with *learning situations* in a lifelong setting is needed. We call it *Lifelong Context Learning*.

6.3.2 Weighted Model Update

The proposed solution is to give a weight to a new *learning situation* so that the modification of a *local model* can be controlled. Let f^{j}_{n,c_l} be the model of a *Context Agent* at a cycle c_l. It represents the resulting model of all the previous adjustments that occurred during the previous c_{l-1} cycles. A new *learning situation* at a cycle c_l has an influence on the *local model* that is represented by its *learning weight* w_{lrn} with $0 < w_{lrn} \leqslant 1$. The *learning weight* represents the learning speed of *local models*.

To satisfy the *learning weight*, artificial *learning situations* are needed. Let $n^{\mathcal{L}^{new}}$ the number of new *learning situations* and $n^{\mathcal{L}^{artificial}}$ the number of artificial *learning situations*. The number of artificial *learning situations* is given by the equation 6.16. Floor values are considered as the number must be an integer.

$$n^{\mathcal{L}^{artificial}} = \lfloor n^{\mathcal{L}^{new}} . \frac{1 - w_{lrn}}{w_{lrn}} \rfloor \tag{6.16}$$

Let $n^{\mathcal{L}^{needed}}$ the total *learning situations* that are needed by a *local model* (i.e. *Model Ambiguity NCS*) and $n^{\mathcal{L}^{total}} = n^{\mathcal{L}^{artificial}} + n^{\mathcal{L}^{new}}$ are the total available *learning situations* to adjust a model for a cycle c_l. If $n^{\mathcal{L}^{total}}$ is less than $n^{\mathcal{L}^{needed}}$, a factor $k \in \mathbb{N}$ is used to increase the number $n^{\mathcal{L}^{total}}$ and satisfy the inequality 6.17.

$$n^{\mathcal{L}^{needed}} \leq k.n^{\mathcal{L}^{total}} \Leftrightarrow n^{\mathcal{L}^{needed}} \leq k.(n^{\mathcal{L}^{artificial}} + n^{\mathcal{L}^{new}}) \tag{6.17}$$

6.3.3 Artificial Learning Situations Generation

To satisfy the *learning weight* during the update of a *local model*, artificial *learning situations* must be generated. The generation of artificial *learning situations* implies the generation of artificial *perceptions* $\mathcal{P}_n^{artificial}$ and artificial *prediction vectors* $\mathcal{O}_m^{artificial}$.

The artificial *perceptions* are created using a normal distribution of as many dimensions as the *perceptions*. The distribution is centered on the considered *Context Agent validity ranges* center : $\mu = r^j_{i,center} = (r^j_{i,start} + r^j_{i,end})/2$.

$$f(p_i) = \frac{1}{\sigma.\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{p_i - r^j_{i,center}}{\sigma}\right)^2} \tag{6.18}$$

The standard deviation σ is set with the empirical rule (equation 6.19).

$$\begin{array}{rcl} \Pr(\mu - k_\sigma\sigma \leq p_i^{artificial} \leq \mu + k_\sigma\sigma) & \approx & 0.6827 \quad \text{with} \quad k_\sigma = 1 \\ \Pr(\mu - k_\sigma\sigma \leq p_i^{artificial} \leq \mu + k_\sigma\sigma) & \approx & 0.9545 \quad \text{with} \quad k_\sigma = 2 \\ \Pr(\mu - k_\sigma\sigma \leq p_i^{artificial} \leq \mu + k_\sigma\sigma) & \approx & 0.9973 \quad \text{with} \quad k_\sigma = 3 \\ \vdots & & \end{array} \tag{6.19}$$

To ensure that artificial *perceptions* are near the centers of the *Context Agents*, σ is set by verifying the equations 6.20. Thus $\sigma = \alpha^{\mathcal{P}_{gen}}.r^j_{i,radius}/k_\sigma$ with $\alpha^{\mathcal{P}_{gen}}$ the *perceptions generation coefficient* that allows to chose the zone of generation depending on the *validity range* radius $r^j_{i,radius} = (r^j_{i,end} - r^j_{i,start})/2$ and k_σ the selected quantile (Fig. 6.9). A *perceptions generation coefficient* of 1 considers the entire *validity range*.

$$\begin{array}{rcl} \mu - k_\sigma\sigma & = & r^j_{i,center} - \alpha^{\mathcal{P}_{gen}} r^j_{i,radius} \\ \mu + k_\sigma\sigma & = & r^j_{i,center} + \alpha^{\mathcal{P}_{gen}} r^j_{i,radius} \end{array} \tag{6.20}$$

The resulting distribution is given in the equation 6.21. It enables to distribute the artificial *perceptions* around the center of the *Context Agents* where they are most confident. Figure 6.9 shows an illustration of the distribution on a one dimension *validity range*.

$$f(p_i) = \frac{1}{\frac{\alpha^{\mathcal{P}_{gen}}.r^j_{i,radius}}{k_\sigma}.\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{p_i - r^j_{i,center}}{\frac{\alpha^{\mathcal{P}_{gen}}.r^j_{i,radius}}{k_\sigma}}\right)^2} \tag{6.21}$$

By fixing k_σ to 3, there is one chance in 370 that an artificial perception $p_i^{artificial}$ is outside the generation interval. Assuming that there is a minimal of one new *learning situation*, this implies that for such a situation to happen, the *learning weight* must be under $1/371 \approx 0.003$, which is very low. Another scenario is that the learning model needs more than 370 to be

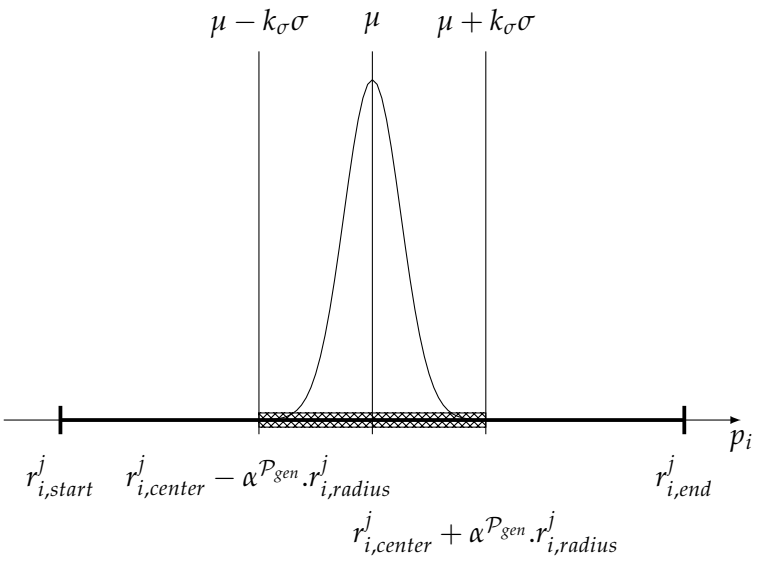


Figure 6.9 – Probability density for artificial *perceptions* generation

reliable. In this study, such models are not considered. If it were the case, k_σ should be fixed to a higher value.

For each artificial generated *perceptions*, an artificial *prediction vector* is asked to the model of the *Context Agent* to complete the artificial *learning situations*. The artificial and the new *learning situations* are then given to the *local model* which is recalculated with past and new experiences. In this thesis, the used learning algorithm is Miller's regression [Miller, 1992]. The consequence of the use of artificial *learning situations* is that when a *Context Agent* receives new *learning situations*, the model can adjust itself choosing the importance of the new experiences with a *learning weight* w_{lrn}. This distribution ensures that the center of the *local models* are slightly altered.

6.3.4 Local Models Update Scenarios

The *local models* update is different if it has been completed or not (cf. sections 5.4.3.1 and 5.4.5.1). These differences are presented in the first instance. In a second stage, it is detailed how *exogenous learning situations* and *endogenous learning situations* are also treated differently.

6.3.4.1 Immature and Mature Models

As presented in the *Model Ambiguity NCS* (section 5.4.3.1), *local models* can be in two different states. When a *local model* has not received sufficient *learning situations*, it is in an incomplete or immature state. When it has received enough, it is in a complete of mature state. This differentiates two kinds of model update.

- ▷ **Incomplete Model Update**. The model was not able to forge a complete representation of its learning so far. Each new *learning situations* are stored and have the same weight

during the construction of the model.

- ▷ **Complete Model Update**. The model has already built a complete representation. Each new *learning situation* has a *learning weight* w_{lrn} to preserve the old model state.

6.3.4.2 Exogenous and Endogenous Learning

ELLSA learning mechanisms can lead to *Exogenous Learning* and *Endogenous Learning*. *Exogenous Learning* concerns *exogenous learning situations* provided one at the time from an external entity of the learning mechanism (i.e. the oracle). *Endogenous Learning* concerns *endogenous learning situations* provided in different quantities. The quantity is defined by the number of *Neighbor Context Agents* during *Cooperative Neighborhood Learning* (section 5.6). Exogenous and endogenous model updates have different *learning weights*.

- ▷ **Exogenous Model Update**. A new *exogenous learning situation* modifies a complete model with an *exogenous learning weight* w_{lrn}^{exo}. If the model is incomplete, the *learning situation* is exploited as previously explained.
- ▷ **Endogenous Model Update**. A set of *endogenous learning situations* updates a complete model with an *endogenous learning weight* w_{lrn}^{endo}. The weight concerns the entire set and not only one *learning situation* like for *exogenous learning situations*. *Endogenous learning situations* do not update incomplete models.

The *exogenous learning weight* and the *endogenous learning weight* enable to select the impact of *Exogenous Learning* and *Endogenous Learning*.

6.3.5 Synthesis

The section presented *Lifelong Context Learning*, a mechanism allowing to update learning models through lifelong new *learning situations*. In particular, it permits not to store all the experienced *learning situations*, only the first ones.

Table 6.7 reminds the involved parameters. They are the weights of the exogenous and endogenous *learning situations*. The *perceptions generation coefficient* represents the portion of the *Context Agents validity ranges* that are exploited when artificial *learning situations* are generated to fulfill the desired learning weights.

	Name	Notation	Constrains	Domain
User Parameters	*exogenous learning weight*	w_{lrn}^{exo}	$]0;1]$	$\mathbb{R}$
	endogenous learning weight	w_{lrn}^{endo}	$]0;1]$	$\mathbb{R}$
Designer Parameter	*perceptions generation coefficient*	$\alpha^{\mathcal{P}_{gen}}$	> 0	$\mathbb{R}$

Table 6.7 – Table of designer parameters involved in *Lifelong Context Learning*.

6.4 Learning Strategies

This section presents the two learning strategies that can be followed with *ELLSA*. The *Active Learning Strategy* and the *Self-Learning Strategy*. The *Active Learning Strategy* assumes that any new *learning situation* can be actively requested to an oracle. *Learning inaccuracies* can be resolved by asking specific situations. The *Self-Learning Strategy* assumes that the learner passively receives *learning situations* from an oracle. *Learning inaccuracies* can only be resolved with internal self-*exploitation situations*.

6.4.1 Active Learning

The *Active Learning Strategy* is based on the availability of *active learning situations* that can be asked to the oracle anytime and anywhere in the search space. In opposition to *passive learning situations*, *active learning situations* are defined as follows.

- ▷ ***Active learning situations*** $\mathcal{L}^{act}_{n,m}$ are composed of *endogenous perceptions* $\mathcal{P}^{endo}_{n}$ and an *exogenous prediction vector* $\mathcal{O}^{exo}_{m}$: $\mathcal{L}^{act}_{n,m} = [\mathcal{P}^{endo}_{n}, \mathcal{O}^{exo}_{m}]$. The *endogenous perceptions* are generated by *ELLSA*. The *exogenous prediction vector* is provided by the oracle.
- ▷ ***Passive learning situations*** $\mathcal{L}^{pass}_{n,m}$ are composed of *exogenous perceptions* $\mathcal{P}^{exo}_{n}$ and an *exogenous prediction vector* $\mathcal{O}^{exo}_{m}$: $\mathcal{L}^{pass}_{n,m} = [\mathcal{P}^{exo}_{n}, \mathcal{O}^{exo}_{m}]$. *Passive learning situations* are *exogenous learning situations* with *exogenous perceptions*.

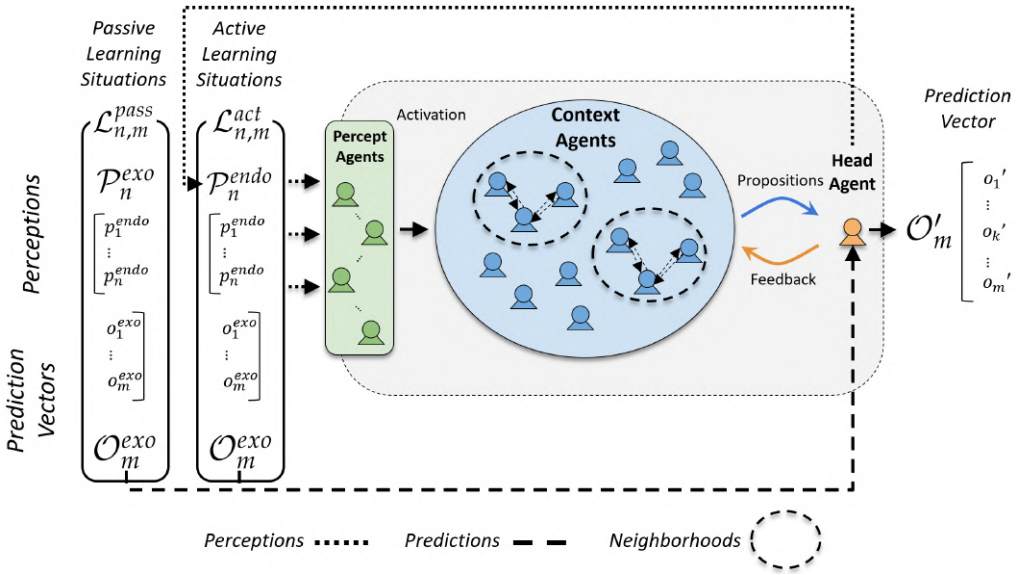


Figure 6.10 – *Active Learning Strategy*

Figure 6.10 shows the general view of the agents communications during the *Active Learning Strategy*. During this strategy, *learning inaccuracies* are resolved with only learning cycles (presented section 6.1.2). In the *Active Learning Strategy* setting, the *learning inaccuracies* detections are processed at the end of each learning cycle. The goal of the *learning inaccuracies* detections is to generate *endogenous perceptions* for future *active learning situations*. All *learning inaccuracies* resolutions are made with active learning cycles for which the

oracle provides an *exogenous prediction vector* associated to the actively requested *endogenous perceptions*. All *learning inaccuracy NCS* resulting behaviors for the *Active Learning Strategy* are detailed in the following.

- The *Model Ambiguity NCS* detection intentionally targets random *endogenous perceptions* inside incomplete *Context Agents local models*. On a future learning cycle, the *endogenous perceptions* are actively requested to the oracle. The targeted incomplete *local models* receive new *active learning situations* to complete their model during the feedback of *Valid Context Agents*.
- The *Conflict NCS* and *Concurrency NCS* detection generate *endogenous perceptions* in the centers of the overlapping areas. In this case, the *endogenous perceptions* provide *active learning situations* that enlighten which *Context Agent* must win the *Conflict NCS* or the *Concurrency NCS*. During learning cycles, its the *Best Context Agent* that is selected with the *learning criticality*. These overlapping areas are then removed during the *Bad Prediction NCS*, the *Conflict NCS* and the *Concurrency NCS* resolutions.
- The *Incompetence NCS* detection creates *endogenous perceptions* associated with *incompetent volumes*. The *active learning situations* will trigger *Unproductivity NCS* with already known *validity ranges* initializations (i.e. the *incompetent volumes*). This allows to actively explore the space of the *perceptions* without adding new *Conflict NCS* and *Concurrency NCS* during the creation of the new *Context Agents*.
- The *Range Ambiguity NCS* detection creates *endogenous perceptions* next to adjacent *Context Agents* with different *local models*. The associated *active learning situations* will trigger good *local model* feedback if a *validity range* border is correct. If a border is not correct, it will trigger *Bad Prediction NCS* that will adjust the *validity range*.

The *Cooperative Neighborhood Learning* mechanism is not used in the *Active Learning Strategy*. Full trust is given to the oracle. This strategy can suffer from the fact that in real world learning scenarios, it is no always possible to request specific situations. Moreover, if the *exogenous learning situations* are provided from the exploration of an environment or robot states, some situations may be unreachable. The *Active Learning Strategy* is for learning scenarios where the *perceptions* space is completely reachable or partially reachable. In other case, the *Self-Learning Strategy* can be used.

6.4.2 Self-Learning

The *Self-Learning Strategy* is based on the non-availability of *active learning situations* and on learning data saving. Only *passive learning situations* are provided to the learning mechanisms. The goal of the *Self-Learning Strategy* is to learn with as few *learning situations* as possible. To do so, *ELLSA* uses *passive learning situations* and *endogenous exploitation situations* to learn.

- ▷ ***Passive learning situations*** $\mathcal{L}_{n,m}^{pass}$ are composed of *exogenous perceptions* $\mathcal{P}_n^{exo}$ and an *exogenous prediction vector* $\mathcal{O}_m^{exo}$: $\mathcal{L}_{n,m}^{pass} = [\mathcal{P}_n^{exo}, \mathcal{O}_m^{exo}]$.
- ▷ ***Endogenous exploitation situations*** $\mathcal{E}_n^{endo}$ are *endogenous perceptions* provided to *ELLSA* for an exploitation cycle: $\mathcal{E}_n^{endo} = [\mathcal{P}_n^{endo}]$. They are self-*exploitation situations*.

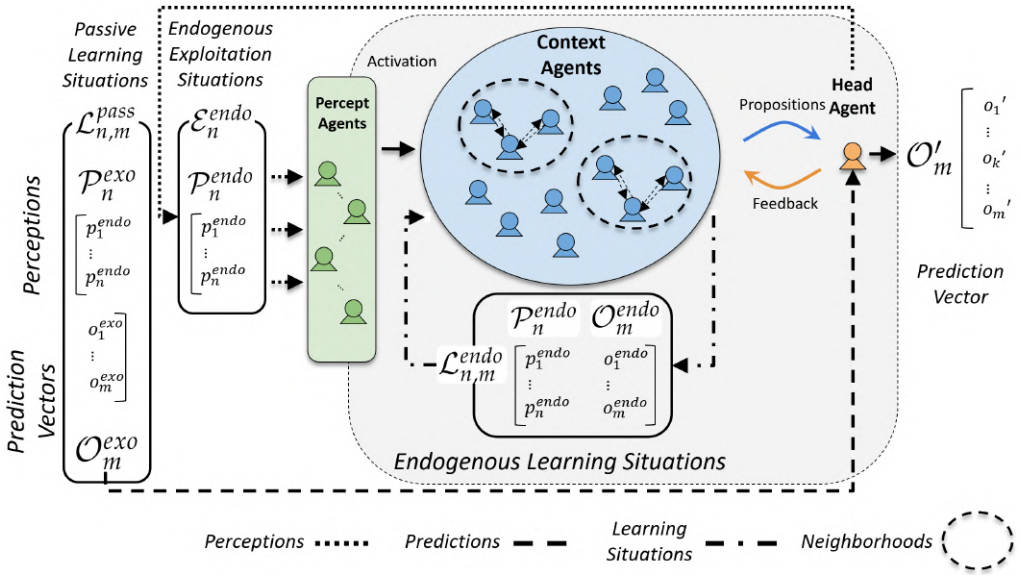


Figure 6.11 – *Self-Learning Strategy*

Figure 6.11 shows the general view of the agents communications during the *Self-Learning Strategy*. During this strategy, *learning inaccuracies* are resolved with learning and exploitation cycles (presented sections 6.1.2 and 6.1.3). In the *Self-Learning Strategy* setting, the *learning inaccuracies* detections are processed at the end of learning and exploitation cycles. The goal of the *learning inaccuracies* detections is to generate *endogenous perceptions* for future *endogenous exploitation situations*. Unlike the *Active Learning Strategy*, the *Self-Learning Strategy* does not use new *exogenous prediction vectors* to resolve the detected *learning inaccuracies*. All detected *learning inaccuracies* resolutions are made with only exploitation cycles for which the *perceptions* are actively requested by the *endogenous perceptions*. All *learning inaccuracy NCS* resulting behaviors for the *Self-Learning Strategy* are detailed in the following.

- ▷ The *Model Ambiguity NCS* detection targets random *endogenous perceptions* inside incomplete *Context Agents local models* to generate new *learning situations*. As new *exogenous prediction vectors* are not available with this strategy, *prediction vectors* are requested to the incomplete model and they serve as *endogenous prediction vectors*.

- ▷ The *Conflict NCS* and *Concurrency NCS* detection generate *endogenous perceptions* in the centers of the overlapping areas. The *endogenous perceptions* lead to an *endogenous exploitation situation* that activates the overlapping *Context Agents*. The *exploitation criticality* is used to determine which *Best Context Agent* wins the *Conflict NCS* or *Concurrency NCS*. The overlapping areas are then removed during the *Conflict NCS* and the *Concurrency NCS* resolutions.
- ▷ The *Incompetence NCS* detection creates *endogenous perceptions* associated with *incompetent volumes* when the numbers of *Neighbor Context Agents* are greater than the *creation neighbors number*. The *endogenous exploitation situations* trigger *Unproductivity NCS* if the numbers of *Neighbor Context Agents* in the new *neighborhoods* are also greater than the *creation neighbors number*. In the *Self-Learning Strategy* setting, a minimal number of *Neighbor Context Agents* is needed both for the detection and the resolution of *Incompetence NCS*. This mechanism is not intended for exploration but rather for filling unexplored *perceptions* that are close to other *Context Agents*. As detailed in section 6.1.3, an *endogenous prediction vector* is created with the predictions of the *Neighbor Context Agents*.
- ▷ The *Range Ambiguity NCS* detection creates *endogenous perceptions* next to adjacent *Context Agents* with different *local models*. As no *exogenous learning situation* is requested in this strategy, it is the role of the *CNL* mechanism to enhance the discontinuities where they are detected. As detailed in the section 5.6, *Prediction Neighbor Context Agents* provide *endogenous learning situations* to the *Best Context Agent* of the current cycle. The sharing of *learning situations* between *Prediction Neighbor Context Agents* allows to smooth the *local models* when continuity is assumed. When the *Neighbor Context Agents* are not in the *prediction neighborhood*, the *endogenous learning situations* are not shared an the assumed discontinuity is represented by the *Context Agents*.

As previously said, the *Self-Learning Strategy* is intended for learning with less *exogenous learning situations* as the *Neighbor Context Agents* are used to provide additional *endogenous learning situations*. Moreover, as the *Self-Learning Strategy* can also be performed with only exploitation cycles, the learning mechanisms can keep enhancing its models even when the learning phase is over. A classical passive exploitation phase thus become an active exploitation phase.

6.4.3 Synthesis

In this section, a focus was made on two different learning strategies with different scopes: the *Active Learning Strategy* and the *Self-Learning Strategy*.

The *Active Learning Strategy* is intended for learning scenarios for which specific new *learning situations* can be requested to an oracle. In the *Active Learning Strategy*, only learning cycles are used to detect and solve *learning inaccuracies*. The detection generates *active learning situations* to speed up the learning process.

The *Self-Learning Strategy* is intended for learning scenarios where the goal is to minimize *learning situations* without requesting additional and specific new *active learning situations*. In the *Self-Learning Strategy* learning and exploitation cycles are used. The learning cycles are for *passive learning situation* and the detection of *learning inaccuracies*. The exploitation cycles are for the *endogenous exploitation situations* that solve the *learning inaccuracies*. The exploitation cycles can also detect *learning inaccuracies*. It is the principle of active exploitation. It is the use of the *Self-Learning Strategy* on an exploitation phase where the learning mechanism can keep seeking for *learning inaccuracies* and enhancing its models while it is being exploited.

6.5 Learning Reflections

In this section, reflections on *ELLSA* according to *Transfer Learning*, *Multi-Task Learning* and *Reinforcement Learning* are made. Learning scenarios in relation to these paradigms are presented. Working mechanisms and future leads are proposed.

6.5.1 Context Transfer Learning

As presented in the state of the art in section 3.1, *Transfer Learning* is an important concern in *AI*. Knowledge generalization and transfer is a challenge that is attracting a lot of attention in *Machine Learning*. One of the objectives of *ELLSA* is to perform *Transfer Learning*. We use the *Transfer Learning* framework of [Pan and Yang, 2010] to introduce *Context Transfer Learning* divided into *Context Domain Adaptation* and *Context Inductive Transfer*.

6.5.1.1 Context Domain Adaptation

Domain adaptation in *Transfer Learning* corresponds to transferring the knowledge of a same task $\mathcal{T}$ between two different domains that are the source domain $\mathcal{D}_S$ and the target domain $\mathcal{D}_T$. In *Context Learning*, a task is represented by a set of *Context Agents*, the knowledge of the task is distributed in the *local models*. The domains are the set of *perceptions*. The two different domains are different *perceptions* in quantity and/or in nature. Figure 6.12 shows an example of domain adaptation with a robot task. The learned task could be a delivery or surveillance trajectory to follow. With two *perceptions*, the task would be performed by a terrestrial robot and with three *perceptions*, it would be an aerial robot.

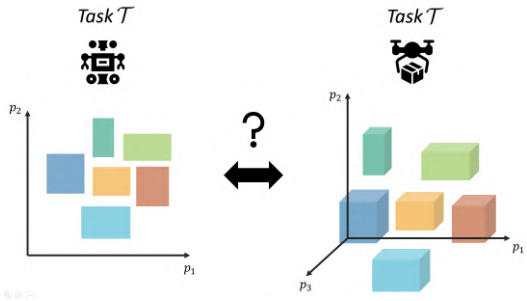


Figure 6.12 – Context Domain Adaptation

Assuming that the *perceptions* are of the same nature (the position in space for example), depending on the target and source domain, two cases appear.

- $\mathcal{D}_S = \{p_1, p_2, p_3\}$ and $\mathcal{D}_T = \{p_1, p_2\}$. The domain adaptation is a reduction of the space. Updating the *validity ranges* is easy as only the target *perceptions* are kept. Updating the *local models* would require to question the predictions of the 3-dimensional *Context Agents* to generate artificial *learning situations* (section 6.3) for the 2-dimensional *Context Agents local models*.
- $\mathcal{D}_S = \{p_1, p_2\}$ and $\mathcal{D}_T = \{p_1, p_2, p_3\}$. The domain adaption is an increase of the space. In this case the *validity ranges* of the new perception needs to be generated with default

values or with the averages of source domain *validity ranges*. The *local models* can then be updated with the same method than the domain reduction.

If the *perceptions* are not from the same nature, the state of wheels and rotors for example, additional mechanisms would be needed to transfer the *validity ranges* and *local models*. *Endogenous Context Learning* seems appropriate for domain adaption as it offers great *Context Agents* malleability. The *validity ranges* and *local models* are easily exploitable and provide great evolutionary properties.

6.5.1.2 Context Inductive Transfer

Inductive Transfer in *Transfer Learning* corresponds to transferring the knowledge of a same domain $\mathcal{D}$ between two different but related tasks that are the source tasks $\mathcal{T}_S$ and the target task $\mathcal{T}_T$. As we said, in *Context Learning*, a task is represented by a set of *Context Agents* where the knowledge of the task is distributed in several *local models*. The unique domain is the set of *perceptions*. The two tasks are learned with two training sets of *learning situations*. The first is larger than the second as the second learning must transfer already known experiences. Figure 6.13 shows an example of inductive learning with robot tasks in a domain with two *perceptions* p_1 and p_2. Keeping the example of a trajectory task, the goal in this case is to learn the second trajectory that is the target task with fewer *learning situations*.

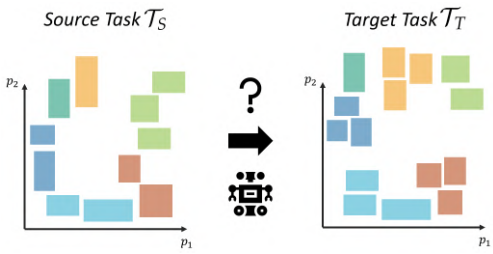


Figure 6.13 – Context Inductive Transfer

The learning of the second task should exploit already learned models to save *learning situations* and reuse the *Context Agents* that are also reliable for the new task. This is exactly what *Endogenous Context Learning* does when it creates new *Context Agents*. Experiments in order to evaluate the transfer properties of *ELLSA* will be conducted section 7.9.

6.5.2 Context Multi-Task Learning

Multi-Task Learning, presented section 3.2, is another challenge of *AI*. The *Multi-Task Learning* challenge is to generalize knowledge across several related learned tasks and use this knowledge to achieve better performances on each task. Figure 6.14 shows an example of several tasks $\mathcal{T}_1$,$\mathcal{T}_2$ and $\mathcal{T}_3$ represented by *Context Agents* with two *perceptions*. The 3 tasks are different but still related.

In this work, tasks are not represented in a symbolic manner. If a task $\mathcal{T}_1$ is learned, the second learned task $\mathcal{T}_2$ will transfer the usable *local models* but conflicting models with the

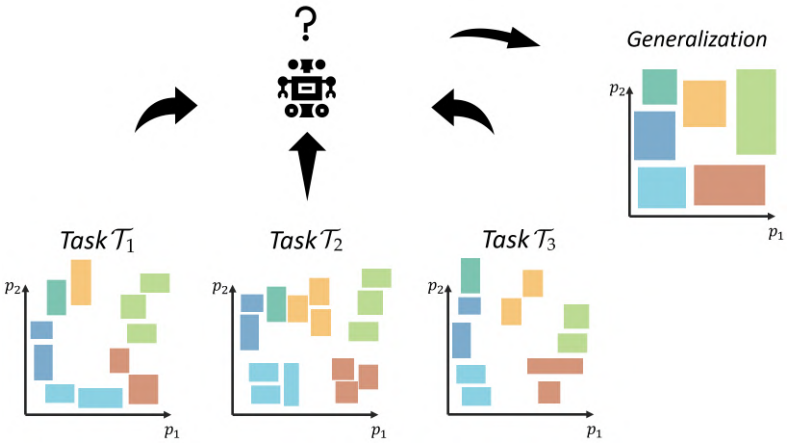


Figure 6.14 – Context Multi-Task Learning

new task will be forgotten. A first step would be to create one collective of *Context Agents* by task. Previously learned tasks could be used during the learning of the new tasks but they would be kept intact. This mechanism would enable to learn several tasks and share common knowledge about them during the learning of new ones. However an issue could be the scalability of such implementation. If the number of tasks is large, all of them cannot be stored.

Generalization across learned tasks would enable to tackle this problem. All tasks could be merged into a high level collective of *Context Agents* which would represent a generalization of all learned related tasks. To go further, a measure of similarity between learned tasks (i.e. sets of *Context Agents*) would need to be introduced in order to detect which tasks should be merged into generalized knowledge and which should not.

6.5.3 Context Reinforcement Learning

Reinforcement Learning, presented section 2.4, is a widely use approach in *AI* to explore and learn from an environment by acting and interacting with it. The discovery of the environment is made thanks to a reward that is given when positives situations are encountered. *Endogenous Context Learning* provides tracks to explore using the *Reinforcement Learning* paradigm. Figure 6.15 shows a simple example of a *Reinforcement Learning* problem where a robot has to reach certain areas (green areas) and avoid other ones (red areas) in a 2D space represented by the *perceptions* p_1 and p_2.

The goal of classical *Reinforcement Learning* is to generate a policy from the experienced rewards so that the positive situations can be reached in an optimal way. What can be done with *Endogenous Context Learning* is to learn the reward function with *Context Agents* and then to convert the collective of *Context Agents* into a policy. The learned rewards only represent the situations that are positive (green), negative (red) and neutral (gray). But the positive situations cannot be reached and the negative situations cannot be avoided as the neutral rewards do not provide any better or worse directions. To generate the policy from a learned reward function, the *Context Agents* should smooth the rewards predictions across all the collective of agents. As introduced with the *CNL* mechanism, *Context Agents* could

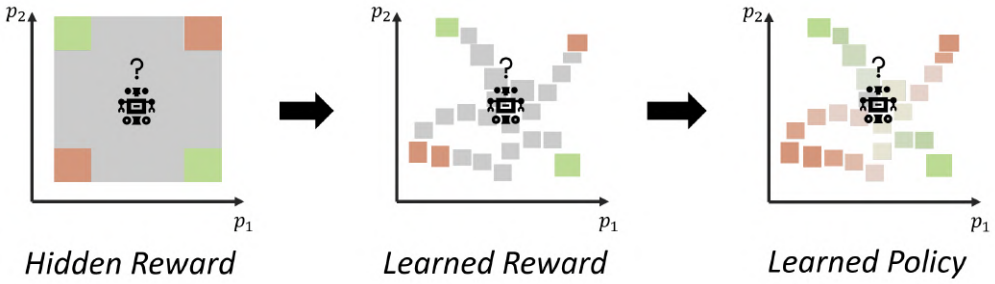


Figure 6.15 – Context Reinforcement Learning

share their predictions to their neighbors to generate a reward gradient. The reward gradient would then be the learned policy enabling to point out which direction to follow.

Coupled with the learned policy, a controller permitting to move the learner (the robot in this example) in the space of the learned policy would be needed. The learned policy would provide the closest *perceptions* with the highest reward. The controller would provide the actions to reach the *perceptions*.

This approach would allow to perform *Reinforcement Learning* in continuous state and action spaces which is not the case in most *Reinforcement Learning* approaches [Montazeri et al., 2011]. Recent works focuses on continuous state and action space *Reinforcement Learning* as it is better suited for real world applications [Fernandez-Gauna et al., 2018].

6.6 Synthesis

This section presents a synthesis of the two contributions chapters. First, conclusions are provided for the *Endogenous Learning* principles and the *Endogenous Lifelong Learner by Self-Adaptation* (*ELLSA*). Then, a summary of all parameters and distances is reminded.

6.6.1 Endogenous Learning Principles

The chapter on the *Endogenous Learning* principles resulted in the introduction of several mechanisms which were introduced in several stages.

- **Definitions and Objectives**. *Endogenous Learning* was defined and its objectives were presented in terms of general learning hypotheses. The hypotheses were declined into *Context Learning* oriented hypotheses. The objectives enabled to clearly identified the possible shortcomings during *Context Learning*.
- **Neighborhood**. To overcome deficiencies, a first step was to design neighborhood mechanisms to add local interactions between the fragments of knowledge that are the *Context Agents*. The *neighborhood* and *influences* enable local sets of *Context Agents* to communicate and exchange information without considering all the *Context Agents* of the learning mechanism. Thus, unnecessary operations can be avoided. The locality principle provides a gain in computation time and memory distribution properties.
- **Learning Inaccuracies**. The shortcomings of *Context Learning* were identified as *learning inaccuracies*. They are divided into exploration *learning inaccuracies* that concern the *Context Agents validity ranges* and model *learning inaccuracies* that concern the *Context Agents local models*. Thanks to the addition of *neighborhood* and *influences*, the detection and resolution of *learning inaccuracies* are detailed. The resolution scheduling strategy is presented with the establishment of priorities to each *learning inaccuracies*.
- **Cooperative Neighborhood Learning**. A last mechanisms entitled *Cooperative Neighborhood Learning* is presented. Still benefiting from the *neighborhood* and *influences* mechanisms, a way to generate *endogenous learning situations* is exhibited. *Neighbor Context Agents* share their own model predictions to reach continuity between their *local models* borders.

6.6.2 ELLSA

The chapter on the *Endogenous Lifelong Learner by Self-Adaptation* (*ELLSA*) detailed the integration of the *Endogenous Learning* principles into a larger *Multi-Agent System* using several learning techniques and strategies. It involved the description of the execution cycles, the introduction of learning and exploitation criticalities, the implementation of a *Lifelong Learning* mechanism for *Context Learning*, the definition of two learning strategies and a reflection and positioning in relation to other learning paradigms.

- **Execution cycles**. The sequence of an execution cycle of the learning mechanism *ELLSA* is outlined from the *perceptions* input to the *output prediction vector*. The other agents

constituting the learning mechanism are detailed. The *Percept Agents* are responsible for optimizing the activation of the *Context Agents*. The *Head Agent* selects the best *Context Agent* with different metrics for the prediction output. Learning and exploitation cycles are differentiated by the availability of *learning situations*. The contribution concerning the execution cycles is firstly the optimization of the *Context Agents* and *Neighbor Context Agents* activations. Another enhancement is the addition of neighborhood mechanisms into *Context Learning*.

- **Criticalities**. The *learning criticality* and the *exploitation criticality* represent the reliability of the *Context Agents*. They rely on metrics of generalization, experience, performance and proximity. The criticalities are configurable so that the importance of the metrics can be chosen. This section brings a formalization of generalization, experience and performance in the *Context Learning* paradigm when only performance was initially used.
- **Lifelong Learning**. A mechanism was introduced to learn in a lifelong setting. It enables to update any learning model without storing all the *learning situations*. To do so, when a *local model* receives a new *learning situation*, artificial *learning situations* are generated with the *local model* that will be updated. The artificial *learning situations* represent past learning. The new *learning situation* is combined with the artificial *learning situations* to merge past and new experiences. *Lifelong Context Learning* main contribution is the possibility to control the learning weight of any new *learning situation* on any learning model without being constrained by it.
- **Learning Strategies**. Two different learning strategies where presented: the *Active Learning Strategy* and the *Self-Learning Strategy*. The *Active Learning Strategy* relies on the availability of specific *learning situations* that can be actively requested by the learner. The *Self-Learning Strategy* focuses on using as little *learning situations* as possible while generating *endogenous learning situations* to counterbalance the lack of information. The strategies involve different ways of resolving the *learning inaccuracies*. These two strategies provide active and self generation of knowledge when only passive learning was performed in classical *Context Learning*.
- **Reflections**. To finish, a reflection was made concerning how *Endogenous Context Learning* is positioned in relation to some challenges *AI* is facing nowadays. The focus is made on *Transfer Learning*, *Multi-Task Learning* and *Reinforcement Learning*. Concerning *Transfer Learning*, *Endogenous Context Learning* offers interesting leads on task transfer that would need to be assessed in practical cases. About *Multi-Task Learning*, additional work would be needed to address this problem as several tasks are represented by several collectives of *Context Agents* and *ELLSA* only considers one set of *Context Agents*. Regarding *Reinforcement Learning*, *Endogenous Context Learning* brings promising directions as it provides continuous state and action spaces when most of *Reinforcement Learning* approaches use discrete spaces.

6.6.3 Parameters, Distances, Metrics and Criticalities

The table 6.8 summaries all the parameters, distances, metrics and criticalities introduced in the contributions chapters. They are differentiated between user and designer use.

		Name	Notation	Constrains	Domain
User	Parameters	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$
		model error margin	m^{f}_{err}	>0	$\mathbb{R}$
		bootstrap cycles number	c_{boot}	$\geqslant n$	$\mathbb{N}$
		exogenous learning weight	w^{exo}_{lrn}	$]0;1]$	$\mathbb{R}$
		endogenous learning weight	w^{endo}_{lrn}	$]0;1]$	$\mathbb{R}$
		discontinuity detection probability	pb^{disc}	$]0;1[$	$\mathbb{R}$
	Distance	*range creation radius*	$r_i^{creation}$	($p^{\mathcal{R}}$)-dependent	$\mathbb{R}$
	Metrics	*model prediction distance*	$d^{f_j}_{\mathcal{L}_{n,m}}$	model-dependent	$\mathbb{R}$
		model similarity distance	d^{f}_{sim}	model-dependent	$\mathbb{R}$
Designer	Parameters	*neighborhood radius coefficient*	$\alpha^{\mathcal{N}}$	>0	$\mathbb{R}$
		influence radius coefficient	$\alpha^{\mathcal{I}}$	>0	$\mathbb{R}$
		maximum range radius coefficient	$\alpha^{\mathcal{R}_{max}}$	>1	$\mathbb{R}$
		range similarity coefficient	$\alpha^{\mathcal{R}_{sim}}$	$]0,1[$	$\mathbb{R}$
		minimum range coefficient	$\alpha^{\mathcal{R}_{min}}$	$]0,1[, \alpha^{\mathcal{R}_{min}} < \alpha^{\mathcal{R}_{sim}}$	$\mathbb{R}$
		creation neighbors number	$n^{creation}$	>0	$\mathbb{N}$
		perceptions generation coefficient	$\alpha^{\mathcal{P}_{gen}}$	>0	$\mathbb{R}$
		model similarity threshold	t^{f}_{sim}	≥ 0	$\mathbb{R}$
		accuracy learning weight	$w^{lrn}_{f_n}$	≥ 0	$\mathbb{R}$
		experience learning weight	$w^{lrn}_{c_{0,1}}$	≥ 0	$\mathbb{R}$
		generalization learning weight	$w^{lrn}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$
		proximity exploitation weight	$w^{expl}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$
		experience exploitation weight	$w^{expl}_{c_{0,1}}$	≥ 0	$\mathbb{R}$
		generalization exploitation weight	$w^{expl}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$
	Distances	*neighborhood radius*	$r_i^{\mathcal{N}}$	($p^{\mathcal{R}}$,$\alpha^{\mathcal{N}}$)-dependent	$\mathbb{R}$
		prediction neighborhood radius	$r^{\mathcal{N}}_{o_k}$	($p^{\mathcal{R}}$,$\alpha^{\mathcal{N}}$)-dependent	$\mathbb{R}$
		influence radius	$r^{\mathcal{I}}_{j,i}$	($\alpha^{\mathcal{I}}$)-dependent	$\mathbb{R}$
		maximum range radius	r_i^{max}	($p^{\mathcal{R}}$,$\alpha^{\mathcal{R}_{max}}$)-dependent	$\mathbb{R}$
		range similarity distance	$d_i^{\mathcal{R}_{sim}}$	($p^{\mathcal{R}}$,$\alpha^{\mathcal{R}_{sim}}$)-dependent	$\mathbb{R}$
		minimum range distance	$d_i^{\mathcal{R}_{min}}$	($p^{\mathcal{R}}$,$\alpha^{\mathcal{R}_{min}}$)-dependent	$\mathbb{R}$
	Metrics	*normalized confidence*	$c^{j}_{0,1}$	$]0;1[$	$\mathbb{R}$
		validity ranges volume	$\mathcal{V}_n^j$	>0	$\mathbb{R}$
		range perceptions proximity distance	$d^{\mathcal{R}^j_n}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$
	Criticalities	*learning criticality*	$Crit^{lrn}$	($w^{lrn}_{f_n}$,$w^{lrn}_{c_{0,1}}$,$w^{lrn}_{\mathcal{R}_n}$)-dependent	$\mathbb{R}$
		exploitation criticality	$Crit^{expl}$	($w^{expl}_{\mathcal{P}_n}$,$w^{expl}_{c_{0,1}}$,$w^{expl}_{\mathcal{R}_n}$)-dependent	$\mathbb{R}$

Table 6.8 – Table of parameters, distances, metrics and criticalities involved in *Endogenous Context Learning*.

The user parameters enable to select the *validity ranges* and *local models* precision. They also allow to select how many cycles must be used to boot the learning. The user can also choose the weight of the *exogenous learning situations* and the *endogenous learning situations*; and the probability of discontinuity detection.

The user metrics are the measures of model affinity with *learning situations* and model

similarity. They depend on the learning model that is implemented by the user.

The designer parameters enable to configure the *neighborhood* and the *influences*, the detection and resolution of *learning inaccuracies*, and the learning and exploitation mechanisms.

The designer distances represent the distances that are manipulated to determine the *neighborhood*, the *influences* and all the mechanisms that modify the *validity ranges*.

The designer metrics are all the measures that represent the *Context Agents* properties according model experience, generalization and proximity to situations (*learning situations* or *exploitation situations*).

The criticalities enable to assess the reliability of the *Context Agents* according to all the presented metrics.

The next chapter will present experimentations on abstract hidden functions. The sensibility of most of the presented parameters will be examined. The learning will be tested on several scenarios by measuring metrics related to the objectives of this thesis (section 1.6).

Experimentations and Validation

7 Experiments on Mathematical Models

In this chapter, several experimentations made on the learning mechanism ELLSA are presented. The learned models are mathematical functions with properties of linearity, non linearity, continuity and discontinuity. These experimentations allow to highlight the strengths and weaknesses of the system according to the objectives presented in section 1.6 (Endogenous Feedback, Agnosticity, Lifelong Learning, Online Learning, Self-Observation, Knowledge Generalization, Scalability, Any Data Amount and Explainability).

THE experiments conducted in this chapter focus on the learning of abstract models in order to highlight specific *ELLSA* properties. First, the used parameters during the experiments are presented. Some of them vary during the specific experiment scenarios. In each scenario, the changing parameters are specified. Then, to evaluate the objectives of this thesis, several metrics are introduced. They are measured all along the experimentations. The experiments focus on the following matters:

— **Learning Inaccuracies**. A toy learning problem is presented in order to assess the detection and resolution of *learning inaccuracies NCS* (Sec. 7.4).

— **Active Learning and Self-learning**. The *Active Learning Strategy* and the *Self-Learning Strategy* are tested on non linear continuous and discontinuous learning problems. A multi-model learning problem is also presented (Sec. 7.5).

— **Lifelong Learning**. This section details an experiment which shows how the *Lifelong Context Learning* mechanism deals with noise in the learning data. A second experiment presents how a lifelong exploitation can enable to keep enhancing the learned models (Sec. 7.6).

— **Few Learning Situations**. An experiment which evaluates the ability of *ELLSA* to learn with few *learning situations* is proposed (Sec. 7.7).

— **Scalability**. The scalability of the learning mechanisms is tested in relation to the number of *Context Agents* and *perceptions* (Sec. 7.8).

— **Transfer Learning**. A last experiment is presented to test the *Transfer Learning* abilities of *ELLSA* (Sec. 7.9).

7.1 Parameters

The table 7.1 reminds all the parameters introduced in the previous chapters and adds experimental parameters. For illustration and explanation needs, most experiments are conducted with 2 *perceptions*. Section 7.8 assesses how higher numbers of *perceptions* impact the learning mechanisms. A learning episode is composed of a learning phase with a certain *number of learning situations* $\mathcal{L}_{\#}$ and an exploitation phase with a certain *number of exploitation situations* $\mathcal{E}_{\#}$. To quantify the learning performances, several metrics are calculated (section 7.2) and averaged over 15 learning episodes. More than 15 learning episodes provide equally stable behaviors and are thus not necessary. For all experiments, the code is implemented in java with the framework AMAK [Perles et al., 2018] and it is executed on a machine [1] with Ubuntu 18.04.3 LTS.

The sensibility of several parameters is assessed all along the learning scenarios that are specified in the table 7.1.

	Name	Notation	Constrains	Domain	Value	Sensibility
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2	7.8
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	500	7.8
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	250	7.6.2
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15	
User	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$	0.1	7.4, 7.6.1, 7.6.2, 7.7
	model error margin	m^{f}_{err}	> 0	$\mathbb{R}$	1	7.5.1
	bootstrap cycles number	c_{boot}	$\geqslant n$	$\mathbb{N}$	10	
	exogenous learning weight	w^{exo}_{lrn}	$]0;1]$	$\mathbb{R}$	0.1	7.7
	endogenous learning weight	w^{endo}_{lrn}	$]0;1]$	$\mathbb{R}$	0.1	7.6.2, 7.7
	discontinuity detection probability	pb^{disc}	$]0;1[$	$\mathbb{R}$	0.1	7.5.1
Designer	*neighborhood radius coefficient*	$\alpha^{\mathcal{N}}$	> 0	$\mathbb{R}$	2	7.6.2, 7.7
	influence radius coefficient	$\alpha^{\mathcal{I}}$	> 0	$\mathbb{R}$	0.5	7.6.2, 7.7
	maximum range radius coefficient	$\alpha^{\mathcal{R}_{max}}$	> 1	$\mathbb{R}$	2	
	range similarity coefficient	$\alpha^{\mathcal{R}_{sim}}$	$]0,1[$	$\mathbb{R}$	0.375	
	minimum range coefficient	$\alpha^{\mathcal{R}_{min}}$	$]0,1[, \alpha^{\mathcal{R}_{min}} < \alpha^{\mathcal{R}_{sim}}$	$\mathbb{R}$	0.25	
	creation neighbors number	$n^{creation}$	> 0	$\mathbb{N}$	7	
	perceptions generation coefficient	$\alpha^{\mathcal{P}_{gen}}$	> 0	$\mathbb{R}$	0.1	7.6.1
	model similarity threshold	t^{f}_{sim}	≥ 0	$\mathbb{R}$	0.001	7.5.2
	accuracy learning weight	$w^{lrn}_{f_n}$	≥ 0	$\mathbb{R}$	1	7.4
	experience learning weight	$w^{lrn}_{c_{0,1}}$	≥ 0	$\mathbb{R}$	1	7.4
	generalization learning weight	$w^{lrn}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$	1	7.4
	proximity exploitation weight	$w^{expl}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$	1	7.4
	experience exploitation weight	$w^{expl}_{c_{0,1}}$	≥ 0	$\mathbb{R}$	1	7.4
	generalization exploitation weight	$w^{expl}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$	1	7.4

Table 7.1 – Table of experimental, user and designer parameters. If not specified, these are the default parameters for this chapter.

1. Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz × 8, RAM 31.4 GB

7.2 Experimental Metrics

To quantify the learning scenarios performances according to the objectives given in the section 1.6, several metrics are defined and some objectives are only evaluated qualitatively. The focus is made on prediction performance, *Endogenous Feedback*, *Agnosticity*, *Lifelong Learning*, *Online Learning*, *Self-Observation*, *Knowledge Generalization*, *Scalability*, *Any Data Amount*, *Explainability* and robotic applications.

Prediction Error

For a learning mechanism, the performance of the prediction is a central metric that needs to be evaluated. To evaluate the predictions of the learning mechanism, a normalized prediction error $\mathcal{O}_{Err}$ is calculated between the values given by the oracle and the learning system: $\mathcal{O}_{Err} = |\mathcal{O} - \mathcal{O}'|/(\mathcal{O}^{max} - \mathcal{O}^{min})$. $\mathcal{O}^{max}$ and $\mathcal{O}^{min}$ are the maximum and minimum experienced predictions in the *exogenous learning situations* during a learning phase. The prediction error metric is calculated and averaged during the exploitation phase over the *number of exploitation situations*. During the exploitation phase, *ELLSA* is asked to make predictions about random uniformly distributed *exploitation situations* in the search space.

Endogenous Feedback

To quantify the *Endogenous Feedback*, all types of *Endogenous Feedback* are considered. The focus is made on the number of generated *active learning situations* $\mathcal{L}_{\#}^{act}$, *endogenous exploitation situations* $\mathcal{E}_{\#}^{endo}$ and *endogenous learning situations* $\mathcal{L}_{\#}^{endo}$. This will permit to validate their presence, observe their impact and be able to say if they bring improvement.

- ▷ The *active learning situations* $\mathcal{L}_{\#}^{act}$ are present in the *Active Learning Strategy*. They are the actively requested *learning situations* $\mathcal{L}_{n,m}^{act} = [\mathcal{P}_n^{endo}, \mathcal{O}_m^{exo}]$ to resolve *learning inaccuracies*.
- ▷ The *endogenous exploitation situations* $\mathcal{E}_{\#}^{endo}$ are present in the *Self-Learning Strategy*. They are the self-requested *exploitation situations* $\mathcal{E}_n^{endo} = [\mathcal{P}_n^{endo}]$ to resolve *learning inaccuracies*.
- ▷ The *endogenous learning situations* $\mathcal{L}_{\#}^{endo}$ are also present in the *Self-Learning Strategy*. They are the self-generated *learning situations* $\mathcal{L}_{n,m}^{endo} = [\mathcal{P}_n^{endo}, \mathcal{O}_m^{endo}]$ during *Cooperative Neighborhood Learning*.
- ▷ The opposite of theses situations are the *passive learning situations* $\mathcal{L}_{\#}^{pass}$. *Passive learning situations* are exogenous *perceptions* associated with exogenous *prediction vectors* $\mathcal{L}_{n,m}^{pass} = [\mathcal{P}_n^{exo}, \mathcal{O}_m^{exo}]$. The interest of counting them is in the comparison of the randomly supplied *passive learning situations* $\mathcal{L}_{\#}^{pass}$ and the requested $\mathcal{L}_{\#}^{act}$ and $\mathcal{E}_{\#}^{endo}$ or generated $\mathcal{L}_{\#}^{endo}$ situations. *Exogenous learning situations* are also observed, they are exogenous or endogenous *perceptions* associated with exogenous *prediction vectors* $\mathcal{L}_{n,m}^{exo} = [\mathcal{P}_n^{exo}/\mathcal{P}_n^{endo}, \mathcal{O}_m^{exo}]$. Their measure enables to compare the quantities of exogenous and endogenous *learning situations*.
- ▷ The *active learning situations* and the *endogenous exploitation situations* associated with the resolutions of *Model Ambiguity NCS*, *Conflict NCS*, *Concurrency NCS*, *Incompetence NCS*

and *Range Ambiguity NCS* are also measured to observe they behavior in the different learning scenarios.

Agnosticity

This work did not focus on designing a metric for the measurement of *Agnosticity*. One possibility for this measure would be to use the learning mechanism in several fields of applications and count all those for which the learning performance is good. A good learning performance would need to be defined for each field of application.

In this thesis, *Agnosticity* is measured qualitatively across all the several conducted experiments. The experiments of this chapter do not focus on any specific application domain. They concentrate on several varieties of hidden models with different properties (linearity, non linearity, continuity and discontinuity). The experiment section 7.5.3 will focus on learning a *hidden function* that groups the different learning models in one.

Lifelong Learning

This work did not concentrate on designing a metric to measure the performances of the lifelong learning mechanisms. In this study, *Lifelong Learning* intervenes in the *local model* update mechanism but also during the exploitation phase. The updating mechanism enables *local models* to learn all *learning situations* according to the *model error margin*. This mechanisms will be assessed section 7.6.1. The generation of *endogenous exploitation situations* and *endogenous learning situations* enables *ELLSA* to continue learning even during the exploitation phase. *Lifelong Learning* in this sense will be evaluated in the section 7.6.2.

Online Learning

Online Learning is an intrinsic property of *Context Learning*. *Learning situations* can be sequentially provided and several learning and exploitation phases can follow one another. Moreover, all learning cycles involve a prediction proposition which is an exploitation. Learning and exploitation are done simultaneously. This property is partially assessed in the experiment section 7.9.

Self-Observation

Self-Observation is also an intrinsic property of *AMAS*. It enables to get feedback according to the agents that compose the system. In this case, it enables to assess the exploration of the *perceptions* space and the *learning inaccuracies* across different volumes.

- $\mathcal{V}_{Ctxt}{}'$ is the volume explored by all the *Context Agents*. It is the normalized sum of the volumes of each *Context Agent* $\mathcal{V}_{\mathcal{C}j}$ to which are subtracted the volumes of conflicts $\mathcal{V}_{Cflt}$ and concurrencies $\mathcal{V}_{Conc}$ between *Context Agents*.
  $\mathcal{V}_{Ctxt}{}' = (\sum_j \mathcal{V}_{\mathcal{C}j} - \mathcal{V}_{Cflt} - \mathcal{V}_{Conc})/\mathcal{V}_{total}$

- $\mathcal{V}_{Cflt}$ is the volume of conflicts. It is the sum of the volumes explored by two *Context Agents* with different models. $\mathcal{V}_{Cflt}{}'$ is its normalization.
  $\mathcal{V}_{Cflt} = (\sum_{(j,k),1 \leqslant j < k \leqslant n_{Ctxt}, f^j \neq f^k} \mathcal{V}_{\mathcal{C}^j \cap \mathcal{C}^k})$ and $\mathcal{V}_{Cflt}{}' = \mathcal{V}_{Cflt} / \mathcal{V}_{total}$
- $\mathcal{V}_{Conc}$ is the volume of concurrencies. It is the sum of the volumes explored by two *Context Agents* with similar models. $\mathcal{V}_{Conc}{}'$ is its normalization.
  $\mathcal{V}_{Conc} = (\sum_{(j,k),1 \leqslant j < k \leqslant \mathcal{N}_{ctxt}, f^j \approx f^k} \mathcal{V}_{\mathcal{C}^j \cap \mathcal{C}^k}) / \mathcal{V}_{total}$ and $\mathcal{V}_{Conc}{}' = \mathcal{V}_{Conc} / \mathcal{V}_{total}$
- $\mathcal{V}_{Inc}{}'$ is the normalized volume of the unexplored search space. $\mathcal{V}_{Inc}{}' = 1 - \mathcal{V}_{Ctxt}{}'$

$\mathcal{V}_{total}$ is the total volume of the considered search space. It is defined with the minimum p_i^{min} and maximum p_i^{max} perceptions experienced by the learning mechanism: $\mathcal{V}_{total} = \prod_i (p_i^{max} - p_i^{min})$. In this chapter, full exploration of the search is expected this is why all volumes are normalized by the total volume of the search space. To defined if an overlapping volume is a conflict or a concurrency, the measure of similarity between *Context Agents local models* is used. It is the presented section 6.2.3.3.

Knowledge Generalization

Two measures are necessary to evaluate the ability of the mechanism to generalize. They are the number of *Context Agents* n_{Ctxt} used to represent the space and the explored volume $\mathcal{V}_{Ctxt}{}'$. The *generalization score* is defined as $\mathcal{G}^{scr} = \mathcal{V}_{Ctxt}{}' / n_{Ctxt}$.

The numbers of *Complete Redundancy NCS* and *Partial Redundancy NCS* also represent generalization but locally. They contribute to decreasing the number of *Context Agents* and increasing their volume. The *Complete Redundancy NCS* and *Partial Redundancy NCS* respectively enable to merge *Context Agents* and to exchange a part of their volume. When these *NCS* occur, it means that *ELLSA* manages to locally enhance generalization. They provide feedback on all the situations that partially led to the final measure of the *generalization score*.

Scalability

To evaluate the scalability of the learning mechanism, time metrics are used to measure the execution time of the learning, the exploitation and the agents cycles. The measures are conducted section 7.8 by varying the number of *perceptions*, the *validity ranges precision* and *number of learning situations*. Decreasing the *validity ranges precision* and increasing the *number of learning situations* enable to create larger quantities of *Context Agents* and assess the behavior of *ELLSA* with larger amounts of *Context Agents*.

Any Data Amount

In this context, learning with *Any Data Amount* is strongly related to *Lifelong Learning* and *Online Learning*. The *Lifelong Learning* implemented mechanism enables to deal with large amounts of *learning situations* without storing them. And the *Online Learning* properties enable the learning mechanisms to provide predictions even with few *learning situations*. This objective is evaluated in the sections concerning *Lifelong Learning* (Sec. 7.6.1) and few *learning situations* 7.7.

Explainability

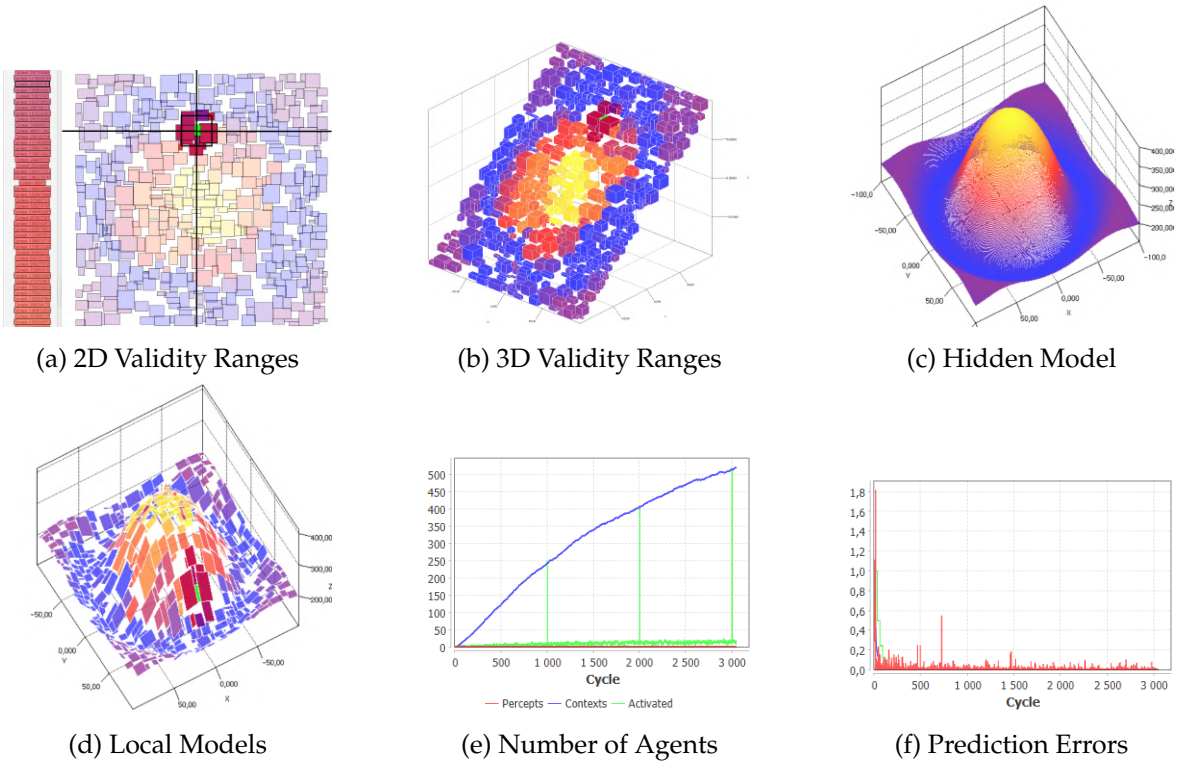
Explainability is a complicated objective to measure. This objective is addressed by the implementation of a user interface described section 7.3. This user interface provides many feedback and model visualization properties to understand the behavior of *ELLSA*. A qualitative reflection about this topic will be made after all the experiments.

Robotic Application

The robotic applications and experiments are described in the chapter 8. Two learning experiments are described. They involve the learning of the *Inverse Kinematic Models* of robotic arms with centralized and distributed control.

7.3 User Interface

In order to visualize numerous information concerning the agents and the learning process, I supervised the design of a user interface called *AMAKFX*. It was developed by Master's students during a project and an internship. It was implemented from *AMAK* [Perles et al., 2018], a framework developed in Java to facilitate the design and development of a *MAS*. *AMAKFX* uses *AMAK* with the software platform JavaFX [Clarke et al., 2009] for the graphical interface. The purpose of *AMAKFX* is to provide understandable feedback on *Context Learning* with explainable and graphical information. Figure 7.1 summaries some main elements of the user interface.



(a) 2D Validity Ranges (b) 3D Validity Ranges (c) Hidden Model

(d) Local Models (e) Number of Agents (f) Prediction Errors

Figure 7.1 – Screen-shots of the user interface *AMAKFX*.

The *validity ranges* can be observed in 2D (Fig. 7.1a) and 3D (Fig. 7.1b) by selecting the *perceptions* to be observed. The hidden model (Fig. 7.1c) can be compared to the learned *local models* (Fig. 7.1d). Darker *validity ranges* or *local models* represent the *Neighbor Context Agents* for the current *perceptions* and *neighborhood* area (cross and square figure 7.1a). The list of the *Context Agents* is also available on the left scrolling zone (Fig. 7.1a) and additional information about them can be displayed (confidence, last predictions, first learning situations...). Figure 7.1e provides the number of agents in the collective and how many are activated during a cycle. Figure 7.1f shows the real time prediction errors.

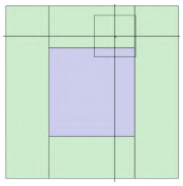
In the following, all experiments are presented with 2 *perceptions* to simplify visualization. For each experiment, its objectives are stated; the hidden model is presented; the protocol is described and examples of learned models are showed. Averaged experimental metrics results are provided and a discussion is conducted.

7.4 Learning Inaccuracies NCS

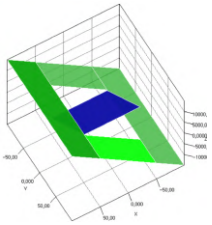
This section presents the experiments on a toy problem to evaluate and validate the *learning inaccuracies NCS* detection and resolution. An evaluation on different *validity ranges precisions*, *learning* and *exploitation criticalities* weights is given.

Objectives

The objective of this experiment is to validate the *learning inaccuracies NCS* on a simple problem where an ideal learning representation can be easily identify. Figure 7.2 shows an ideal learning representation for the *validity ranges* (Fig. 7.2a) and for the *local models* (Fig. 7.2b). It corresponds to 5 *Context Agents* representing 100% of the space and thus a maximal generalization score of $100/5 = 20\%$. The goal of this experiment is to show the resolution of each *learning inaccuracies* by incrementally adding the *learning inaccuracies NCS* detections.



(a) Capture of Ideal Learned *Validity Ranges*



(b) Capture of Ideal Learned *Local Models*

Figure 7.2 – 2 Perceptions Linear Toy Problem.

Protocol

The toy problem is a learning scenario with a *hidden function* $\mathcal{F}$ composed of two linear functions (Eq. 7.1) with two *perceptions* and one output.

$$\mathcal{F}(p_1, p_2) = \begin{cases} 150.p_2 + 20000 & \text{if } -h < p_1 < h \text{ and } -h < p_2 < h \\ 150.p_1 + 20000 & \text{else.} \end{cases} \tag{7.1}$$

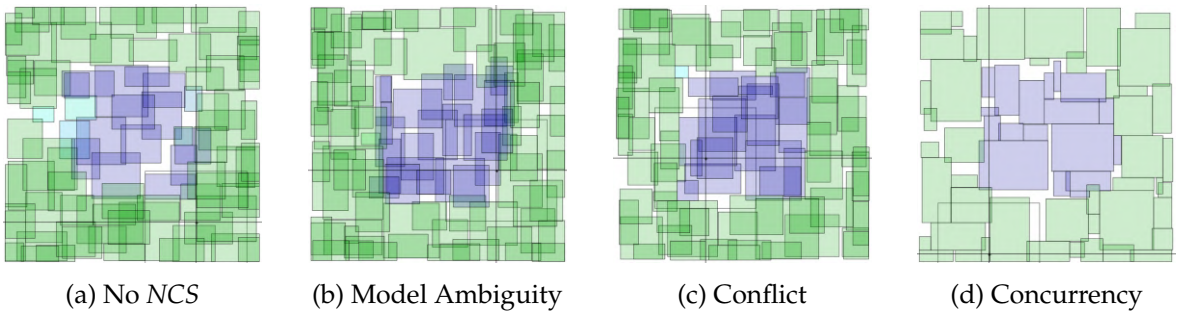
with $h = 50$ and $p_1, p_2 \in [-100, 100]$. For all the experiments of this chapter, p_1 and p_2 will always belong to this interval. Table 7.2 shows the experimental parameters for this scenario. All other user and designer parameters are the ones presented in the table 7.1. The *Active Learning Strategy* is used in this setting.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	500
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	250
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15

Table 7.2 – Table of experimental parameters for the validation of *learning inaccuracies NCS*.

Results

Figure 7.3 shows captures of the learning representation at the end of a learning episode. Each *learning inaccuracy NCS* is incrementally added. It can be seen that when all *learning inaccuracies NCS* are activated, the representation of the learning converges towards the ideal representation.



(a) No *NCS* (b) Model Ambiguity (c) Conflict (d) Concurrency

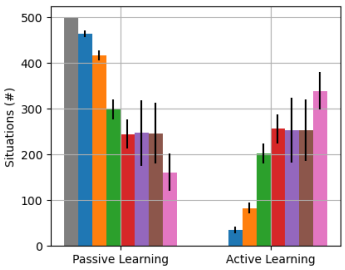
(e) Incompetence (f) Complete Redundancy (g) Partial Redundancy (h) Range Ambiguity

Figure 7.3 – Screenshots of *Context Agents* after 500 training cycles with different *NCS* resolution incrementally added. Each color is a different linear model.

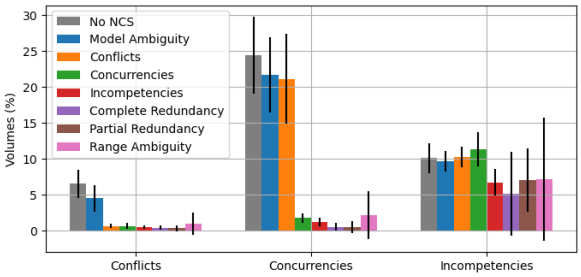
Incrementally adding the detection and resolution of each *learning inaccuracies NCS* enables to assess the impact of each one on the learning performances. Figure 7.4 shows the metrics presented section 7.2 for each case. All the values are averaged over the *number of exploitation situations*. The black thin bars represent the standard deviations. It is the case for all the metric results figures of the manuscript.

- ▷ **No *NCS***. When no *NCS* are solved, it is the reference case (gray label in the figure 7.4). Figure 7.3a shows a capture of the *validity ranges* obtained without any *NCS*. There are only *passive learning situations* (Fig. 7.4a). The volumes of conflicts and concurrencies are at their highest and the volume of incompetencies is of 10% (Fig. 7.4b). There are around 80 *Context Agents* (Fig. 7.4c), the *generalization score* is around 1% (Fig. 7.4d) and

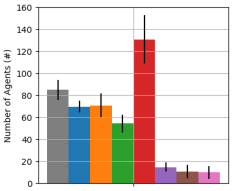
the prediction error is under 1% (Fig. 7.4e).



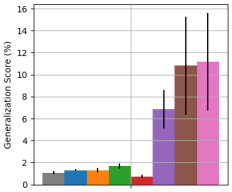
(a) Passive $\mathcal{L}_{\#}^{pass}$ and Active $\mathcal{L}_{\#}^{act}$ *learning situations* (#)



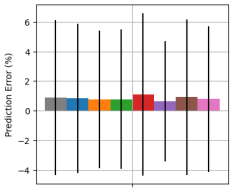
(b) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



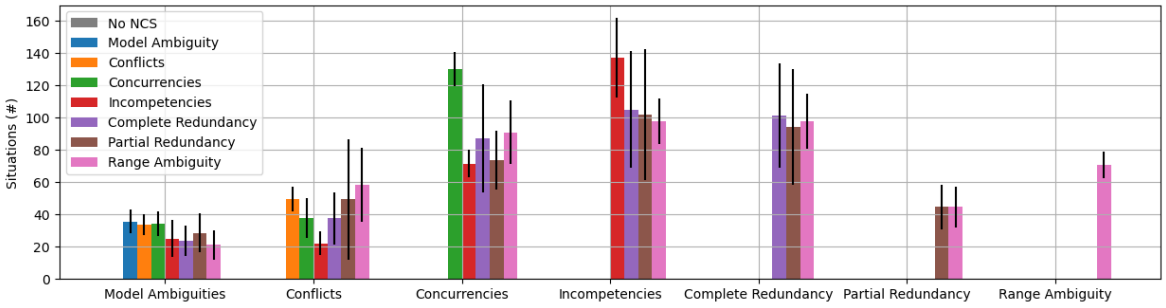
(c) Number of *Context Agents* n_{Ctxt} (#)



(d) Generalization score $\mathcal{G}^{scr}$ (%)



(e) Prediction error $\mathcal{O}_{Err}$ (%)



(f) Learning inaccuracies *NCS* counts.

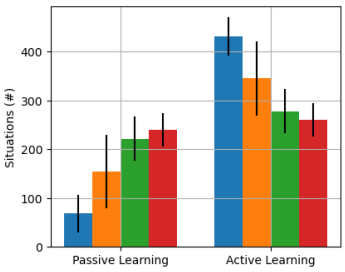
Figure 7.4 – Metrics results on the toy learning problem with the *Active Learning Strategy*. The *learning inaccuracies* NCS are incrementally added. The parameters are set to the values of the tables 7.2 and 7.1.

- ***Model Ambiguity NCS***. The *Model Ambiguity NCS* adds the generation of *active learning situations* to complete the *local models* (blue label in the figure 7.4). The difference is not visually noticeable in the figure 7.3b as it concerns the early stages of the *Context Agents*. Around 40 *active learning situations* are generated to solve this *NCS* (Fig. 7.4f). The volumes of *learning inaccuracies* slightly decrease (Fig. 7.4b). The lower number of *passive learning situations* reduces the numbers of *Context Agents* (Fig. 7.4c). The prediction error is slightly reduced (Fig. 7.4e).
- ***Conflict NCS***. The *Conflict NCS* adds the generation of *active learning situations* to suppress the conflicts between *Context Agents* (orange label in the figure 7.4). The *Conflict NCS* addition decreases the conflicting overlaps between the *Context Agents validity ranges* (Fig. 7.3c) and thus the volume of conflicts (Fig. 7.4b). Another impacted metric is the prediction error which also decreases (Fig. 7.4e).

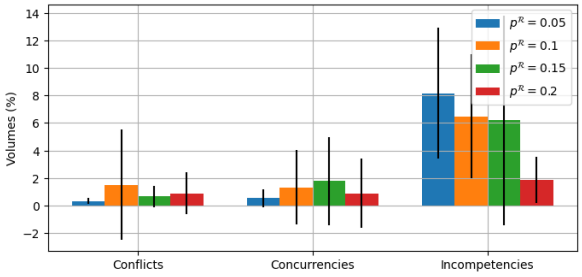
- ***Concurrency NCS***. The *Concurrency NCS* adds the generation of *active learning situations* to suppress the concurencies between *Context Agents* (green label in the figure 7.4). The *Concurrency NCS* addition clearly decreases the concurrent overlaps between the *Context Agents validity ranges* (Fig. 7.3d) and thus the volume of concurrencies (Fig. 7.4b). Resolving the concurrencies decreases the number of *Context Agents* (Fig. 7.4c) and increases the incompetent volumes (Fig. 7.4b). It makes sense because the resolution of overlaps can generate volume loss.
- ***Incompetence NCS***. The *Incompetence NCS* adds the generation of *active learning situations* to seek incompetent areas (red label in the figure 7.4). It enables to reduce the incompetent volumes (Fig. 7.4b) and prevents the conflicts and concurencies as less are counted (Fig. 7.4f). However, more *Context Agents* are created (Fig. 7.4c) and the prediction error is worse (Fig. 7.4e).
- ***Complete Redundancy NCS***. The *Complete Redundancy NCS* adds the possibility for *Context Agents* to merge (purple label in the figure 7.4). This significantly decreases the number of *Context Agents* (Fig. 7.4c) and increases the *generalization score* (Fig. 7.4d). It prevents some incompetencies but generates slightly more conflicts and concurrencies to solve (Fig. 7.4f).
- ***Partial Redundancy NCS***. The *Partial Redundancy NCS* adds the possibility for *Context Agents* to cede a part of their *validity ranges* (brown label in the figure 7.4). It allows to unlock situations where *Context Agents* cannot merge like in figure 7.3f. This tends to generate more incompetencies (Fig. 7.4b) but the *generalization score* is better (Fig. 7.4d).
- ***Range Ambiguity NCS***. The *Range Ambiguity NCS* adds the generation of *active learning situations* to enhance discontinuities between *Context Agents* (pink label in the figure 7.4). This *NCS* enables to reach the best *generalization score* (Fig. 7.4d) but not the best prediction error (Fig. 7.4e).

Validity Ranges Precision Sensibility

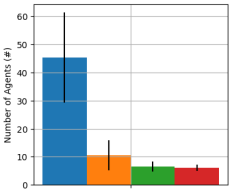
Figure 7.5 shows the sensibility of the learning metrics in relation to the *validity ranges precision*. For high *validity ranges precisions*, there is less need for *active learning situations* (Fig. 7.5a). Indeed, figure 7.5f shows that for low *validity ranges precisions* there are more *Model Ambiguity NCS* and *Incompetence NCS*. Smaller *validity ranges precisions* implies smaller *neighborhoods* and *Context Agents* are less likely to share their *local models* in smaller areas. Thus, more *Context Agents* need to complete their models with *Model Ambiguity NCS*. The high count of *Incompetence NCS* is explained by the higher volume of incompetencies for low *validity ranges precisions* (Fig. 7.5b). Smaller *Context Agents* need a bigger collective to represent the space which is why there are more *Context Agents* for low *validity ranges precisions* (Fig. 7.5c). For high *validity ranges precisions* the *generalization score* almost reaches the ideal of 20% (Fig. 7.5d) but at the price of a worse prediction error (Fig. 7.5e).



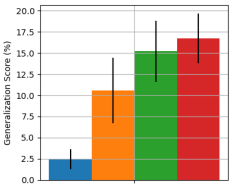
(a) Passive $\mathcal{L}_{\#}^{pass}$ and Active $\mathcal{L}_{\#}^{act}$ *learning situations* (#)



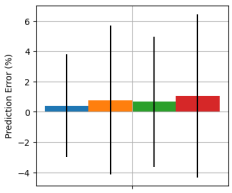
(b) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



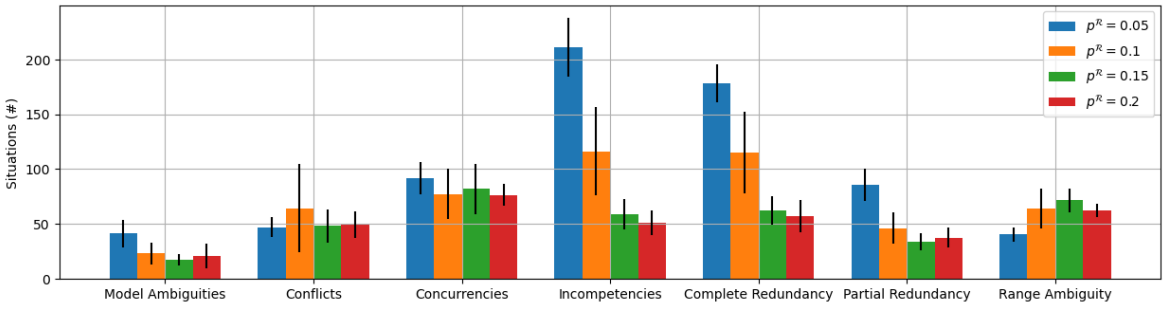
(c) Number of *Context Agents* n_{Ctxt} (#)



(d) Generalization score $\mathcal{G}^{scr}$ (%)



(e) Prediction error $\mathcal{O}_{Err}$ (%)



(f) Learning inaccuracies *NCS* counts.

Figure 7.5 – Metrics results on the toy learning problem with the *Active Learning Strategy* with different *validity ranges precisions* $p^{\mathcal{R}} = \{0.05; 0.1; 0.15; 0.2\}$. The other parameters are set to the values of the tables 7.2 and 7.1.

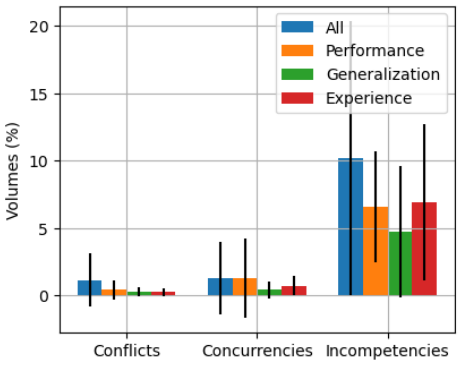
Learning and Exploitation Criticalities Weights Sensibility

Figure 7.6 presents an evaluation on different concerns for the *learning criticality* and the *exploitation criticality* (all, performance, generalization and experience). The different interests result in the following weight settings.

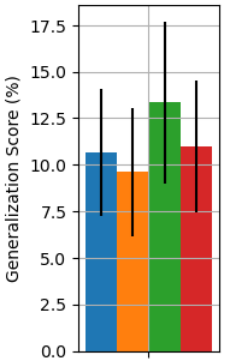
- ▷ **All**: $w_{f_n}^{lrn} = 1, w_{c_{0,1}}^{lrn} = 1, w_{\mathcal{R}_n}^{lrn} = 1, w_{\mathcal{P}_n}^{expl} = 1, w_{c_{0,1}}^{expl} = 1, w_{\mathcal{R}_n}^{expl} = 1$
- ▷ **Performance**: $w_{f_n}^{lrn} = 1, w_{c_{0,1}}^{lrn} = 0.5, w_{\mathcal{R}_n}^{lrn} = 0.5, w_{\mathcal{P}_n}^{expl} = 1, w_{c_{0,1}}^{expl} = 0.5, w_{\mathcal{R}_n}^{expl} = 0.5$
- ▷ **Generalization**: $w_{f_n}^{lrn} = 0.5, w_{c_{0,1}}^{lrn} = 0.5, w_{\mathcal{R}_n}^{lrn} = 1, w_{\mathcal{P}_n}^{expl} = 0.5, w_{c_{0,1}}^{expl} = 0.5, w_{\mathcal{R}_n}^{expl} = 1$
- ▷ **Experience**: $w_{f_n}^{lrn} = 0.5, w_{c_{0,1}}^{lrn} = 1, w_{\mathcal{R}_n}^{lrn} = 0.5, w_{\mathcal{P}_n}^{expl} = 0.5, w_{c_{0,1}}^{expl} = 1, w_{\mathcal{R}_n}^{expl} = 0.5$

These weights intervene for the selection of the *Best Context Agent* during learning and exploitation. Figure 7.6c shows that giving more weight to performance (*local model* affinity

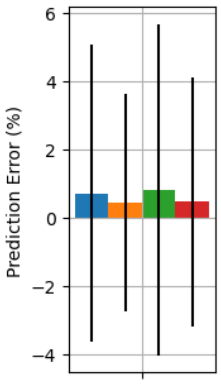
during learning and *validity ranges* proximity during exploitation) provides better predictions errors than giving weight to all focuses. Figures 7.6a and 7.6b show that giving more weight to generalization enables to decrease the volume of *learning inaccuracies* and increase the *generalization score*. It can be seen that giving weight to experience provides better generalization results that the focus on performance and better prediction results than the focus on generalization. A third metric would be needed to measure the global experience of all *Context Agents* in order to better evaluate the gain that it represents.



(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(b) Generalization score $\mathcal{G}^{scr}$ (%)



(c) Prediction error $\mathcal{O}_{Err}$ (%)

Figure 7.6 – Metrics results on the toy learning problem with the *Active Learning Strategy* with different learning and exploitation concerns for the *learning criticality* and the *exploitation criticality*: all, performance, generalization and experience. The other parameters are set to the values of the tables 7.2 and 7.1.

Synthesis

This experiment enabled to validate the resolutions of the *learning inaccuracies* on a simple learning problem by incrementally adding each resolutions. The *validity ranges precision* enables to select the precision degree at the cost of a worse generalization. The high deviation values for the prediction error are explained by the discontinuous nature of the problem. The evaluation of *learning criticality* and *exploitation criticality* weights validates that different learning and exploitation concerns can be set by giving importance to performance, generalization, experience or all of them.

All the *learning inaccuracies NCS* are independent from the number of *perceptions*. However, additional experiments are necessary to validate the mechanisms with more *perceptions*. Section 7.8 will assess this learning problem on higher dimensions.

7.5 Active Learning VS Self-Learning

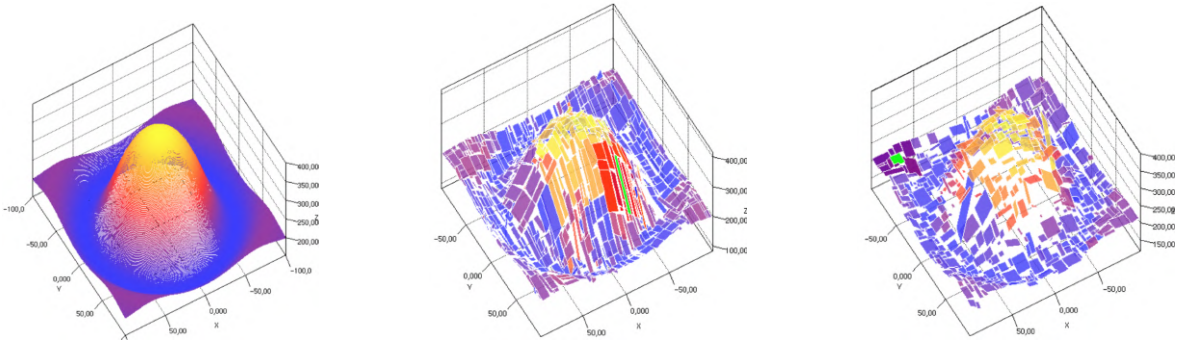
This section assesses the performances of the *Active Learning Strategy* and the *Self-Learning Strategy* on non linear learning problems without and with discontinuities. A multi model problem is also presented.

7.5.1 Non Linear Continuous Problem

This section presents the experiments on a non linear continuous problem. An evaluation on different *model error margins* and *discontinuity detection probabilities* is given.

Objectives

The objective of this experiment is to validate the *Active Learning Strategy* and the *Self-Learning Strategy* on a non linear continuous learning problem. Non linearity better represents real world problem. This experiment also intends to show the strengths and weaknesses of the *Active* and *Self-Learning Strategies* compared to a *Naive Learning Strategy*. Figure 7.7 shows the hidden model and the learned *local models* by the *Active Learning Strategy* and the *Self-Learning Strategy*.



(a) Hidden Model

(b) *Active Learning Strategy* learned models

(c) *Self-Learning Strategy* learned models

Figure 7.7 – Captures of the 2 Perceptions Continuous Non Linear Problem

Protocol

The *hidden function* is a continuous non linear function with two *perceptions* and one output (Eq. 7.2 and Fig. 7.7a). It was inspired from the intensity diffraction patterns in physics.

$$\mathcal{F}(p_1, p_2) = \alpha_1 . \prod_i \exp^{-\frac{(p_i-\mu)^2}{2\sigma^2}} . \cos(\alpha_2 . (p_1^2 + p_2^2)) + \alpha_3 \tag{7.2}$$

with μ, σ, α_1, α_2 and α_3 arbitrary parameters. Table 7.3 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1. The *Naive Learning*

Strategy only uses *passive exogenous learning situations*. It does not use any *learning inaccuracies NCS* and neither the mechanisms that involve *Neighbor Context Agents*.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	2000
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	250
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	0.04

Table 7.3 – Table of experimental parameters for the experiments on continuous non linearity.

Results

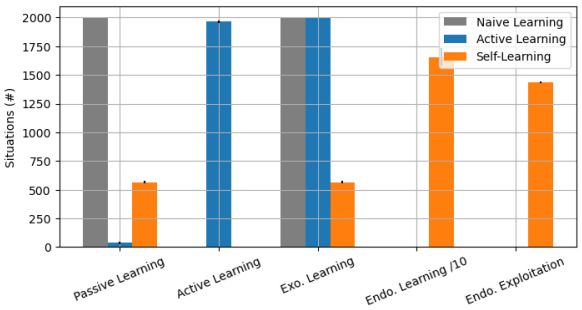
Figure 7.8 shows the learning metrics for the non linear continuous learning problem for the *Naive Learning Strategy* (gray label), the *Active Learning Strategy* (blue label) and the *Self-Learning Strategy* (orange label).

Figure 7.8a presents the used situations for this learning problem. The *Naive Learning Strategy* only uses *passive exogenous learning situations* as expected. The *Active Learning Strategy* uses *learning situations* that are mostly active and all of them are exogenous (i.e. their are provided by an oracle). The *Self-Learning Strategy* uses *passive learning situations* which are the only *exogenous learning situations* provided by an oracle. The other situations are *endogenous exploitation situations* and *endogenous learning situations*.

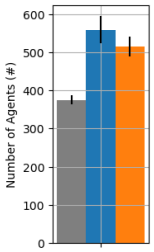
Compared to the *Naive Learning Strategy*, the *Active Learning Strategy* and the *Self-Learning Strategy* lead to a worse *generalization score* (Fig. 7.8c) because more *Context Agents* are created (Fig. 7.8b) and the volumes of incompetencies are more important (Fig. 7.8e). However the important volume of conflicts with the *Naive Learning Strategy* are greatly reduced with the *Active* and *Self-Learning Strategy*. The prediction errors means are slightly augmented but the deviations are reduced (Fig. 7.8d).

In the *Active Learning Strategy*, there are more *Context Agents* (Fig. 7.8b) and the volume of incompetencies is also higher (Fig. 7.8e). This explains the better *generalization score* for the *Self-Learning Strategy* (Fig. 7.8c). In the *Active Learning Strategy*, most of the *Context Agents* are created with *Incompetence NCS* (Fig. 7.8f). Even is there are more *Context Agents* with the *Active Learning Strategy*, the space is more poorly explored because it is locally explored. This prevents a global exploration like in the *Self-Learning Strategy*. The consequence is that the prediction error for both strategies are close but the deviation for the *Active Learning Strategy* is higher (Fig. 7.8d).

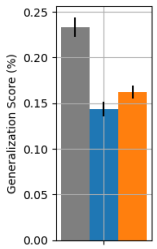
It can be seen that there are more *Conflict NCS* with the *Self-Learning Strategy* as the *validity ranges* are rather initialized in a passive way as oppose to the active strategy which does not need a minimal amount of *Neighbor Context Agents* to fill a void. There is a huge difference in the amount of *Model Ambiguity NCS* between the two strategies. This can be explained by the lower *Neighbor Context Agents* as the space is explored randomly in the *Self-Learning Strategy*. With fewer *Neighbor Context Agents*, the sharing of good *local models* is



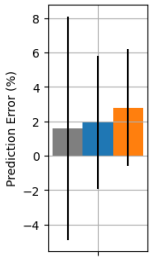
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



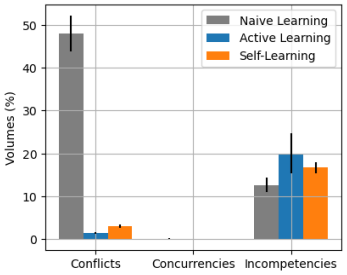
(b) Number of *Context Agents* n_{Ctxt} (#)



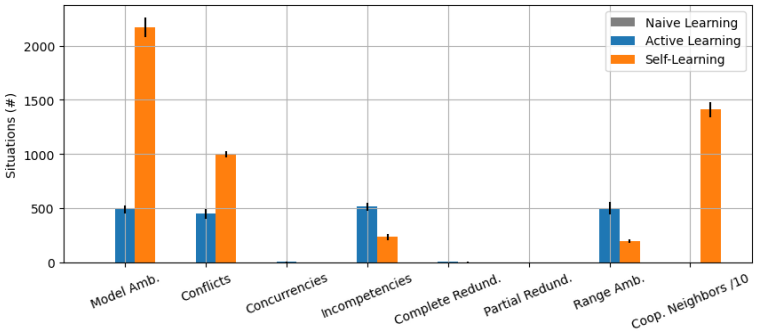
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



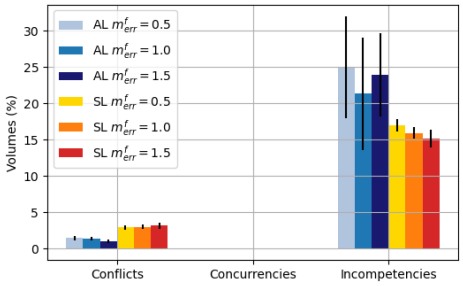
(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.8 – Metrics results on the non linear continuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.3 and 7.1.

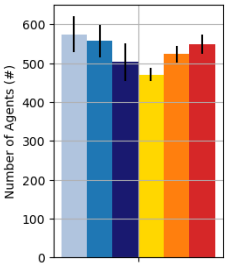
less likely to happen and more *Model Ambiguity NCS* are generated. An issue is the too high number of *Range Ambiguity NCS* as this problem is known continuous. *Range Ambiguity NCS* should not be present in this learning scenario.

Model Error Margin Sensibility

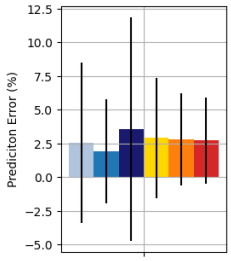
Figure 7.9 presents an evaluation of some metrics on different *model error margins*. For the *Active Learning Strategy*, lower *model error margins* imply more *Context Agents*. The prediction error seems related to the volumes of incompetencies. A lower *model error margin* than 1.0 combined with the current parameter results in difficulties to map the space as there are more incompetencies. A higher *model error margin* than 1.0 implies less constraining prediction needs. Thus less *Context Agents* are needed but the prediction error is worse. For the *Self-Learning Strategy*, the *model error margin* has a low impact on the prediction error. It is compensated by the *Cooperative Neighborhood Learning* and the greater number of *Context Agents* when the *model error margin* is higher.



(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(b) Number of *Context Agents* n_{Ctxt} (#)

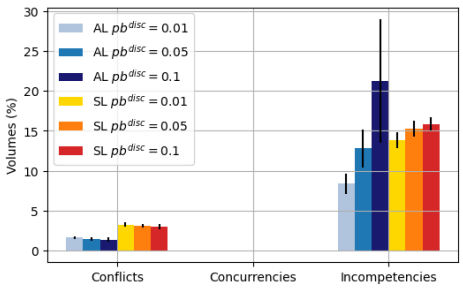


(c) Prediction error $\mathcal{O}_{Err}$ (%)

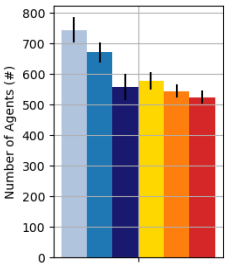
Figure 7.9 – Impacted metrics results on the non linear continuous learning problem with the *Active Learning Strategy* (AL) and the *Self-Learning Strategy* (SL) with different *model error margins* $m^f_{err} = \{0.5; 1.0; 1.5\}$. The other parameters are set to the values of the tables 7.3 and 7.1.

Discontinuity Detection Probability Sensibility

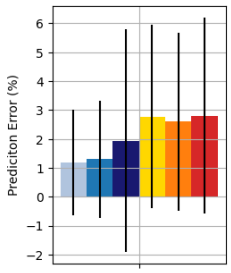
Figure 7.10 presents an evaluation of some metrics on different *discontinuity detection probabilities*. The impact is mainly the same for both strategies. When *Range Ambiguity NCS* are less sought (i.e. the *discontinuity detection probability* is lower), the situations that where allocated for the resolution of *Range Ambiguity NCS* are used to further explore the space. The results are lower volumes of incompetencies, more *Context Agents* and a lower prediction error (average and deviation). For the *Self-Learning Strategy* though, the prediction error is rather independent of the *discontinuity detection probability*.



(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(b) Number of *Context Agents* n_{Ctxt} (#)



(c) Prediction error $\mathcal{O}_{Err}$ (%)

Figure 7.10 – Impacted metrics results on the non linear continuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy* with different *discontinuity detection probabilities* $pb^{disc} = \{0.01; 0.05; 0.1\}$. The other parameters are set to the values of the tables 7.3 and 7.1.

Synthesis

This experiment showed that the *Active Learning Strategy* and the *Self-Learning Strategy* can learn a non linear continuous model. The *Naive Learning Strategy* resulted in better generalization. This raises a lack in the generalization score because the *Naive Learning Strategy* generated a lot of conflicts compared to the *Active Learning Strategy* and *Self-Learning Strategy*. An improvement of the *generalization score* would be to take into account the volume of conflicts (and concurrencies). The prediction error was improved in term of deviation, the *Active* and *Self-Learning Strategy* provided more stable predictions.

The *Self-Learning Strategy* enabled to reach a close prediction error to the *Active Learning Strategy* with lesser *exogenous learning situations*. The *Active Learning Strategy* is more efficient than the *Self-Learning Strategy* when it has sufficient *learning situations* to explore the space. Otherwise the *Self-Learning Strategy* is better indicated when less *exogenous learning situations* are available as it reaches close prediction errors to the other strategies with 4 times less *exogenous learning situations*. There is still room for improvement concerning the *Range Ambiguity NCS* because it slows down the learning process on a continuous problem where no discontinuities should be detected.

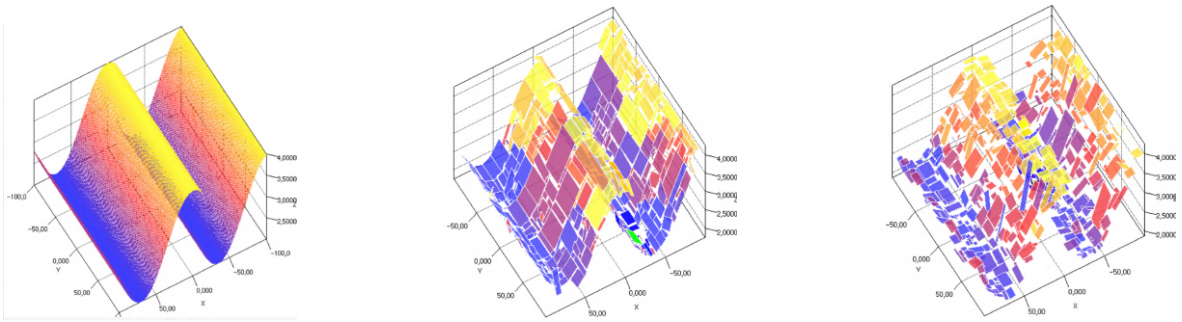
With the *Active Learning Strategy*, only lowering the *model error margin* is not enough to reach better prediction errors as it creates more incompetent zones and a need for more *learning situations* to fill them. Concerning the *Self-Learning Strategy*, the prediction error is little impacted by the *model error margin* and the *discontinuity detection probability* because the *endogenous learning situations* enable to compensate the lack of other *learning situations* or precision constrains.

7.5.2 Non Linear Discontinuous Problem

This section presents the experiments on a non linear discontinuous problem. An evaluation on different *model similarity thresholds* is given.

Objectives

The objective of this experiment is to validate the *Active Learning Strategy* and the *Self-Learning Strategy* on a non linear discontinuous learning problem compared to the *Naive Learning Strategy*. The discontinuity of the hidden model is present to test the ability of *ELLSA* to learn with the addition of this difficulty. Unlike the previous experiment, this *hidden function* allows generalization on certain zones to also test the generalization abilities of *ELLSA*. Figure 7.11 shows the hidden model and the learned *local models* by the *Active Learning Strategy* and the *Self-Learning Strategy*.



(a) Hidden Model

(b) *Active Learning Strategy* learned models

(c) *Self-Learning Strategy* learned models

Figure 7.11 – Captures of the 2 Perceptions Discontinuous Non Linear Problem

Protocol

The *hidden function* is a discontinuous non linear function with two *perceptions* and one output (Eq. 7.3 and Fig. 7.11a).

$$\mathcal{F}(p_1, p_2) = \begin{cases} \cos(\alpha_1 p_1) + \alpha_2 & \text{if } h < p_1 \\ \sin(\alpha_1 p_1) + \alpha_2 & \text{else.} \end{cases} \tag{7.3}$$

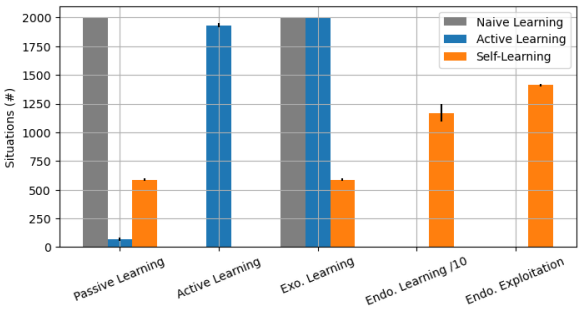
with h, α_1 and α_2 arbitrary parameters. h is the perception p_1 where the discontinuity is present. Table 7.4 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1. The *Naive Learning Strategy* is the one presented in the previous experiment (Section 7.5.1).

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	2000
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	250
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$	0.04
	model error margin	m^{f}_{err}	> 0	$\mathbb{R}$	0.05

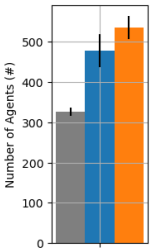
Table 7.4 – Table of experimental parameters for the experiments on discontinuous non linearity.

Results

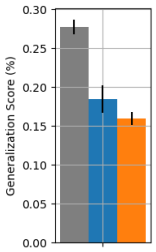
Figure 7.12 shows the learning metrics for the non linear discontinuous learning problem for the *Naive Learning Strategy* (gray label), the *Active Learning Strategy* (blue label) and the *Self-Learning Strategy* (orange label).



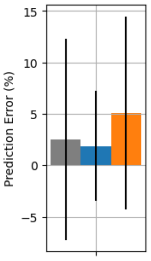
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



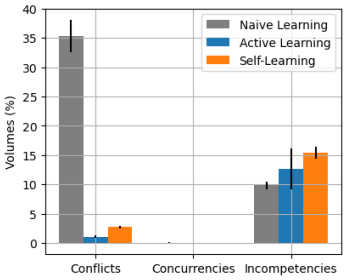
(b) Number of *Context Agents* n_{Ctxt} (#)



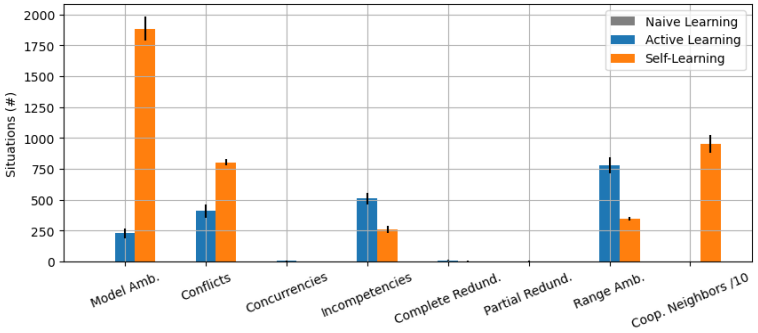
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

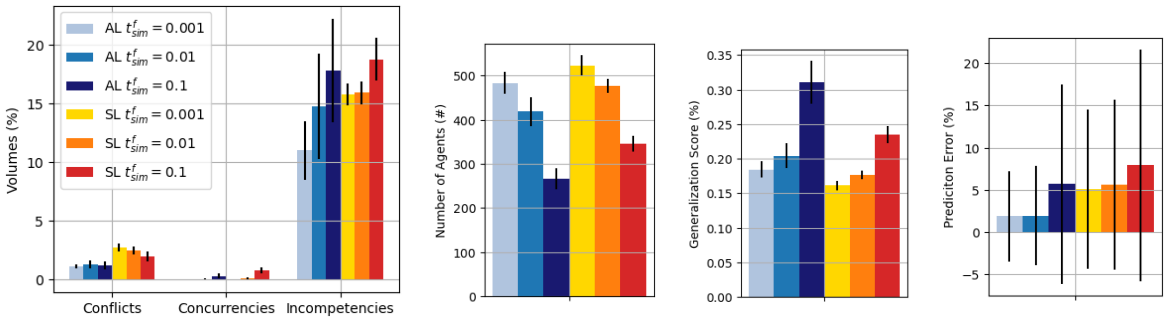
Figure 7.12 – Metrics results on the non linear discontinuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.4 and 7.1.

The distribution of passive, active, exogenous and endogenous *learning situations* is sim-

ilar to the previous problem (Fig. 7.12a).

Concerning the *generalization score*, the same behavior as before is experimented (Fig. 7.12c). It is way better for the *Naive Learning Strategy* because conflicts are not accounted in the generalization metric. The resolution of all theses conflicts generates more incompetencies than the *Naive Learning Strategy* (Fig. 7.12e). The *Active Learning Strategy* provides better prediction performances than the *Naive Learning Strategy* but the prediction performances of *Self-Learning Strategy* are worse (Fig. 7.12d).

If we now compare the *Active Learning Strategy* and the *Self-Learning Strategy*, the *generalization score* is better for the *Active Learning Strategy* (Fig. 7.12c) as it creates less *Context Agents* (Fig. 7.12b) and less incompetent volumes (Fig. 7.12e). Due to the discontinuity, there are more *Range Ambiguity NCS* than the previous experiment and the other *learning inaccuracies NCS* behave like in the previous experiment (Fig. 7.12f). However, there aren't any *Complete Redundancy NCS* or *Partial Redundancy NCS* as expected. The prediction error is lower for the *Active Learning Strategy* (Fig. 7.12d). It can be again related to the discontinuity and to the lower volume of incompetencies for the *Active Learning Strategy* (Fig. 7.12e).

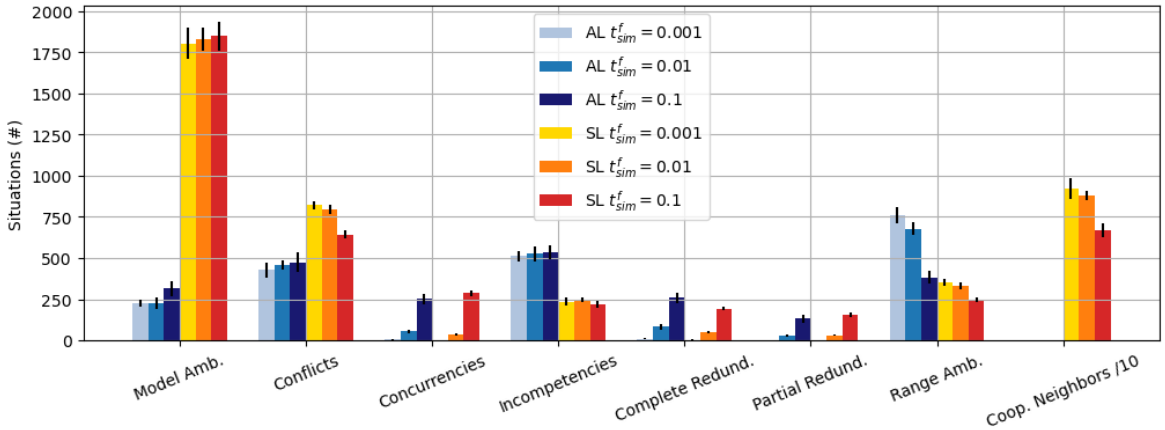


(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)

(b) Number of *Context Agents* n_{Ctxt} (#)

(c) Generalization score $\mathcal{G}^{scr}$ (%)

(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.13 – Metrics results on the toy learning problem with the *Active Learning Strategy* with different *model similarity thresholds* $t^f_{sim} = \{0.001; 0.01; 0.1\}$. The other parameters are set to the values of the tables 7.4 and 7.1.

Model Similarity Threshold Sensibility

In order to study the generalization behavior for this experiment, figure 7.13 presents an evaluation of some metrics on different *model similarity thresholds*. Augmenting the *model similarity threshold* generates more *Complete Redundancy NCS* and *Partial Redundancy NCS* as expected because the similarity constraint is weaker (Fig. 7.13e). Thus, there are less *Context Agents* (Fig. 7.13b), the *generalization score* is higher (Fig. 7.13c) but the prediction error is higher (Fig. 7.13d) and the volume of incompetencies increases too (Fig. 7.13a). It can be noticed that the number of *Concurrency NCS* increases with the *model similarity threshold* because some *Conflict NCS* become *Concurrency NCS* by reducing the similarity constraint. Moreover, there are less *Range Ambiguity NCS* because less *local models* are considered different.

Synthesis

This experiment shows that the *Active Learning Strategy* and the *Self-Learning Strategy* can learn a non linear discontinuous model. It turns out that the *Active Learning Strategy* reaches better prediction performances than the *Naive Learning Strategy* when the *Self-Learning Strategy* prediction performances were worse than both other strategies. Indeed, with the used parameter and the presence of a discontinuity, the prediction error for the *Self-Learning Strategy* is higher than the one with the *Naive Learning Strategy* but with fewer *exogenous learning situations*.

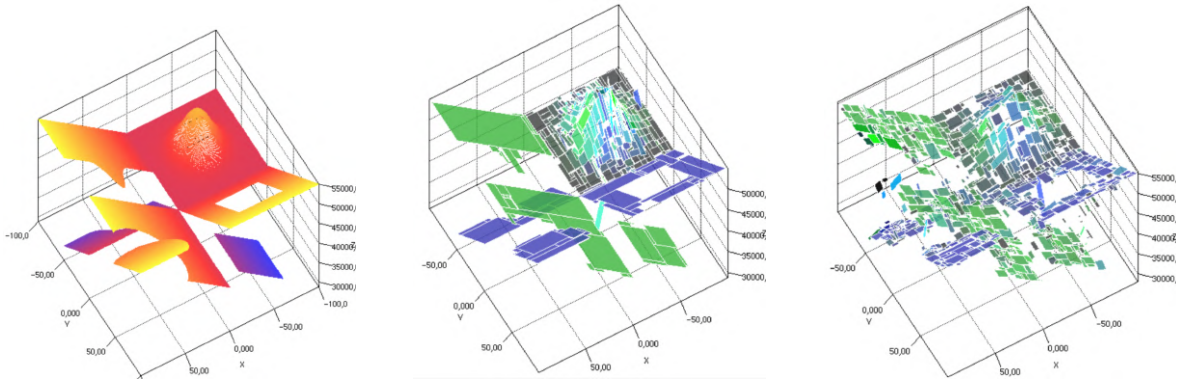
The *Context Agents* generalization mechanisms are highly dependent of the *model similarity threshold*. This parameter also impacts the *Conflict NCS*, the *Concurrency NCS*, the *Range Ambiguity NCS* and the exploration of incompetent areas.

7.5.3 Multi-Model

This section presents the experiments on a multi-model problem. This experiment introduces non linear frontiers between hidden models.

Objectives

The objective of this experiment is to validate the *Active Learning Strategy* and the *Self-Learning Strategy* on a multi-model learning problem compared to the *Naive Learning Strategy*. This experiment gathers all the previous learning problems (linear and non linear) and difficulties (continuity, discontinuity and generalization). This experiment adds non linear frontiers. Figure 7.14 shows the hidden model and the learned *local models* by the *Active Learning Strategy* and the *Self-Learning Strategy*.



(a) Hidden Model

(b) *Active Learning Strategy* Learned Models

(c) *Self-Learning Strategy* Learned Models

Figure 7.14 – Captures of the 2 Perceptions Multi-Model Problem

Protocol

The *hidden function* is composed of four zones with different difficulties, two *perceptions* and one output (Eq. 7.4 and Fig. 7.14).

$$\mathcal{F}(p_1, p_2) = \begin{cases} \mathcal{F}_1(p_1, p_2) & \text{if } 0 \leqslant p_1 \text{ and } 0 \leqslant p_2 \\ \mathcal{F}_2(p_1, p_2) & \text{if } 0 > p_1 \text{ and } 0 < p_2 \\ \mathcal{F}_3(p_1, p_2) & \text{if } 0 > p_1 \text{ and } 0 \leqslant p_2 \\ \mathcal{F}_4(p_1, p_2) & \text{else.} \end{cases} \tag{7.4}$$

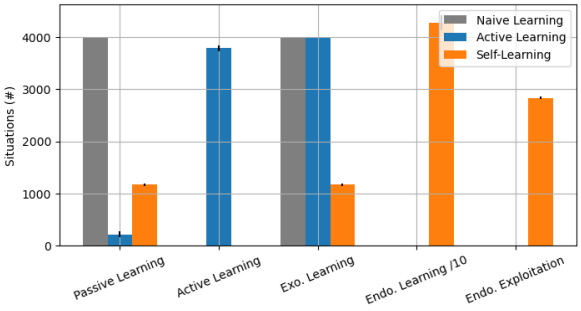
- $\mathcal{F}_1$ is composed of two linear models f_1 and f_2 with a non linear discontinuity represented by the equation of a circle.
- $\mathcal{F}_2$ is a non linear model represented by a Gaussian function centered in the considered zone.
- $\mathcal{F}_3$ is the same problem than the section 7.4 with the linear models f_1 and f_2.

▷ $\mathcal{F}_4$ is composed of two linear models f_1 and f_2 with a non linear discontinuity represented by a Gaussian function.

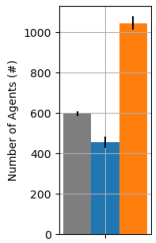
The linear models f_1 and f_2 are the one used in the section 7.4: $f_1(p_1, p_2) = 150p_2 + 20000$; $f_2(p_1, p_2) = 150p_1 + 20000$. Table 7.5 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1. The same *Naive Learning Strategy* is again compared to the *Active* and *Self-Learning Strategies*.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	4000
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	500
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	0.03
	model error margin	m^{f}_{err}	> 0	$\mathbb{R}$	1

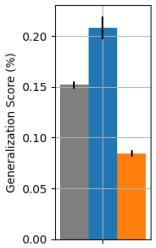
Table 7.5 – Table of experimental parameters for the mutli-model experiments.



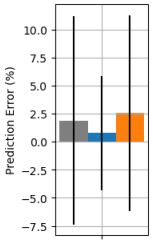
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



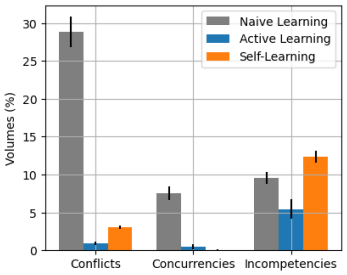
(b) Number of *Context Agents* n_{Ctxt} (#)



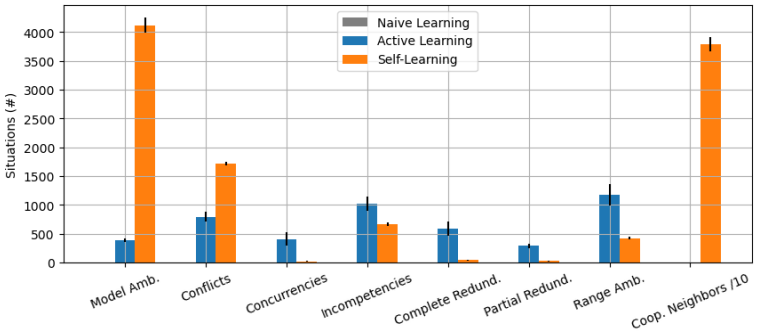
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.15 – Metrics results on the non linear discontinuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.4 and 7.1.

Results

Figure 7.15 shows the learning metrics for the multi-model learning problem for the *Naive Learning Strategy* (gray label), the *Active Learning Strategy* (blue label) and the *Self-Learning Strategy* (orange label). The distribution of passive, active, exogenous and endogenous *learning situations* is similar to the previous problems (Fig. 7.15a).

Compared to the *Naive Learning Strategy*, the best generalization is reached for the *Active Learning Strategy* and the worse for the *Self-Learning Strategy* (Fig. 7.15c). As neither conflicts nor concurrencies impact the *generalization score*, it is better for the *Naive Learning Strategy* than for the *Self-Learning Strategy*. Both the *Active Learning Strategy* and the *Self-Learning Strategy* significantly decrease the volumes of conflicts and concurrencies compared to the *Naive Learning Strategy* (Fig. 7.15e). Figure 7.15d shows that the *Active Learning Strategy* reaches better prediction errors than the *Naive Learning Strategy* and the *Self-Learning Strategy* reaches similar ones.

If we now compared the *Active Learning Strategy* and the *Self-Learning Strategy* only, the *generalization score* of *Active Learning Strategy* is more than twice the *generalization score* of the *Self-Learning Strategy* (Fig. 7.15c). With the *Active Learning Strategy*, there are much less *Context Agents* (Fig. 7.15b) and much less incompetent volumes (Fig. 7.15e). This is explained by the better exploration of the *Active Learning Strategy* and its higher number of *Complete Redundancy NCS* and *Partial Redundancy NCS* (Fig. 7.15f). As the *Self-Learning Strategy* enhances the *local models* by smoothing their frontiers with their neighbors, it takes longer for them to converge toward the hidden models. While the *Active Learning Strategy* combined with the *Model Ambiguity NCS* provides to *local models* the exact hidden model if all the active *Model Ambiguity NCS learning situations* belong to same linear model. This can be seen by the presence of concurrence volumes (Fig. 7.15e) and *Concurrency NCS* (Fig. 7.15f) for the *Active Learning Strategy*. For the *Self-Learning Strategy*, there are not concurrence volumes nor *Concurrency NCS*, there are mostly conflict volumes and *Conflict NCS*.

Again, the prediction error is lower for the *Active Learning Strategy* (Fig. 7.15d). It is due to the better exploration (less incompetent volumes are observable on the figure 7.15e) and generalization (Fig. 7.15c) than the *Self-Learning Strategy*. However, the *Self-Learning Strategy* manages to reach a comparable prediction error with four times less *exogenous learning situations*.

Synthesis

This experiment showed that the *Active Learning Strategy* and the *Self-Learning Strategy* can learn a multi-model problem composed of continuity, discontinuity and non linearity. The *Active Learning Strategy* performed better than the *Naive Learning Strategy* and the *Self-Learning Strategy* performed similarly.

The learning was performed on all zones with a unique set of parameters which shows that *ELLSA* can adapt to different learning difficulties without requiring special parameters tuning for each different model.

7.6 Lifelong Learning

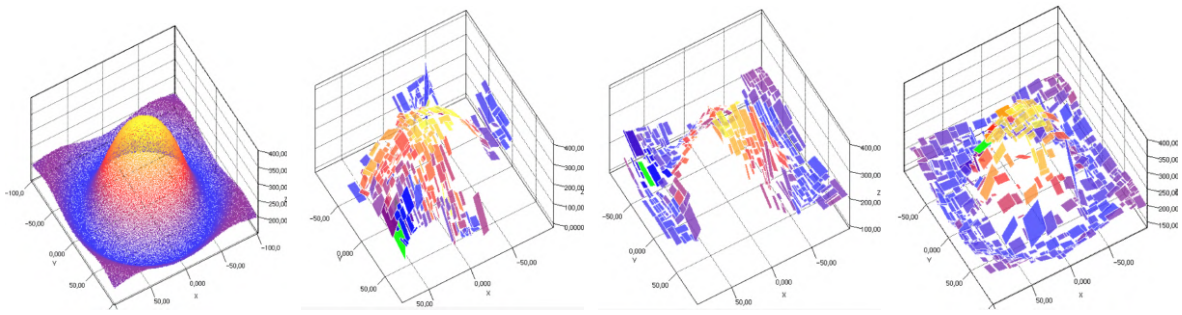
This section presents experiments on the *Lifelong Learning* implemented mechanism. One with noise in the *learning situations* and one with lifelong exploitation of the *Context Agents*. *Lifelong Context Learning* was designed to keep providing *learning situations* to the *local models* without storing them. This mechanism has two goals. The first one is to deal with noise in the *learning situations*. The second one is to enable the *local models* to never stop learning by being able to use new *learning situations* (exogenous or endogenous) during lifelong experiences.

7.6.1 Noisy Problem

In this section an experimentation on the *Lifelong Context Learning* mechanism is presented. We have previously seen that to enable *Lifelong Learning*, artificial situations are generated in order to update the *local models* with a desired weight. This mechanism is intended for being robust to noisy data.

Objectives

The objective of this experiment is to assess the *Active Learning Strategy* and the *Self-Learning Strategy* on a noisy model learning problem. The *Active Cooperative Learning Strategy* is introduced to evaluate the impact of *Cooperative Neighborhood Learning* with the *Active Learning Strategy*. Another goal is to test how the parameters involved in *Lifelong Context Learning* impact the learning performances. Figure 7.16 shows the hidden model and the learned *local models* by the *Active Learning Strategy*, the *Active Cooperative Learning Strategy* and the *Self-Learning Strategy* with the same amounts of *learning situations*.



(a) Hidden Noisy Model (b) *Active Learning Strategy* Learned Models (c) *Active Cooperative Learning Strategy* Learned Models (d) *Self-Learning Strategy* Learned Models

Figure 7.16 – Captures of the 2 Perceptions Noisy Model Problem

Protocol

In this experiment three learning strategies are proposed. The *Active Learning Strategy*, the *Self-Learning Strategy* and the *Active Cooperative Learning Strategy*. The *Active Cooperative Learning Strategy* is the *Active Learning Strategy* with the *Cooperative Neighborhood Learning*

mechanism. The used *hidden function* is the same as the section 7.5.1 but with the addition of noise (Fig. 7.16a).

Noise is added to the *perceptions* and to the *prediction vectors* of the oracle. The noise is generated with a normal distribution $\mathcal{N}(\mu, \sigma^2)$. $\mu = p_i$ for the noisy *perceptions* and $\mu = o_k$ for the noisy *prediction vectors*. σ is set using the empirical rule 7.5.

$$\begin{aligned} \Pr(p_i - 2\sigma \leq p_i^{noisy} \leq p_i + 2\sigma) &\approx 0.9545 \\ \Pr(o_k - 2\sigma \leq o_k^{noisy} \leq o_k + 2\sigma) &\approx 0.9545 \end{aligned} \tag{7.5}$$

Table 7.6 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1.

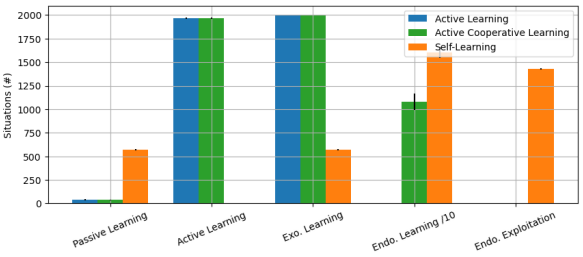
	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_\#$	> 0	$\mathbb{N}$	2000
	number of exploitation situations	$\mathcal{E}_\#$	> 0	$\mathbb{N}$	250
	number of learning episodes	$\mathcal{E}ps_\#$	> 0	$\mathbb{N}$	15
	noise deviation	2σ	$\geqslant 0$	$\mathbb{R}$	1.0
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$	0.04
	model error margin	m^f_{err}	> 0	$\mathbb{R}$	1.0

Table 7.6 – Table of experimental parameters for the noisy experiment.

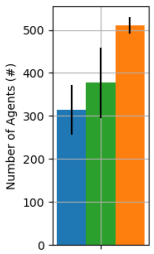
Results

Figure 7.16 shows that the *Active Learning Strategy* and the *Active Cooperative Learning Strategy* have great exploration difficulties as their poor exploration attests. Figure 7.17 shows the learning metrics for the noisy model learning problem for the *Active Learning Strategy* (blue label), the *Active Cooperative Learning Strategy* (green label) and the *Self-Learning Strategy* (orange label).

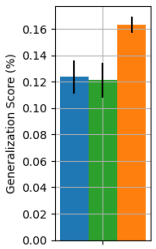
The figure 7.8a shows that the *Active Cooperative Learning Strategy* uses the same *learning situations* as the *Active Learning Strategy* with the addition of *endogenous learning situations*. The addition of noise in the *exogenous learning situations* leads to worse prediction error for the *Active Learning Strategies* (Fig. 7.17d). This is the consequence of the lack of exploration due to the noise. Most of the space is unexplored with the *Active Learning Strategies* (Fig. 7.17e). The noise in the *perceptions* generates more *Conflict NCS* than usual which slows the exploration (Fig. 7.17f). The addition of the *endogenous learning situations* enables to reduce the number of *Range Ambiguity NCS* and thus decrease the volume of incompetencies (Fig. 7.17e), increase the number of *Context Agents* (Fig. 7.17b) and still augment the *generalization score* (Fig. 7.17c). The *Self-Learning Strategy* is impacted by the noise only for the *passive exogenous learning situations* because the *learning inaccuracies* are solved with *endogenous exploitation situations* that are not impacted by the oracle's noise.



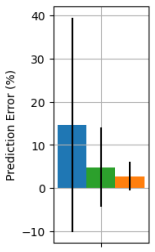
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



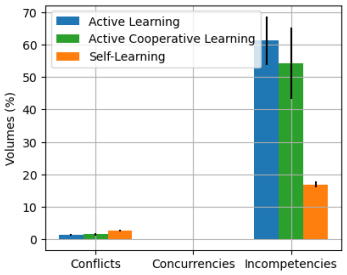
(b) Number of *Context Agents* n_{Ctxt} (#)



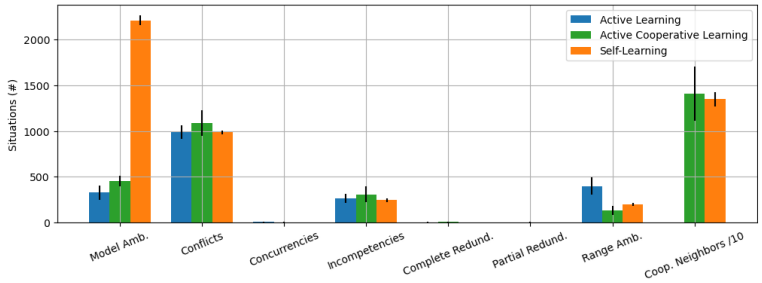
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

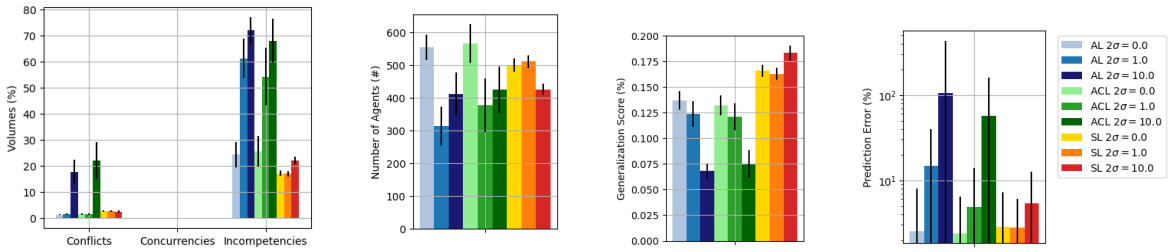
Figure 7.17 – Metrics results on the noisy learning problem with the *Active Learning Strategy*, the *Active Cooperative Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.6 and 7.1.

Noise Sensibility

Figure 7.18 presents an evaluation of some metrics on different *noise deviations* for the noisy problem. When there isn't any noise ($2\sigma = 0$), the exploration of the space is similar for all the learning strategies (Fig. 7.18a) and the prediction errors are of the same order of magnitude (Fig. 7.18d). It can be noticed that without noise, the *Active Cooperative Learning Strategy* is the strategy that reaches the lowest prediction error. For high *noise deviation* values, there is an increase of the conflict and incompetent volumes and the prediction error sharply increases. However, the *Self-Learning Strategy* appears to be more robust to noise as the increase in the prediction error is much lower.

Validity Range Precision Sensibility

Figure 7.19 presents an evaluation of some metrics on different *validity ranges precisions* for the noisy problem. It can be seen that for the active strategies, the better results in terms of exploration (Fig. 7.19a), generalization (Fig. 7.19c) and prediction (Fig. 7.19d) are obtained with the bigger *validity ranges precisions*. For the *Self-Learning Strategy*, it is the same behavior except for the prediction error that is better for small *validity ranges precisions*.



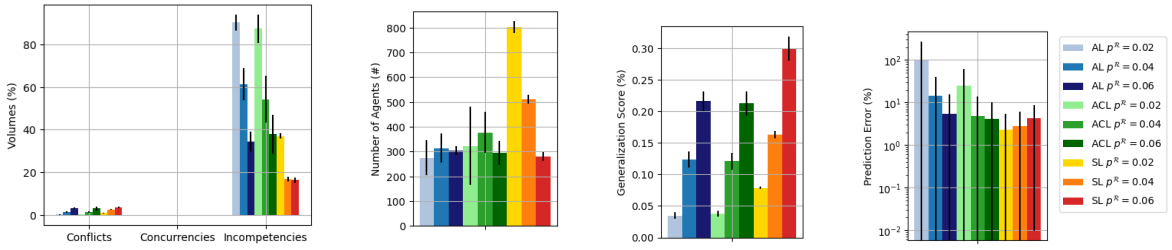
(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)

(b) Number of *Context Agents* n_{Ctxt} (#)

(c) Generalization score $\mathcal{G}^{scr}$ (%)

(d) Prediction error $\mathcal{O}_{Err}$ (%)

Figure 7.18 – Metrics results on the noisy learning problem with the *Active Learning Strategy* (AL), the *Active Cooperative Learning Strategy* (ACL) and the *Self-Learning Strategy* (SL) with different *noise deviation* $2\sigma = \{0; 1; 10\}$. The other parameters are set to the values of the tables 7.6 and 7.1.



(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)

(b) Number of *Context Agents* n_{Ctxt} (#)

(c) Generalization score $\mathcal{G}^{scr}$ (%)

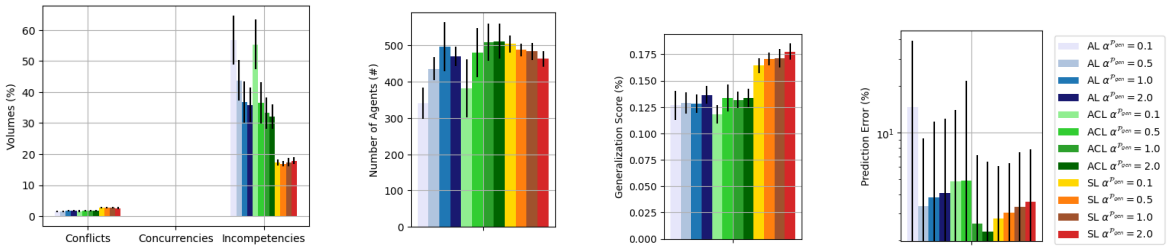
(d) Prediction error $\mathcal{O}_{Err}$ (%)

Figure 7.19 – Metrics results on the noisy learning problem with the *Active Learning Strategy* (AL), the *Active Cooperative Learning Strategy* (ACL) and the *Self-Learning Strategy* (SL) with different *validity ranges precisions* $p^{\mathcal{R}} = \{0.02; 0.04; 0.06\}$. The other parameters are set to the values of the tables 7.6 and 7.1.

Perceptions Generation Coefficient Sensibility

Figure 7.20 presents an evaluation of some metrics on different *perceptions generation coefficients* for the noisy problem. The *perceptions generation coefficient* determines the range in which the artificial *perceptions* are generated. It is noted that for the active strategies, the highest *perceptions generation coefficients* enables to decrease the incompetent volumes (Fig. 7.20a) but it increases the number of *Context Agents* (Fig. 7.20b). Concerning the prediction error (Fig. 7.20d), from the tested coefficients, the best result is obtained for the *Active Cooperative Learning Strategy* with $\alpha^{\mathcal{P}_{gen}} = 2.0$. It means that the artificial perceptions are generated beyond the entire *validity ranges* of the *Context Agents*. For the *Self-Learning Strategy*, higher *perceptions generation coefficients* result in higher *generalization scores* (Fig. 7.20c), but worse prediction errors (Fig. 7.20d).

This difference between the strategies makes sense because the *Self-Learning Strategy* uses *exogenous learning situations* for the centers of the *Context Agents* and the *endogenous learning*



(a) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)

(b) Number of *Context Agents* n_{Ctxt} (#)

(c) Generalization score $\mathcal{G}^{scr}$ (%)

(d) Prediction error $\mathcal{O}_{Err}$ (%)

Figure 7.20 – Metrics results on the noisy learning problem with the *Active Learning Strategy* (AL), the *Active Cooperative Learning Strategy* (ACL) and the *Self-Learning Strategy* (SL) with different *perceptions generation coefficients* $\alpha^{\mathcal{P}_{gen}} = \{0.1; 0.5; 1; 2\}$. The other parameters are set to the values of the tables 7.6 and 7.1.

situations purpose is to enhance their frontiers by keeping the center predictions as little changed as possible. By generating artificial *learning situations* close to the center of *validity ranges*, the *exogenous learning situation* is protected. The active strategies focus on the entire *validity ranges* of the *Context Agents*. With the *Active Learning Strategy*, generating artificial *learning situations* to close to the center ($\alpha^{\mathcal{P}_{gen}} = 0.1$) or to far from the borders ($\alpha^{\mathcal{P}_{gen}} = 2$) gives weight on predictions that do not properly represent the confidence zone of *Context Agents*. Surprisingly, the *Active Cooperative Learning Strategy* still benefits from distant artificial *learning situations*.

Synthesis

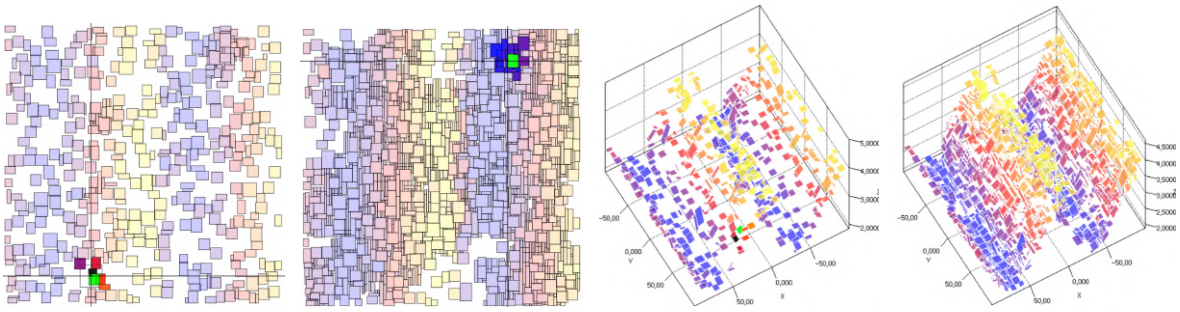
This experiment showed that the *Self-Learning Strategy* is better suited for noisy *learning situations*. The *Active Learning Strategy* is very sensitive to noise but with the *Cooperative Neighborhood Learning* mechanism, it behaves better. The learning strategies perform differently in relation the *validity ranges precision* and the *perceptions generation coefficient* because of the way they process the *learning situations*. The *Self-Learning Strategy* performs better with low *validity ranges* and $\alpha^{\mathcal{P}_{gen}}$. The *Active Learning Strategy* performs better with large *validity ranges* and medium $\alpha^{\mathcal{P}_{gen}}$. The *Active Cooperative Learning Strategy* performs better with large *validity ranges* and $\alpha^{\mathcal{P}_{gen}}$.

7.6.2 Lifelong Exploitation Problem

In this section, another experiment on *Lifelong Learning* is conducted but more in the sense of *Lifelong Exploitation*. As the *Self-Learning Strategy* uses *endogenous learning situations* and *endogenous exploitation situations*, the learning mechanism can keep on improving the *local models* while it is being exploited.

Objectives

The objective of this experiment is to highlight the abilities of *ELLSA* to keep working and generating knowledge during an exploitation. During the exploitation, the learning mechanism continues to seek *learning inaccuracies* and to enhance the continuities and discontinuities between the *local models* thanks to the *CNL* mechanism. Figure 7.21 shows captures of the *validity ranges* and the *local models* before and after the *Lifelong Exploitation*.



(a) *Validity ranges* Before Exploitation (b) *Validity ranges* After Exploitation (c) *Local models* Before Exploitation (d) *Local models* After Exploitation

Figure 7.21 – Capture of the 2 Perceptions Lifelong Exploitation Problem

Protocol

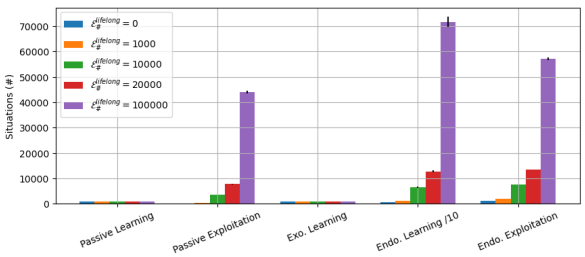
The *Self-Learning Strategy* is employed in this experiment. The used *hidden function* is the same as the section 7.5.2. Table 7.7 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1. The *hidden function* is learned with 2000 learning cycles. *Exploitation situations* $\mathcal{E}_{\#}^{lifelong}$ are requested to the learning mechanism to simulate a lifelong exploitation. 250 additional *exploitation situations* are used to calculate the prediction error.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	2000
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	$(\mathcal{E}_{\#}^{lifelong})$ 250
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0,1[$	$\mathbb{R}$	0.02
	model error margin	$m_{err}^{\dagger}$	> 0	$\mathbb{R}$	0.05

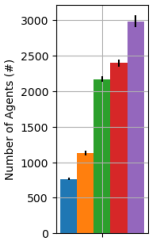
Table 7.7 – Table of experimental parameters for the transfer experiment.

Results

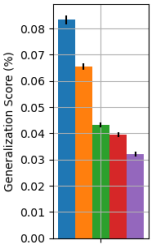
Figure 7.22 shows the learning metrics for the lifelong exploitation learning problem for the *Self-Learning Strategy* for several numbers of *exploitation situations* ($\mathcal{E}_{\#}^{lifelong} = \{0; 1000; 10000; 20000; 100000\}$).



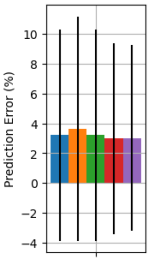
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



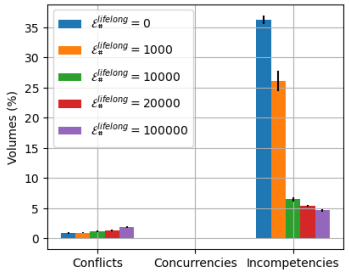
(b) Number of *Context Agents* n_{Ctxt} (#)



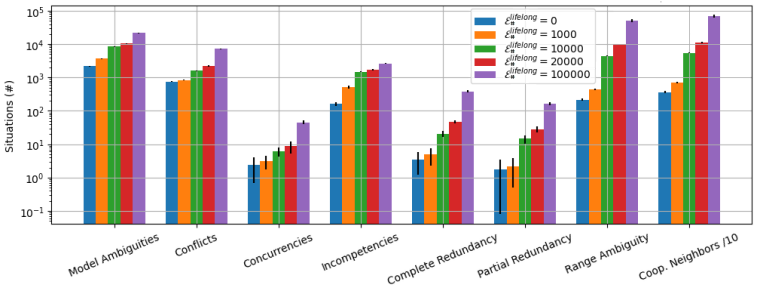
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



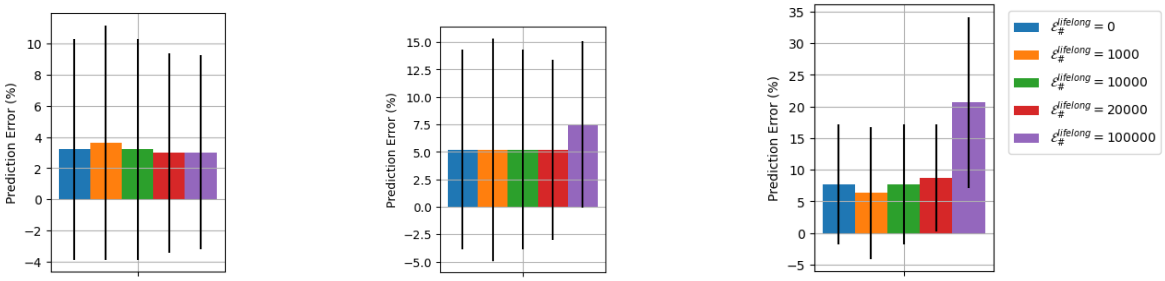
(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.22 – Metrics results on the lifelong exploitation learning problem *Self-Learning Strategy* with different numbers of *exploitation situations* $\mathcal{E}_{\#}^{lifelong} = \{0; 1000; 10000; 20000; 100000\}$. The parameters are set to the values of the tables 7.7 and 7.1.

Figure 7.22a shows that during the exploitation cycles, *endogenous exploitation situations* and *endogenous learning situations* are generated. On the figure 7.22f, the *endogenous exploitation situations* concern the *Conflict NCS*, the *Concurrency NCS*, the *Incompetence NCS* and the *Range Ambiguity NCS*; the *endogenous learning situations* concern the *Model Ambiguity NCS*, the *Incompetence NCS* and the *cooperative neighbors*. Figure 7.22e shows that the lifelong active exploitation slightly increases the volume of conflicts but it significantly decreases the volume of incompetencies as it can also be seen on the figure 7.21. This implies the creation of much more *Context Agents* (Fig. 7.22b) and lower *generalization scores* (Fig. 7.22c). The additional *Complete Redundancy NCS* and *Partial Redundancy NCS* (Fig. 7.22f) do not enable to increase the generalization. Concerning the prediction performances, a slight error increase is observed with 1000 *exploitation situations* but with more *exploitation situations*, the exploitation phase manages to reduce the prediction error (Fig. 7.22d).

Parameters Sensibilities

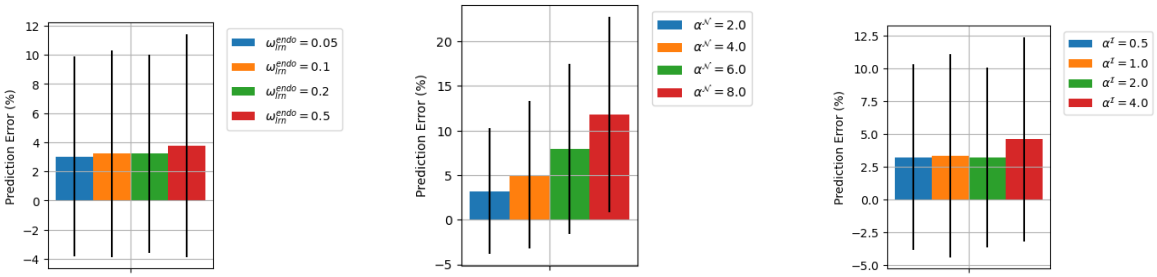
Figure 7.23 details how the prediction error is impacted by the *validity ranges precisions* size with different numbers of *exploitation situations*. It can be seen that for larger *validity ranges precisions*, the exploitation phase provides better performances at first, but it gets worse as there are more *Exploitation situations* (Figures 7.23band 7.23c).



(a) $p^{\mathcal{R}} = 0.02$ (b) $p^{\mathcal{R}} = 0.04$ (c) $p^{\mathcal{R}} = 0.06$

Figure 7.23 – Prediction error $\mathcal{O}_{Err}$ (%) with the *Self-Learning Strategy* with several *validity ranges precisions* $p^{\mathcal{R}}$ and different numbers of *exploitation situations* $\mathcal{E}_{\#}^{lifelong} = \{0; 1000; 10000; 20000; 100000\}$. The parameters are set to the values of the tables 7.7 and 7.1.

Figure 7.24 presents the impact on the prediction error of the *endogenous learning weight*, the *neighborhood radius coefficient* and the *influence radius coefficient*. The *endogenous learning weight* and the *influence radius coefficient* have low impact but tend to increase the error for the highest tested values (Figures 7.24a and 7.24c). The *neighborhood radius coefficient* clearly increases the prediction error when it augments (Fig. 7.24b).



(a) $w_{lrn}^{endo} = \{0.05; 0.1; 0.2; 0.5\}$ (b) $\alpha^{\mathcal{N}} = \{2; 4; 8\}$ (c) $\alpha^{\mathcal{I}} = \{0.5; 1; 2; 4\}$

Figure 7.24 – Prediction error $\mathcal{O}_{Err}$ (%) with the *Self-Learning Strategy*; sensibility to the *endogenous learning weight* w_{lrn}^{endo}, the *neighborhood radius coefficient* $\alpha^{\mathcal{N}}$ and the *influence radius coefficient* $\alpha^{\mathcal{I}}$; *exploitation situations* $\mathcal{E}_{\#}^{lifelong} = 10000$. The other parameters are set to the values of the tables 7.7 and 7.1.

Synthesis

This experiment showed that the *Self-Learning Strategy* enables to enhance the prediction performances during an exploitation phase where no *exogenous learning situations* are provided. This improvement is dependent of the *validity ranges precision*. For larger *validity ranges precisions*, the observed improvements turn into deterioration when *exploitation situations* grow.

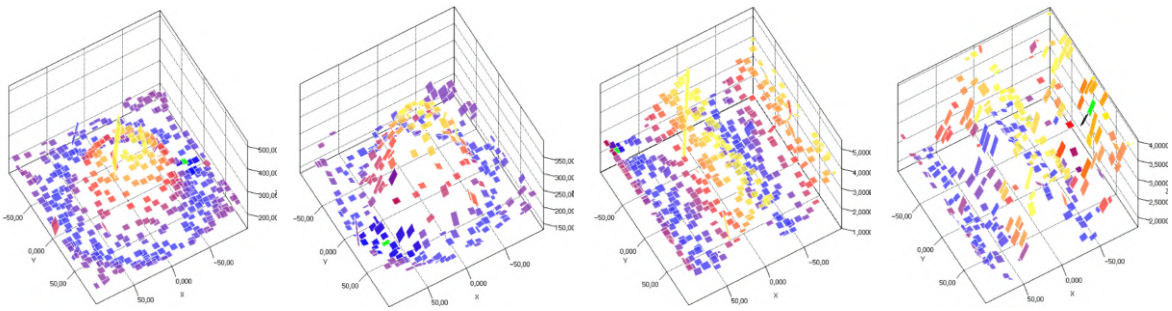
In this setting, the enlargement of the neighborhood and the influence areas did not enable better prediction performances. This shows that for this model, larger *neighborhoods* do not bring useful information.

7.7 Few Learning Situations

In this section the experimentation with few *learning situations* and the *Self-Learning Strategy* is presented. It is compared to a *Naive Learning Strategy* with no neighborhood abilities.

Objectives

The objective of this experiment is to show that few *learning situations* are enough to obtain an exploitable collective of *Context Agents* with and without discontinuities. Another goal is to show that *Self-Learning Strategy* can provide better learning performances than the *Naive Learning Strategy*. Figure 7.25 shows captures of the learned *local models* with the *Naive* and *Self-Learning Strategies* for the non linear continuous model (NLC) and the non linear discontinuous model (NLD).



(a) NLC Model, Learned *local models* with the *Naive Learning Strategy*

(b) NLC Model, Learned *local models* with the *Self-Learning Strategy*

(c) NLD Model, Learned *local models* with the *Naive Learning Strategy*

(d) NLD Model, Learned *local models* with the *Self-Learning Strategy*

Figure 7.25 – Captures of the 2 perceptions non linear continuous (NLC) and discontinuous (NLD) problems and their learning with 500 *learning situations* with the *Naive Learning Strategy* and the *Self-Learning Strategy*.

Protocol

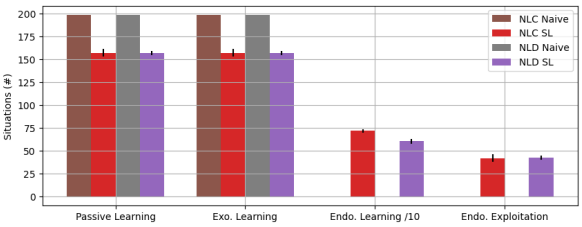
The *Self-Learning Strategy* and the *Naive Learning Strategy* are used on two different *hidden functions*. The continuous non linear problem (Sec. 7.5.1) and the discontinuous non linear problem (Sec. 7.5.2). The *Naive Learning Strategy* is the one presented section 7.5. It learn with *passive learning situations*. It does not use any *learning inaccuracies NCS* and all the mechanisms that involve *Neighbor Context Agents* are not used. Table 7.8 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1. In this experiment, *Context Agents* are not created with the *Neighbor Context Agents* like is usually done in the previous evaluations of the *Self-Learning Strategy*.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	200
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	250
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	0.02
	model error margin	m_{err}^{f}	> 0	$\mathbb{R}$	1.0 / 0.05

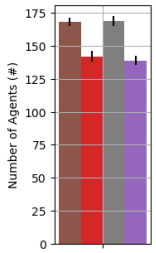
Table 7.8 – Table of experimental parameters for the few learning experiment.

Results

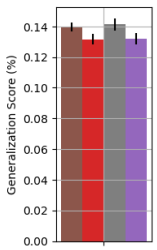
Figure 7.26 shows the learning metrics for the learning problem with few *learning situations* for the *Self-Learning Strategy* and the *Naive Learning Strategy* (brown label: NLC model and *Naive Learning Strategy*; red label: NLC model and *Self-Learning Strategy*; gray label: NLD model and *Naive Learning Strategy*; purple label: NLD model and *Self-Learning Strategy*).



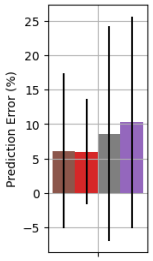
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



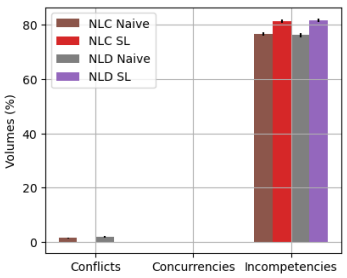
(b) Number of *Context Agents* n_{Ctxt} (#)



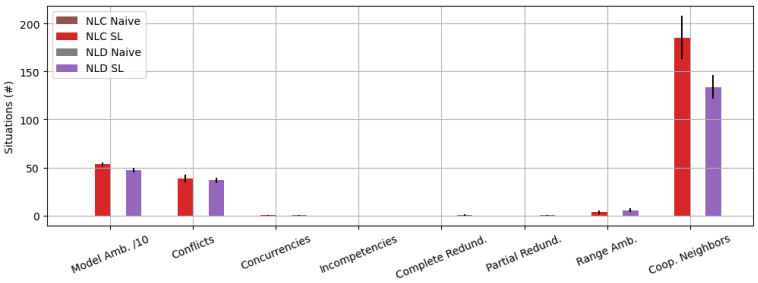
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

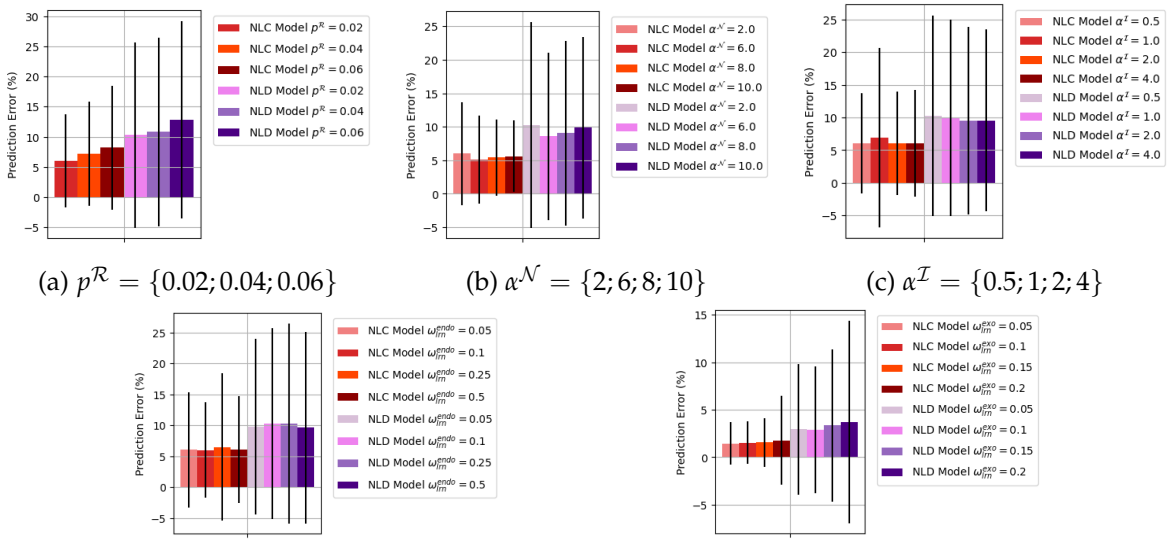
Figure 7.26 – Metrics results on the non linear continuous (NLC) and discontinuous (NLD) learning problems with few *learning situations*; The *Self-Learning Strategy* and *Naive Learning Strategy* are compared. The other parameters are set to the values of the tables 7.8 and 7.1.

Figure 7.26a shows that even for a low number of *learning situations*, *endogenous exploitation situations* are generated to enhance the *local models* for the *Self-Learning Strategy*. They are mostly *Conflict NCS* (Fig. 7.26f). In this case, most of the space is unexplored and filled with incompetent volumes (Fig. 7.26e). As the *Naive Learning Strategy* does not generate *endoge-*

nous exploitation situations, it slightly more explores the space and generates more *Context Agents* (Fig. 7.26b). Concerning the prediction and the non linear continuous problem, it can be noticed that the *Self-Learning Strategy* decreases the prediction error deviation when compared with the *Naive Learning Strategy* (red and brow labels on the figure 7.26d). For the non linear discontinuous problem, the *Self-Learning Strategy* does not improves the prediction error as experienced section 7.5.2.

Parameters Sensibilities

Figure 7.27 presents an evaluation of the prediction error by varying several parameters.



(a) $p^{\mathcal{R}} = \{0.02; 0.04; 0.06\}$ (b) $\alpha^{\mathcal{N}} = \{2; 6; 8; 10\}$ (c) $\alpha^{\mathcal{I}} = \{0.5; 1; 2; 4\}$

(d) $w_{lrn}^{endo} = \{0.05; 0.1; 0.25; 0.5\}$ (e) $w_{lrn}^{exo} = \{0.05; 0.1; 0.15; 0.2\}$
$\mathcal{L}_{\#} = 4000$

Figure 7.27 – Prediction error $\mathcal{O}_{Err}$ (%) with the *Self-Learning Strategy*; sensibility to the *validity ranges precision* $p^{\mathcal{R}}$, the *neighborhood radius coefficient* $\alpha^{\mathcal{N}}$ and the *influence radius coefficient* $\alpha^{\mathcal{I}}$, the *endogenous learning weight* w_{lrn}^{endo} and the *exogenous learning weight*. The other parameters are set to the values of the tables 7.8 and 7.1.

Figure 7.27a shows that enlarging the *validity ranges precision* with few *learning situations* increases the prediction error. Concerning the *neighborhood radius coefficient*, it seems to exist an optimal value ($\alpha^{\mathcal{N}} = 6$ in this case) for both problems (Fig. 7.27b). The *influence radius coefficient* decreases the prediction error when it grows but the highest value gives a worse prediction error for the discontinuous model (Fig. 7.27c). The *endogenous learning weight* has a weak impact on the prediction error. An increasing trend of prediction error can be guessed (Fig. 7.27d). To assess the *exogenous learning weight*, 4000 *learning situations* are necessary to observe variations. Figure 7.27e shows that higher *exogenous learning weights* worsen the prediction error.

Synthesis

This experiment showed that a low random exploration of the space can provide prediction on the whole space, even the unexplored areas. The *Self-Learning Strategy* was able to enhance the learning process with few examples for the non linear continuous model.

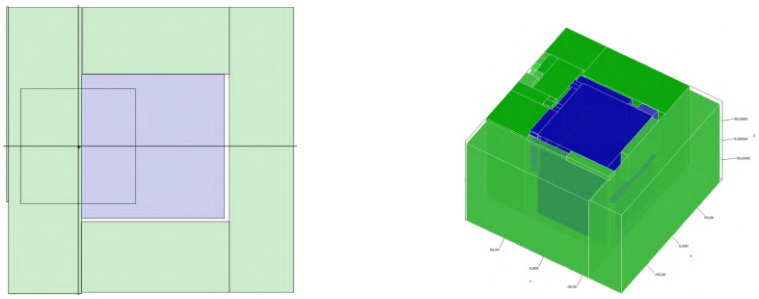
For these learning problems, enlarging the explored space by enlarging the *validity ranges precision* does not bring better prediction error as it could have been thought. Varying the *neighborhood radius coefficient* and the *influence radius coefficient* enable to refine the prediction error. Adding weight on the *endogenous learning situations* and the *exogenous learning situations* only leads to worse predictions.

7.8 Scalability

In this section, the scalability of the learning mechanisms is assessed with the *Self-Learning Strategy* and the *Active Learning Strategy*.

Objectives

The objective of this experiment is to evaluate how the *Self-Learning Strategy* and the *Active Learning Strategy* behave at higher dimensions and with large quantities of *Context Agents*. The used hidden model is the toy problem of section 7.4. Figure 7.28 shows captures of learned *validity ranges* for 2 and 3 *perceptions*.



(a) 2 *perceptions* (b) 3 *perceptions*

Figure 7.28 – Captures of learned *validity ranges* for the toy learning problem

Protocol

In this experiment, the *hidden function* of section 7.4 is used for several *perceptions* (2, 3, 5 and 10).

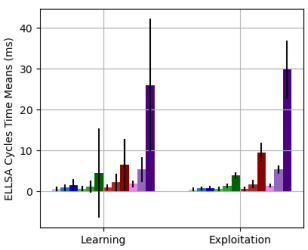
Table 7.9 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	$2; 3; 5; 10$
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	$500; 2000; 10000$
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	250
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	$0.06; 0.1$
	model error margin	m^{f}_{err}	> 0	$\mathbb{R}$	1

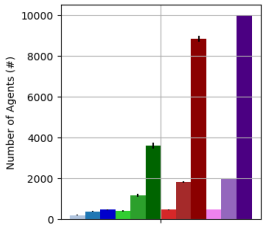
Table 7.9 – Table of experimental parameters to evaluate scalability.

Results

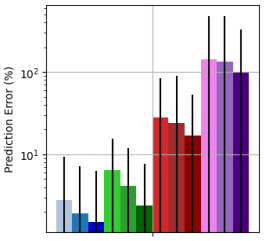
Figure 7.29 shows the learning metrics for the toy problem with the *Self-Learning Strategy* and several *perceptions*. Figure 7.29d shows that for higher *perceptions*, filling the space



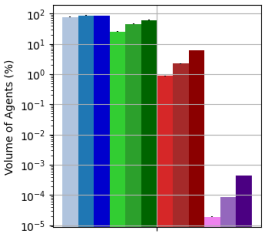
(a) Execution times of learning and exploitation cycles (ms)



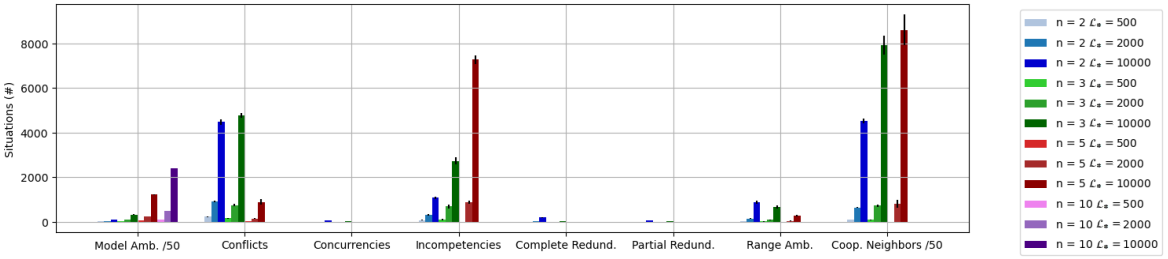
(b) Number of *Context Agents* n_{Ctxt} (#)



(c) Prediction error $\mathcal{O}_{Err}$ (%)



(d) Volume of *Context Agents* $\mathcal{V}_{Ctxt}{}'$ (%)

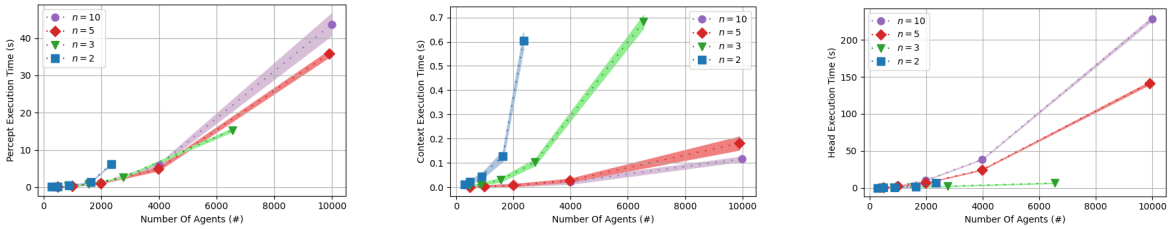


(e) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.29 – Metrics results on the square problems with the *Self-Learning Strategy* and several *perceptions*. $p^{\mathcal{R}} = 0.06$. The other parameters are set to the values of the tables 7.9 and 7.1.

with *Context Agents* requires much larger amounts of *learning situations*. It explains the much worst prediction errors obtained figure 7.29c. The learning and exploitation execution times grow with the higher dimensions and with larger amounts of *learning situations* (Fig. 7.29a). With this strategy, larger amounts of *learning situations* always generate more *Context Agents* (Fig. 7.29b). Figure 7.29e shows how the *learning inaccuracies NCS* are affected by higher dimensions. For the same amount of *learning situations*, there are more *Model Ambiguity NCS* with higher *perceptions*. It make sense because as linear regression is used, additional *learning situations* are required when *perceptions* grow. It can be noticed that there are less *Conflict NCS* for 5 *perceptions* and there aren't any for 10 *perceptions*. Indeed, for higher *perceptions*, *Context Agents* are more distant. *Neighbor Context Agents* are required to generate *Incompetence NCS* with the *Self-Learning Strategy* and for 10 *perceptions* none are generated as well as no *Cooperative Neighborhood Learning* situations are produced. This phenomenon is also observable with the amount *Range Ambiguity NCS* that decreases with higher dimensions. Indeed, proximity between *Context Agents* is essential for the *Range Ambiguity NCS* detection.

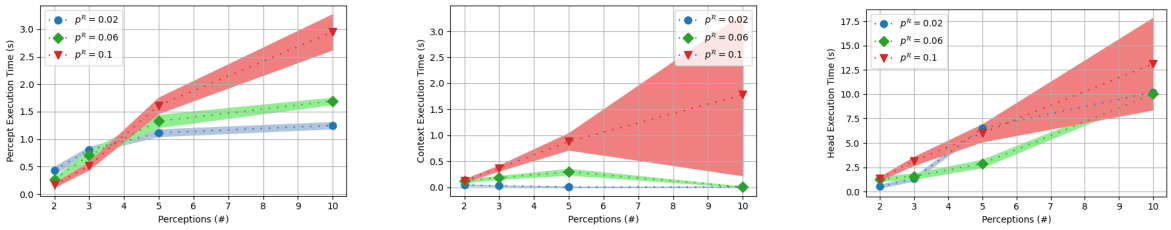
Figure 7.30 plots the execution time sums of each agents types depending on the number of agents and *perceptions*. The *Percept Agents* time execution follow a polynomial trend regardless of the number of *perceptions* (Fig. 7.30a). The *Context Agents* time execution also follow polynomial curves but low *perceptions* lead to longer execution times (Fig. 7.30b). It is explained by the larger amounts of activate *Context Agents* and *Neighbor Context Agents* with low *perceptions*. However the considered duration are negligible compared to the execution time of the *Percept Agents* and the *Head Agent*. The *Head Agent* time execution is the one that



(a) *Percept Agents* (s) (b) *Context Agents* (s) (c) *Head Agent* (s)

Figure 7.30 – Execution time sums results on the square problem depending on the number of *Context Agents* with the *Self-Learning Strategy* and several *perceptions* $n = \{2;3;5;10\}$. $p^{\mathcal{R}} = 0.02$. The other parameters are set to the values of the tables 7.9 and 7.1.

has the most impact and it also matches polynomial behaviors with longer durations for higher *perceptions* (Fig. 7.30c).

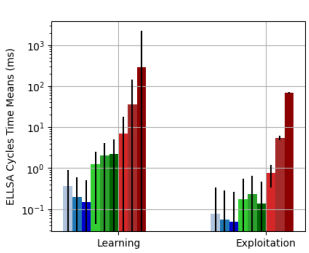


(a) *Percept Agents* (s) (b) *Context Agents* (s) (c) *Head Agent* (s)

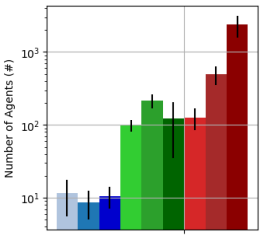
Figure 7.31 – Execution time sums results on the square problem depending on the number of *perceptions* with the *Self-Learning Strategy* and different *validity ranges precisions* $p^{\mathcal{R}} = \{0.02;0.06;0.1\}$. The other parameters are set to the values of the tables 7.9 and 7.1.

Figure 7.31 plots the execution duration sums of agents depending on the *perceptions* with different *validity ranges precisions*. There is a logarithmic dependence of *Percept Agents* execution time on the *perceptions* (Fig. 7.31a). The *Context Agents* execution is linearly dependent on the *perceptions* for $p^{\mathcal{R}} = 0.1$. For lower *validity ranges precisions*, the duration increases until 5 *perceptions* and decreases for 10 *perceptions*. This can be cause by the fewer activated *Context Agents* at high dimensions for small *validity ranges precisions* (Fig. 7.31b). The *Head Agent* time execution is linearly dependent on the *perceptions* and it is regardless of the *validity ranges precision* (Fig. 7.31c).

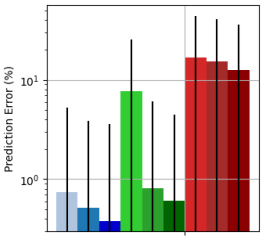
Figure 7.32 shows the learning metrics for the square problem with the *Active Learning Strategy* and several *perceptions*. The number of *perceptions* 10 was not tested with the *Active Learning Strategy* because of too long experimental execution times for high numbers of *learning situations*. The duration of cycles increases with the *perceptions*. But for 2 *perceptions* it decreases with the number of *learning situations*. For 3 *perceptions*, it also decreases with the number of *perceptions* during the exploitation and it stabilizes during the learning (Fig. 7.32a). This is due to the increase in generalization when the number of *Context Agents* decreases (Fig. 7.32b) and the explored volume increases (Fig. 7.32d). Compared to the *Self-Learning Strategy*, the *Active Learning Strategy* generates almost no *Model Ambiguity NCS* because models are more shared during the *Context Agents* creation (Fig. 7.32e). Indeed, there



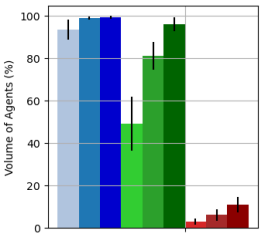
(a) Execution times of learning and exploitation cycles (ms)



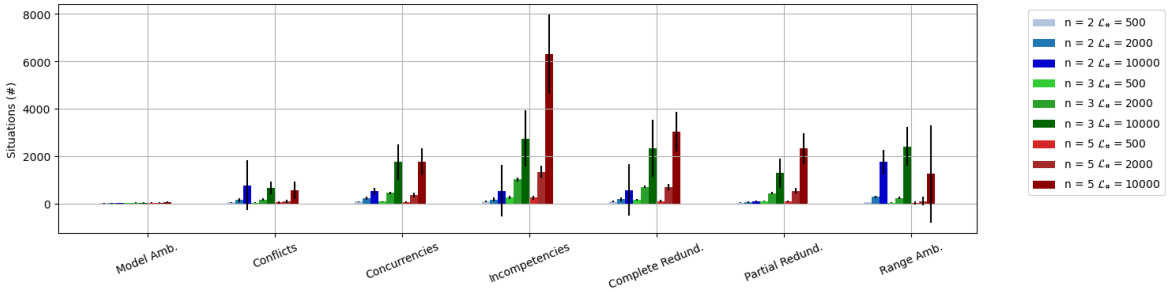
(b) Number of *Context Agents* n_{Ctxt} (#)



(c) Prediction error $\mathcal{O}_{Err}$ (%)



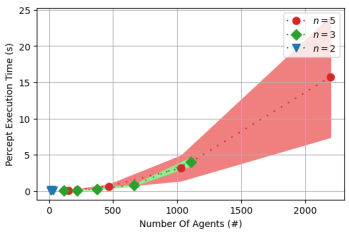
(d) Volume of *Context Agents* $\mathcal{V}_{Ctxt}{}'$ (%)



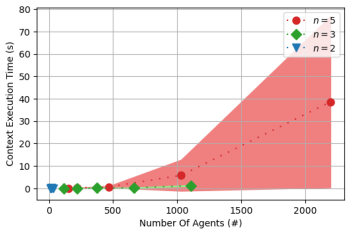
(e) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.32 – Metrics results on the square problems with the *Active Learning Strategy* and several *perceptions*. $p^{\mathcal{R}} = 0.1$ The other parameters are set to the values of the tables 7.9 and 7.1.

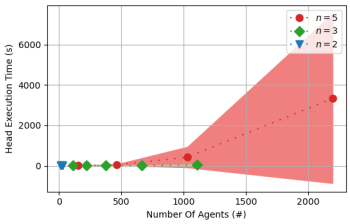
are more similar models as the presence of *Concurrency NCS*, *Complete Redundancy NCS* and *Partial Redundancy NCS* attests. More similarities between *local models* enable to decrease the number of *Context Agents* but for 5 *perceptions* the space is not sufficiently explored to enable generalization. This lack of exploration is also visible on the figure 7.32c where the prediction error is worse for high *perceptions* and few *learning situations*.



(a) *Percept Agents* (s)



(b) *Context Agents* (s)

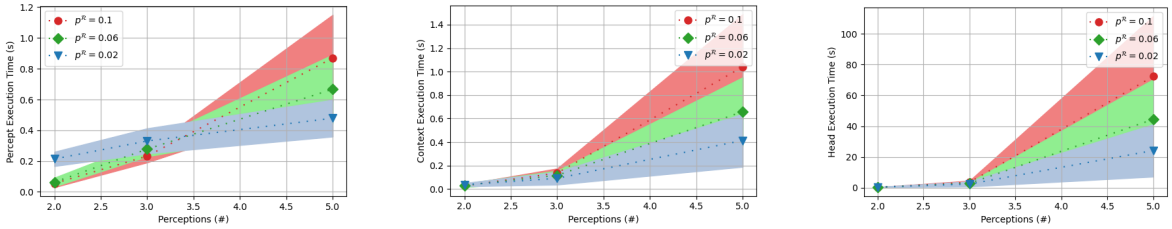


(c) *Head Agent* (s)

Figure 7.33 – Execution time sums results on the square problem depending on the number of *Context Agents* with the *Active Learning Strategy* and several *perceptions* $n = \{2; 3; 5\}$. $p^{\mathcal{R}} = 0.06$. The other parameters are set to the values of the tables 7.9 and 7.1.

Figure 7.33 plots the execution time sums of each agents types depending on the number of agents with different *perceptions* and the *Active Learning Strategy*. All curves follow polynomial trends. For the *Percept Agents*, the number of *perceptions* as no impact (Fig. 7.34a). Compare to the previous strategy, the *Context Agents* cycles take much more time, and there

is a huge increase for 5 *perceptions* (Fig. 7.34b). The agent that is responsible of most of the execution time is again the *Head Agent* but with an increase of more than 10 times the duration of the *Head Agent* with the *Self-Learning Strategy*.



(a) *Percept Agents* (s) (b) *Context Agents* (s) (c) *Head Agent* (s)

Figure 7.34 – Execution time sums results on the square problem depending on the number of *perceptions* with the *Active Learning Strategy* and different *validity ranges precisions* $p^{\mathcal{R}} = \{0.02; 0.06; 0.1\}$. The other parameters are set to the values of the tables 7.9 and 7.1.

Figure 7.34 plots the execution duration sums of agents depending on the *perceptions* with different *validity ranges precisions* and the *Active Learning Strategy*. Depending on the number of *perceptions*, the *Percept Agents* execution duration follow a linear curve (Fig. 7.34a). The *Context Agents* and the *Head Agent* follow a polynomial curve (Figures 7.34b and 7.34c). An increase in time execution is observed for higher *validity ranges precisions* and higher numbers of *perceptions*. This can be due to larger *neighborhoods*. Larger *neighborhoods* involve more *Neighbor Context Agents* and thus more operations when resolving the *NCS*.

Synthesis

In this section, it has been showed that exploration for higher numbers of *perceptions* is an issue. Achieve a full exploration for *perceptions* that are higher than 3 would required huge amounts of *learning situations* and *Context Agents*. This lack of exploration obviously leads to poorer predictions when the whole space is considered.

It has been found that the *neighborhood* and *influence* mechanism are also affected by growing dimensions. From a certain number of *perceptions*, *neighborhoods* are empty since the *Self-Learning Strategy* no longer founds cooperative *Neighbor Context Agents* to generate *endogenous learning situations*.

Using the *Active Learning Strategy*, it has been proven that generalization also works for 3 *perceptions*. However, it would require much more *learning situations* for higher *perceptions*.

The *Percept Agents* optimized activation has proven to be effective as the number of *perceptions* grow. However, *Context Agents* and *Head Agent* execution times have not reach linear dependence on the number of *perceptions* yet. Depending on the number of *Context Agents*, the time performance is not linear either and it behaves differently with the number of *perceptions*.

A huge difference in time execution was noticed between the *Self-Learning Strategy* and the *Active Learning Strategy*. This is what is responsible for the absence of 10 *perceptions* for the *Active Learning Strategy*. The *Active Learning Strategy* increases the execution time by a

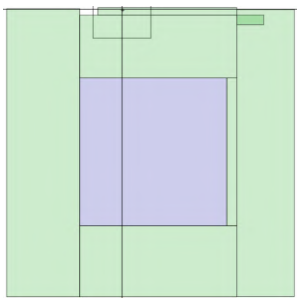
factor of 10. This is certainly due the *Incompetence NCS* that is computationally expensive for higher dimensions and needs further work to be scalable. Otherwise, it can be used less often like in the *Self-Learning Strategy* where the time execution explosion is not observed.

7.9 Transfer Learning

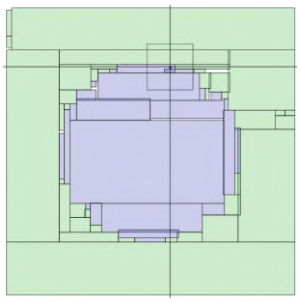
In this section, the ability of the learning mechanism to reuse its knowledge and to adapt to a variation in the search space is tested. The ability to transfer previously learned knowledge during the learning of a new task is an important challenge of *Machine Learning*. The goal is to enable a learning mechanism to generate connections between its past and present experiences without any intervention of the designer.

Objectives

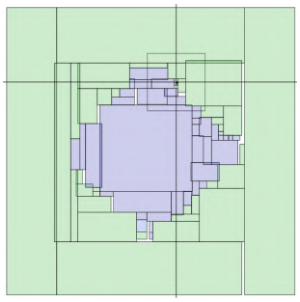
The objective of this experiment is to show that past learned models can be used for a new learning task when they are related to it. The goal of this experiment is also to highlight the adaptive abilities of *ELLSA* to changing models or tasks without any intervention of the learning mechanism creator. Figure 7.35 shows captures of learned *validity ranges* for the 3 hidden models of increasing difficulty that are considered.



(a) Learned *validity ranges* of $\mathcal{F}_1$ (square)



(b) Learned *validity ranges* of $\mathcal{F}_2$ (disc)



(c) Learned *validity ranges* of $\mathcal{F}_3$ (rhombus)

Figure 7.35 – Captures of learned *validity ranges* of 2 *perceptions Transfer Learning* problems.

Protocol

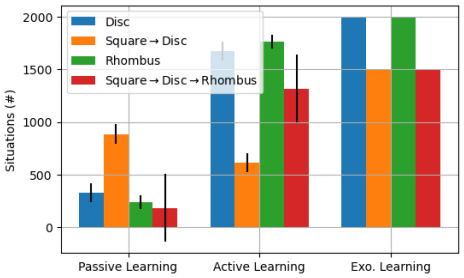
There are three *hidden functions* that represent close learning problems, two *perceptions* and one output. $\mathcal{F}_1$ is the learning problem of the section 7.4 with the shape of a square. $\mathcal{F}_2$ and $\mathcal{F}_3$ are variations of $\mathcal{F}_1$ with the respective shapes of a disc and a rhombus.

Table 7.10 shows the parameters used for this scenario. All other parameters are the ones presented in the table 7.1. When a *hidden function* is learned alone, 2000 learning cycles are used. When is is learned after after another *hidden function*, 1500 learning cycles are provided. $\mathcal{F}_2$ (disc) and $\mathcal{F}_3$ (rhombus) are learned independently. $\mathcal{F}_2$ is learned with 1500 learning cycles after $\mathcal{F}_1$ (square→disk) and $\mathcal{F}_3$ is learned with 1500 learning cycles after $\mathcal{F}_1$ and $\mathcal{F}_2$ (square→disc→rhombus). The *Active Learning Strategy* is used. The results are average over 50 episodes because the behaviors for these learning problems are more fluctuating.

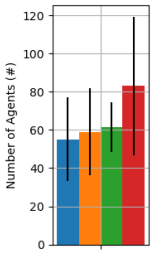
	Name	Notation	Constrains	Domain	Value
Exp.	*number of perceptions*	n	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	2000 (1500)
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	500
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	50
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	0.05
	model error margin	m^{f}_{err}	> 0	$\mathbb{R}$	1

Table 7.10 – Table of experimental parameters for the transfer experiment.

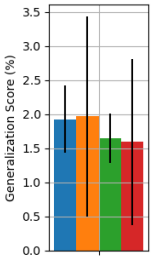
Results



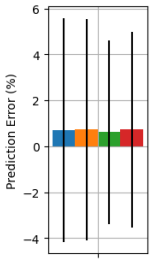
(a) Passive, active, exogenous and endogenous learning and exploitation situations (#)



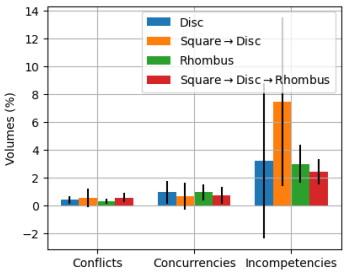
(b) Number of *Context Agents* n_{Ctxt} (#)



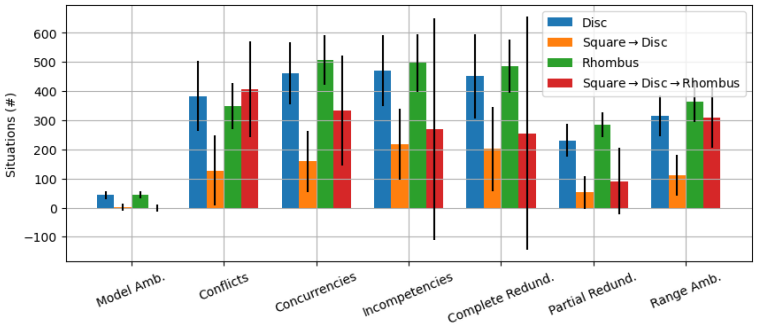
(c) Generalization score $\mathcal{G}^{scr}$ (%)



(d) Prediction error $\mathcal{O}_{Err}$ (%)



(e) Volumes of *learning inaccuracies*: $\mathcal{V}_{Cflt}{}'$, $\mathcal{V}_{Conc}{}'$ and $\mathcal{V}_{Inc}{}'$ (%)



(f) *Learning inaccuracies NCS* counts and *cooperative neighbors* counts.

Figure 7.36 – Metrics results on the *Transfer Learning* problems with the *Active Learning Strategy*. The parameters are set to the values of the tables 7.10 and 7.1.

Figure 7.36 shows the learning metrics for the *Transfer Learning* problem with the *Active Learning Strategy*. The blue label is the learning of the disc ($\mathcal{F}_2$) with 2000 learning cycles. The yellow label is the learning of the square ($\mathcal{F}_1$) with 2000 learning cycles, followed with the learning of the disc ($\mathcal{F}_2$) with 1500 learning cycles. The green label is the learning of the rhombus ($\mathcal{F}_3$) with 2000 learning cycles. The red label is the learning of the square ($\mathcal{F}_1$) with 2000 learning cycles, followed with the learning of the disc ($\mathcal{F}_2$) with 1500 learning cycles, followed with learning of the rhombus ($\mathcal{F}_3$) with 1500 learning cycles. The *NCS*, the learning and the exploitation situations are usually counted during all the learning cycles.

To represent the gain of the previous learning cycles for the scenarios square→disc and square→disc→rhombus, only the last 1500 learning cycles are considered for the counts.

Figure 7.36a shows that when the disc problem is learned after the square problem, much less *active learning situations* are required. All *learning inaccuracies NCS* are generated in less quantities, specially the *Model Ambiguity NCS*. This shows that *local models* are shared to avoid asking unnecessary *active learning situations* (Fig. 7.36f). The *generalization score* mean is slightly improved compared to the learning of the disc problem alone but with deviation increase (Fig. 7.36c). This is due to the highly fluctuating obtained incompetent volumes with the learning of the disk and the square→disc (Fig. 7.36e). It is explained by the generalization obtained with the square problem (Sec. 7.4). When large *Context Agents* update they *validity ranges*, their are more likely to create large void areas. The learning of the square→disc problem manages to reach similar prediction error with less *learning situations* concerning than the disc problem alone (Fig. 7.36d).

After the learning of the square and the disc problems, the rhombus problem is still a difficult problem because it still requires lot of *active learning situations*. The square→disc→rhombus problem generates less *NCS* from all types than the rhombus problem (like the square→disc problem compared to the disc problem) except for *Conflict NCS* (Fig. 7.36f). Learning the square and disc models before the rhombus enables to reach lesser incompetent volumes (Fig. 7.36e), but it generates more *Context Agents* (Fig. 7.36b) and the *generalization score* is not improved (Fig. 7.36c). The prediction error on the square→disc→rhombus problem is neither improved when compared to the learning of rhombus problem alone.

Synthesis

This experiment showed that *ELLSA* is capable of reusing models and sharing them between *Context Agents* to speedup the learning process on two related problems. It also demonstrated that the collective of *Context Agents* can self-adapt in order to adjust to any change in the hidden model. All the learning metrics were not enhanced with the addition of the different past related experiences and it is still an easy learning problem. Additional mechanism need to be investigate in order to reach better transfer of knowledge between *Context Agents*. However, this experiment confirms that *Context Learning* provides undeniable flexibility properties for the manipulated models.

7.10 Synthesis

To conclude on the mathematical models experiments, the objectives of section 1.6 are reminded and a discussion is provided with respect to them.

- ***Endogenous Feedback***. The experiment section 7.4 validated the implementation of a part of the implemented *Endogenous Feedback*: the detection and resolution of *learning inaccuracies* with *active learning situations*. The *Endogenous Feedback* were also observed along all the different experiments. The experiments using the *Self-Learning Strategy* showed that the *learning inaccuracies* can also be resolved with *endogenous exploitation situations* that do not required predictions from an oracle. Another *Endogenous Feedback* is the *endogenous learning situations*. When using the *Self-Learning Strategy*, the *endogenous learning situations* enabled to enhance the learning metric by using less *learning situations*. One issue that remains is that for high numbers of *perceptions*, *endogenous learning situations* are harder to generate because the *Context Agents* do not perceive any neighbors. Additional experiments concerning the generation of *endogenous learning situations* will be conducted in the next chapter (8).
- ***Agnosticity***. The experiments that were conducted in this chapter were intentionally independent from any domain application and focused on properties of linearity, non linearity, continuity and discontinuity. Section 7.5.3 showed that the *Active Learning Strategy* and the *Self-Learning Strategy* performed better than a naive learning strategy on a hidden model composed of all these properties.
- ***Lifelong Learning***. The experiments on *Lifelong Learning* gathered noisy problems and lifelong exploitation problems. It was experimentally proven that the *Self-Learning Strategy* is well suited for learning problems with noise in the data. The *Self-Learning Strategy* also enabled lifelong active exploitation during which the learning mechanism continues enhancing its models without *exogenous learning situations*.
- ***Online Learning***. The learning mechanisms learns by definition in an online manner as each *learning situation* is provided sequentially. The experiment that most represents this property is detailed section 7.9. In this experiment several learning and exploitation phases follow one another when the hidden model is changed. Previous relevant experiences are kept and exploited as the new *learning situations* are used to modify the models that need to be changed.
- ***Self-Observation***. The *Self-Observation* was highlighted through the detection of all *learning inaccuracies* during the *Active Learning Strategy* and the *Self-Learning Strategy*. A track that has not be treated is to use the *confidence* of *Context Agents* to strengthen their *local models* when their *confidence* is low. Add additional *learning inaccuracy* could be to target *Context Agents* with the lowest *confidences*.
- ***Knowledge Generalization***. This objective was addressed all along the experiments by observing the *generalization score*. *Knowledge Generalization* in the case of *Context Learning* is dependent on the measure of similarity between *local models* and on the disposition of *validity ranges* (*Context Agents* can merge only when their *validity ranges* are aligned). The *Active Learning Strategy* enabled better generalization than the *Self-Learning Strategy*. Indeed, the *Active Learning Strategy* locally converges faster towards the hidden

model with the additional *learning situations* that are requested at the creation of a *Context Agent*. The *Self-Learning Strategy* uses the *CNL* mechanism to locally converge towards the hidden model. When generalization is possible, the *CNL* mechanism does not converge fast enough toward the hidden models to permit the same generalization than the *Active Learning Strategy*. When the number of *perceptions* grows, the generalization is achieved until 3 *perceptions*. Beyond, much more *learning situations* are necessary. *Knowledge Generalization* in the sense of *Transfer Learning* was assessed section 7.9. It showed that when the learning tasks are related, past experiences are helpful for the new task to be learned but the transfer of information could be more important.

- ***Scalability***. The objective was assessed with the two learning strategies section 7.8. It was showed that the exploration of spaces with high numbers of *perceptions* is an issue because full exploration is impossible. Moreover, for high numbers of *perceptions*, the *neighborhood* mechanism needs to be enhanced because no *Neighbor Context Agents* are detected. A good point is that the execution times of the agents follow logarithmic, linear and polynomial dependencies on the number of *Context Agents* and *perceptions*.
- ***Any Data Amount***. The *Self-Learning Strategy* is intended for learning with few *learning situations* thanks to *Cooperative Neighborhood Learning*. The model updating mechanism implemented with *Lifelong Context Learning* is intended for learning in a lifelong setting where many *learning situations* can be provided. The experiments with few *learning situations* and higher numbers of *perceptions* enabled to validate both cases.
- ***Explainability***. This objective was not validated with a metric measure. However, the implemented interface and the ease of measuring metrics shows that *Context Learning* enables to explain how is the learning is modeled, represented and structured. This is provided by the embedding of the *local models* into agents.
- ***Robotic Application***. Robots evolve in the real world which is subject to noise in the perceived data. It is in this objective that the experiment with noise was conducted. It showed that the *Active Learning Strategy* is very affected by noise in the *perceptions* and the *prediction vectors*. However, thanks to the *CNL* mechanism, the *Self-Learning Strategy* provides robustness to noise. The next chapter will present a concrete robotic application using *ELLSA*.

8 Robotic Problems

In this chapter, I present several robotic experimentations made with the learning mechanism ELLSA. The learning is applied to the problem of Inverse Kinematics which is classical in the robotic community. The application scenario is the control learning for multi-joint robotic arms. These experimentations allow to highlight the properties of the system according to the objectives presented section 1.6 in a concrete problem.

ROBOTIC arms or industrial robots are widely used nowadays. They enable automation for mass production, they can provide human assistance in the medical field or human replacement for all kind of manipulations where humans can't handle it. The International Federation of Robotics (IFR) reported a record of 2.7 million industrial robots operating in factories around the world in 2020 [IFR, 2020].

8.1 Background

A robotic arm is composed of a mechanical structure with n joints and a terminal tool. The structure enables to set the position $\mathcal{P}$ and orientation $\mathcal{O}$ of the terminal tool. The terminal tool realizes a task and interacts with the environment. The state of a robotic arm is defined by all its joint variables. These variables can be associated to translations or rotations. Rotations are more usual for industrial robots. Only rotations are considered in this study and the joint variables are thus only angles θ_i. The state of a robotic arm is defined by the following elements.

- The position $\mathcal{P}_{\mathcal{T}|F_0} = (x_n, y_n, z_n)$ of the terminal tool in a static reference frame $\mathcal{F}_0$ that is the tasks space.
- The orientation $\mathcal{O}_{\mathcal{T}|F_0}$ of the terminal tool in a static reference frame $\mathcal{F}_0$. The orientation can be expressed in different ways.
  - Euler angles: $\mathcal{O}_{\mathcal{T}|F_0} = (\psi, \theta, \phi)$, the precession, the nutation and the intrinsic rotation. They are chained rotations that can represent any orientation in a 3-dimensional space.
  - Bryant angles: $\mathcal{O}_{\mathcal{T}|F_0} = (yaw, pitch, roll)$, it is a different convention of chained rotations usually used for aircrafts.

— Quaternions: $\mathcal{O}_{\mathcal{T}|F_0} = (w, x, y, z)$, they are a useful mathematical notation for representing spatial orientation in robotics. They enable to avoid the gimbal lock[1] problem but they are not as intuitive as the Euler and Bryant angles.

▷ The joints variables $\mathcal{J}_n = (\theta_0, \ldots, \theta_i, \ldots, \theta_{n-1})$. They are the actuator and sensors of the robot. They enable to control it but also to obtain information about its configuration.

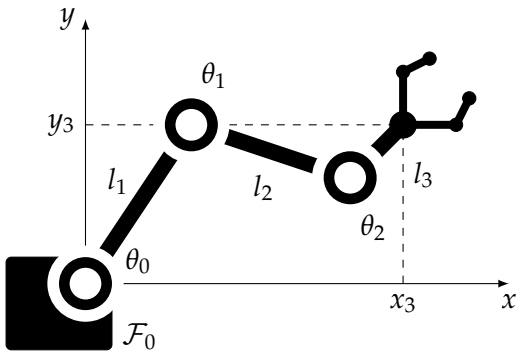


Figure 8.1 – Robotic arm example in a 2D task space with 3 joints. $\mathcal{P}_{\mathcal{T}|F_0} = (x_3, y_3, 0)$, $\mathcal{O}_{\mathcal{T}|F_0} = (\theta_{2|F_0}, 0, 0)$ and $\mathcal{J}_3 = (\theta_0, \theta_1, \theta_2)$.

Figure 8.1 shows an example of robotic arm in a 2-dimensional task space. In this study, only 2-dimensional tasks spaces are considered with n number of joints. The position and orientation become $\mathcal{P}_{\mathcal{T}|F_0} = (x_n, y_n, 0)$ and $\mathcal{O}_{\mathcal{T}|F_0} = (\theta_{n-1|F_0}, 0, 0)$ (with Euler angles).

An industrial robot is controlled in two different ways: with *Direct Kinematics* and *Inverse Kinematics* (Fig. 8.2). The *Direct Kinematic Model* (*DKM*) calculates the position $\mathcal{P}_{\mathcal{T}|F_0}$ and orientation $\mathcal{O}_{\mathcal{T}|F_0}$ of the tool from the joints variables $\mathcal{J}_n$. The *Inverse Kinematic Model* (*IKM*) calculates all possible configurations of joint variables $\mathcal{J}_n, \mathcal{J}_n', \mathcal{J}_n'', \ldots$ in order to reach a desired position $\mathcal{P}_{\mathcal{T}|F_0}$ and orientation $\mathcal{O}_{\mathcal{T}|F_0}$ of the tool.

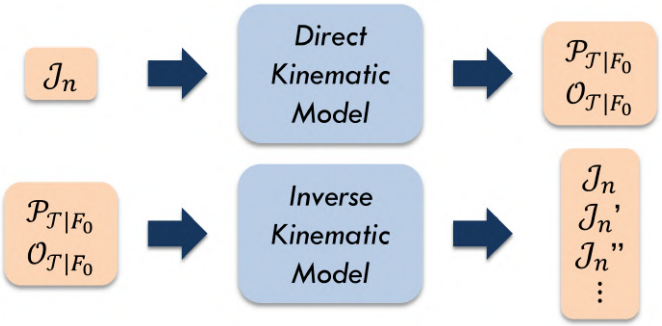


Figure 8.2 – Direct Kinematic and Inverse Kinematic Models for control.

8.1.1 Direct Kinematics

The *Direct Kinematic Model* is the model that calculates $\mathcal{P}_{\mathcal{T}|F_0}$ and $\mathcal{O}_{\mathcal{T}|F_0}$ from $\mathcal{J}_n$. Homogeneous coordinates are used and the model is an homogeneous transfer matrix $\mathcal{T}_{0,n}$

1. The gimbal lock is the loss of one degree of freedom when two rotation axis are parallel.

between the static frame $\mathcal{F}_0$ and the frame of the tool $\mathcal{F}_n$.

$$\mathcal{T}_{0,n} = \begin{pmatrix} \mathcal{R}_{0,n} & \overrightarrow{O_0 O_{n|F_0}} \\ 0_{1\times 3} & 1 \end{pmatrix} \tag{8.1}$$

with $\mathcal{R}_{0,n}$ the rotation matrix between the frames and O_0 and O_n their respective origins. The transfer matrix is usually calculated by decomposing it joint to joint with the a $\mathcal{T}_{i-1,i}$ matrix. With a 2-dimensional task space and only rotation joints the transfer matrix $\mathcal{T}_{i-1,i}$ is given by the equation 8.2.

$$\mathcal{T}_{i-1,i} = \begin{pmatrix} \cos\theta_{i-1} & -\sin\theta_{i-1} & 0 & l_i \cos\theta_{i-1} \\ \sin\theta_{i-1} & \cos\theta_{i-1} & 0 & l_i \sin\theta_{i-1} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{8.2}$$

with l_i the size of the arm segment between the joint θ_{i-1} and θ_i. The final transfer matrix is obtained by doing the product of all $\mathcal{T}_{i-1,i}$.

$$\mathcal{T}_{0,n} = \mathcal{T}_{0,1}.\cdots.\mathcal{T}_{i-1,i}.\cdots.\mathcal{T}_{n-1,n} \tag{8.3}$$

The position of the terminal tool $\mathcal{P}_{\mathcal{T}|F_0}$ is obtained with the vector $\overrightarrow{O_0 O_{n|F_0}}$ To get the rotation, one must convert the rotation matrix $\mathcal{R}_{0,n}$ into Euler angles, Bryant angles or Quaternions.

8.1.2 Inverse Kinematics

Robotic applications usually rely on task space controllers. The model allowing to control a robot in its task space is called the *Inverse Kinematic Model* (*IKM*). It provides all possible configurations of joint variables $\mathcal{J}_n, \mathcal{J}_n', \mathcal{J}_n'', \ldots$ in order to reach a desired position $\mathcal{P}_{\mathcal{T}|F_0}$ and orientation $\mathcal{O}_{\mathcal{T}|F_0}$ of the terminal tool.

The *IKM* can be obtained with analytical approaches for rigid bodied robots of low DOF (Degrees Of Freedom). These approaches perform poorly on complex systems with lot of DOF or soft robots as they are difficult to model [Thuruthel et al., 2016]. Common methods for solving the *IKM* of a redundant manipulator system are numerical solutions: pseudo-inverse methods [Gardner and Velinsky, 2000, Bayle et al., 2003] and Jacobian transpose methods [Hootsmans and Dubowsky, 1991]. They allow better scalability for higher DOF but they still rely on the availability of accurate robot parameters which can be difficult to obtain. Recent motion caption techniques allow to generate pre-learned postures and use data-driven approaches for *Inverse Kinematics* problems [Ho et al., 2013, Holden et al., 2016]. Hybrid methods combine previous techniques and attempt to reduce the complexity of the problem by decomposing it in several components [Unzueta et al., 2008]. Data-driven techniques are the most exploited approaches in the last decades in the domain of computer graphics [Aristidou et al., 2018]. We also found geometric approaches that provide direct solutions using geometrical heuristics [Jamali et al., 2011]. From the point a view of developmental robotics, Baranes and Oudeyer [Baranes and Oudeyer, 2013] tackled the *Inverse*

Kinematics problem by using intrinsically motivated goal exploration while learning the limits of reachability.

Future robots will possess soft joints and high numbers of DOF making them difficult or yet impossible to model. Thus, learning the *IKM* for complex robots is an inevitable way. The implemented architecture for learning the *IKM* presented is this thesis can be considered as a data-driven approach.

8.2 Inverse Kinematics Learning

The learning of the *Inverse Kinematic Model* is detailed in this section. For our experimentations, we repeat one of the experimental setups of Baranes and Oudeyer [Baranes and Oudeyer, 2013] which is the learning of the inverse kinematics with a redundant robotic arm. We consider a robotic arm with segments of length l_i and n joints in a 2-dimensional task space: $(\theta_0, \ldots, \theta_i, \ldots, \theta_{n-1})$ (Fig. 8.1 shows an example for 3 joints). To control it, one must use its *Direct Kinematic Model* (*DKM*) and its *Inverse Kinematic Model* (*IKM*) which are both non linear models dependent on the characteristics of the arm. We propose here to learn the *IKM* using the *DKM* as a supervisor.

The used experimental metrics are presented. Two learning implementations are proposed which involve centralized and decentralized control of joints.

8.2.1 Experimental Metrics

The metrics that are measured during the robotic experiments are the goal position and orientation errors, the amount of generated *endogenous learning situations* and the execution time.

Goal Position Error

To evaluate the predictions of the learning mechanism when a goal position $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$ is requested after a learning phase, a goal prediction error is defined.

$$\mathcal{E}_{\mathcal{P}} = ||\overrightarrow{\mathcal{P}^{goal}_{\mathcal{T}|F_0}\mathcal{P}^{explo}_{\mathcal{T}|F_0}}||/\mathcal{D}_{reachable} \tag{8.4}$$

$\mathcal{P}^{explo}_{\mathcal{T}|F_0}$ is the tool position obtained after the exploitation of the learning. The error is the distance between the desired position and the one obtained by the exploitation (Fig 8.3). This distance is normalized by $\mathcal{D}_{reachable}$. It is the diameter of the reachable area for the robotic arm which is a disc when there aren't any constrains for the joints (Fig 8.4).

Goal Orientation Error

The orientation can also be a requested goal $\mathcal{O}^{goal}_{\mathcal{T}|F_0}$. It is a central element for robotic arms as their task usually requires specif orientations. The orientation error is calculated as it follows.

$$\mathcal{E}_{\mathcal{O}} = |\mathcal{O}^{goal}_{\mathcal{T}|F_0} - \mathcal{O}^{explo}_{\mathcal{T}|F_0}|/2\pi \tag{8.5}$$

The orientation error is the difference between the requested orientation $\mathcal{O}^{goal}_{\mathcal{T}|F_0}$ and the orientation $\mathcal{O}^{explo}_{\mathcal{T}|F_0}$ obtained after the exploitation (Fig 8.3). This difference is normalized by 2π, the maximal possible orientation error in a 2D task space.

Goal Position and Orientation Error

If both the position and the orientation are requested. The goal position and orientation error $\mathcal{E}_{\mathcal{PO}}$ is the mean of the goal position error $\mathcal{E}_{\mathcal{P}}$ and the goal orientation error $\mathcal{E}_{\mathcal{O}}$.

Endogenous Learning Situations

To evaluate the impact of the *Cooperative Neighborhood Learning* mechanism on the goal performances, the number of generated *endogenous learning situations* $\mathcal{L}^{endo}_{\#}$ is observed.

Execution Time

A focus on the execution time of learning episodes is made to appraise how the scalability of the learning mechanism is affected in the presented robotic problems.

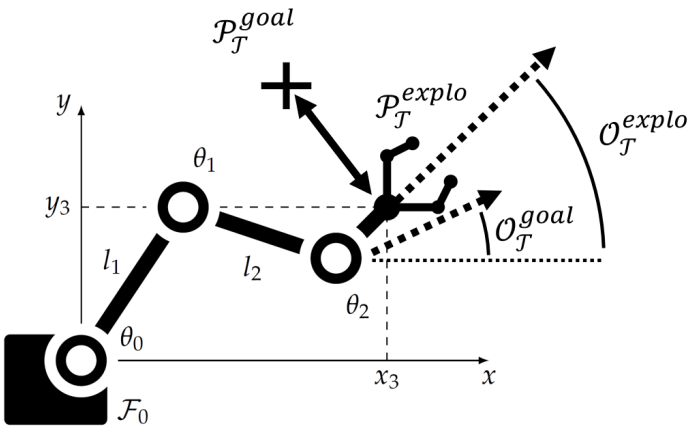


Figure 8.3 – Goal position and orientation error in a 2-dimensional task space.

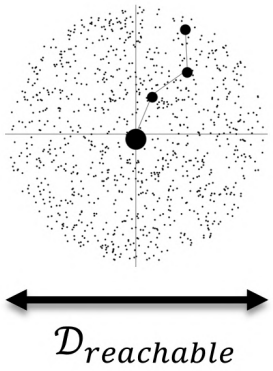


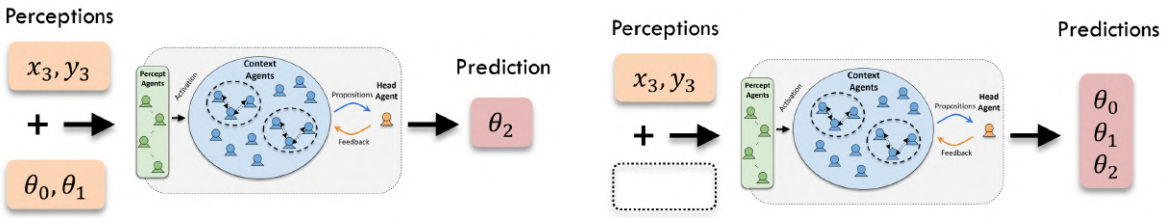
Figure 8.4 – Example of an explored reachable space by a simulated robotic arm in a 2-dimensional task space.

8.2.2 Centralized Control

This section presents the learning of the *IKM* in a centralized setting. The implementation of the experiment is firstly described. Then, the experiment objectives and protocol are presented. The results and their analysis end this section.

Implementation

This experiment involves one learning instance (i.e. one *ELLSA* instance) for learning the *IKM* of robotic arms between 2 and 30 joints. The learning of the *IKM* with one learning instance requires special training, exploration and exploitation that are detailed in the following.



(a) Learning perceptions and prediction (b) Exploitation perceptions and predictions

Figure 8.5 – *IKM* learning implementation with 3 joints in the centralized setting.

Training. The training is made from several randomly generated positions of the tool in the task space $\mathcal{P}^{rdm}_{\mathcal{T}|F_0} = (x^{rdm}_n, y^{rdm}_n)$ and all the corresponding random joint angles except one $(\theta^{rdm}_0, \theta^{rdm}_i, \ldots, \theta^{rdm}_{n-2})$ as *perceptions*. The last angle θ^{rdm}_{n-1} is the *prediction vector* which in our case is a single value. The function that is learned by the mechanism is given by the following equation.

$$\mathcal{F}_{\theta_{n-1}}(x_n, y_n, \theta_0, \theta_i, \ldots, \theta_{n-2}) = \theta_{n-1} \tag{8.6}$$

The *learning situations* are defined as $\mathcal{L}_{n+1,1} = [(x_n, y_n, \theta_0, \theta_i, \ldots, \theta_{n-2}), \theta_{n-1}]$. Figure 8.5a shows an example of learning implementation with 3 joints. For this experiment, only the position of the terminal tool is concerned. Learning the orientation would require an additional perception. Concerning the *learning criticality*, only the distances to the *local models* are considered. All the weight is given to the learning accuracy ($w^{lrn}_{f_n} = 1$, $w^{lrn}_{c_{0,1}} = 0$ and $w^{lrn}_{\mathcal{R}_n} = 0$).

Exploration. The generation of the training set is made using random angles for the joints following a normal distribution. Considering that an outstretched arm is defined by all the angles being set to 0 $[2\pi]$, the distribution of each angle is centered around this value except for θ_0 which has an homogeneous random distribution between 0 and 2π. The dispersion is set empirically for 3, 10 and 30 joints with the objective of having homogeneous situations

in the task space. Figure 8.4 shows an example of exploration with 3 joints after 1000 *learning situations*. The respective empirical dispersions are $\pi/2$, $\pi/4$ and $\pi/6$. The dispersion for n number of joints is given by the formula of the trend curve satisfying these points : $2.5593n^{-0.479}$.

Exploitation. The exploitation of the learning mechanisms is an exploitation with sub-*perceptions* because only one learning instance is used but several actuators are needed (the joint variables). During the exploitation of the learned *IKM*, the objective is to get a set of angles to set the tool of the robotic arm in a desired position $\mathcal{P}^{goal}_{\mathcal{T}|F_0} = (x_n^{goal}, y_n^{goal})$ (Fig. 8.3). All $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$ are randomly generated in the reachable zone of the task space. The exploitation that is performed with the learning instance of *ELLSA* is an exploitation with sub-*perceptions* (Section 6.1.3). The sub-*perceptions* are x_n^{goal} and y_n^{goal}. The *exploitation situations* are defined as $\mathcal{E}_{n+1} = (x_n, y_n)$. Figure 8.5b shows an example of exploitation with 3 joints. The exploitation with sub-*perceptions* usually generates several *Valid Context Agents*. The selection for the *Best Context Agent* is made with the *exploitation criticality* by setting all the weight in the proximity ($w^{expl}_{\mathcal{P}_n} = 1$) and no weight for experience and generalization ($w^{expl}_{c_{0,1}} = 0$ and $w^{expl}_{\mathcal{R}_n} = 0$).

Objectives

The objective of this experiment is to show that the *endogenous learning situations* generated by the *Cooperative Neighborhood Learning* mechanism enable to enhance the learning performances on a concrete problem. Another goal is to test the scalability of this approach to assess its viability on a real world application.

Protocol

In this context, the *Self-Learning Strategy* is used. *Active learning situations* are not available. The *Incompetence NCS* and *Model Ambiguity NCS* are not used. The focus is made on *Cooperative Neighborhood Learning*.

Each learning experience is stopped after 1000 training cycles and the error metrics are calculated over 200 exploitation cycles. The metrics are averaged over 15 learning episodes. The inverse models to be learned are non linear models of several dimensions: 3, 4, 6, 10, 20 and 30. 30 is the number of degrees of freedom on a usual humanoid robot. Humanoid linkages are not serial but the goal here is to test the scalability with the same degrees of freedom order of magnitude. The stretched length for each arm sizes is the same (50 units in our simulation). Thus, for each arm size scenario, the size of the reachable space is the same. The error margin is set to 1. For visualization needs, all angles are multiplied by 100. In this experiment, the generalization and experience weights for the *learning criticality* and the *exploitation criticality* are set to 0. Table 8.1 reminds the parameters used for this scenario. All other parameters are the ones presented in the table 7.1.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of joints*	n	> 1	$\mathbb{N}$	$2, 3, 5, 10, 20, 30$
	number of ELLSA instances	n_{ELLSA}	> 0	$\mathbb{N}$	1
	number of perceptions	$n_{perceptions}$	> 0	$\mathbb{N}$	$n + 1$
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	1000
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	200
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	$0.01/0.03$
	model error margin	m^{f}_{err}	> 0	$\mathbb{R}$	1
Designer.	*accuracy learning weight*	$w^{lrn}_{f_n}$	≥ 0	$\mathbb{R}$	1
	experience learning weight	$w^{lrn}_{c_{0,1}}$	≥ 0	$\mathbb{R}$	0
	generalization learning weight	$w^{lrn}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$	0
	proximity exploitation weight	$w^{expl}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$	1
	experience exploitation weight	$w^{expl}_{c_{0,1}}$	≥ 0	$\mathbb{R}$	0
	generalization exploitation weight	$w^{expl}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$	0

Table 8.1 – Table of experimental, user and designer parameters for the learning of the *IKM* with centralized control.

Results

This section presents the results obtained with the learning of the *IKM* in the centralized control setting.

Figure 8.6 shows the *validity ranges* of a learned *IKM* for 2 joints. With $\mathcal{P}_3 = (x_2, y_2, \theta_0)$ as *perceptions* and $\mathcal{O}_1 = \theta_1$ the *prediction vector*. The model in the *perceptions* space is a helical hollow cylinder. For a desired position $\mathcal{P}^{goal}_{\mathcal{T}|F_0} = (x_2^{goal}, y_2^{goal})$ they are several possible joints configurations. *Context Learning* enables to visualize the redundancy of this problem.

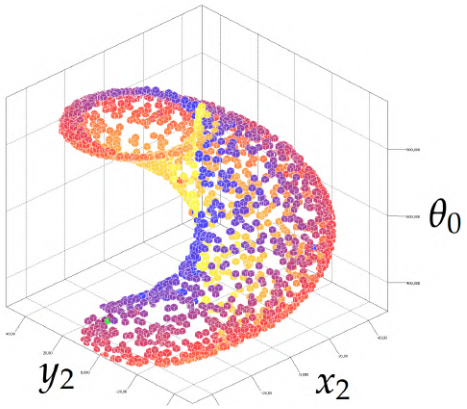


Figure 8.6 – *Validity ranges* for the *IKM* with 2 joints in a 2-dimensional task space. *Perceptions* $\mathcal{P}_3 = (x_2, y_2, \theta_0)$ and *prediction* $\mathcal{O}_1 = \theta_1$.

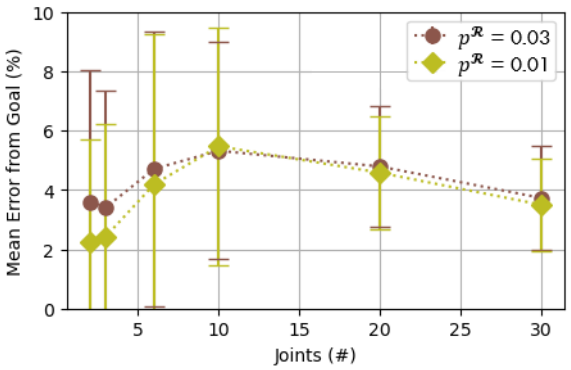


Figure 8.7 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on the number of joints without *Co-operative Neighborhood Learning*.

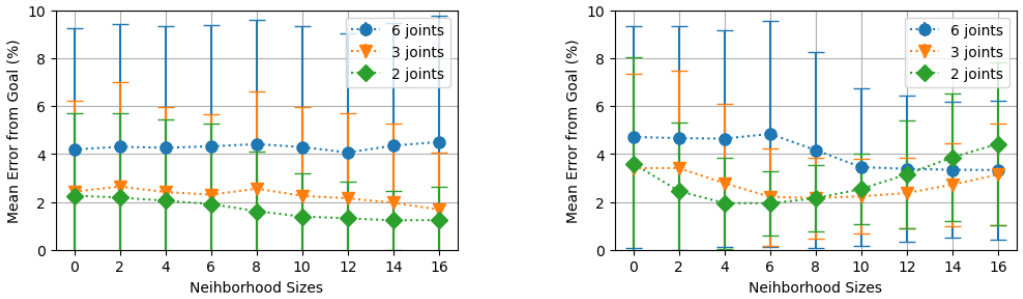
Arm Dimensions

The figure 8.7 shows that without *Cooperative Neighborhood Learning* the best performance is obtained for 2 joints and with a *validity ranges precision* of 0.01. The *validity ranges precision* of 0.01 also gives lower mean error for the arms of 2, 3 and 6 joints than the *validity ranges precision* of 0.03. For arms with more joints the gap is less visible. The mean error and its dispersion increases around 10 joints. Then, it decreases as the number of joints gets higher.

The lowest error is obtained for the lowest arm dimensions. The error increases around 10 joints and it decreases for higher arm dimensions. At low dimensions, the good performance is due to the low redundancy of the problem making the exploration less extensive. The decreasing of the error at high dimensions is caused by the exploitation of the *Context Agents* with sub-*perceptions*. If the requested goal $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$ during the exploitation is in a less explored area, it is the closest model of *Context Agent* that is used for the last angle prediction. The smaller the size of the last arm segment is, the smaller the distance error on the goal is. This is the case for the higher dimensions. The rest of the angles are fixed using the *validity ranges* of the *Best Context Agent*. Moreover, with higher joints the information needed to reach a certain $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$ is distributed across the angles *perceptions* and not only the prediction of the last angle. The better performances are obtained at the cost of *Context Agents* with higher dimensions and longer execution times.

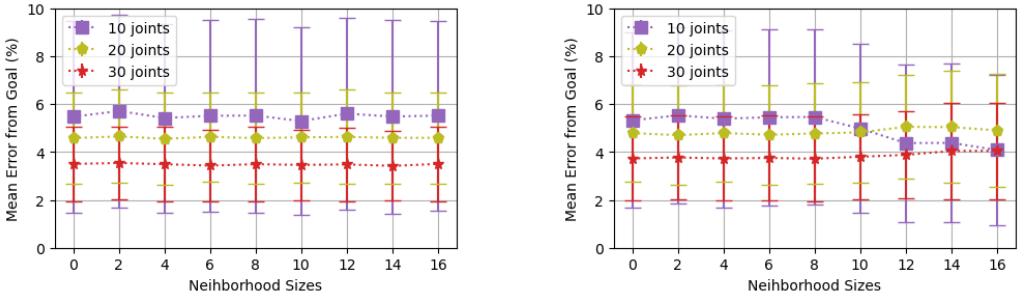
Neighborhood Sizes

Figures 8.8a and 8.8b show that the size of the neighborhood has an impact on the mean error for the arms of 2, 3 and 6 joints. The error slightly decreases for 2 and 3 joints with a *validity ranges precision* of 0.01. With a *validity ranges precision* of 0.03 the error decreases and reaches a minimum value with a delay in the *neighborhood radius coefficient* for the different arms. It then increases for 2 and 3 joints for higher neighborhood sizes. For 10, 20 and 30 joints, the error only decreases for 10 joints with a *validity ranges precision* of 0.03 (Fig. 8.9a). The other cases are not impacted by the variation of neighborhood (Fig. 8.9b).



(a) $p^{\mathcal{R}} = 0.01$ (b) $p^{\mathcal{R}} = 0.03$

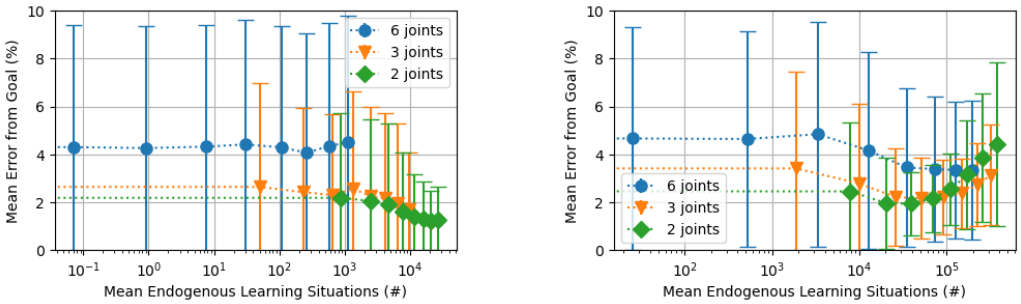
Figure 8.8 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on *neighborhood radius coefficients* over robotic arms of 2, 3 and 6 joints.



(a) $p^{\mathcal{R}} = 0.01$ (b) $p^{\mathcal{R}} = 0.03$

Figure 8.9 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on *neighborhood radius coefficients* over robotic arms of 10, 20 and 30 joints.

Endogenous Learning



(a) $p^{\mathcal{R}} = 0.01$ (b) $p^{\mathcal{R}} = 0.03$

Figure 8.10 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on mean *endogenous learning situations* over robotic arms of 2, 3 and 6 joints.

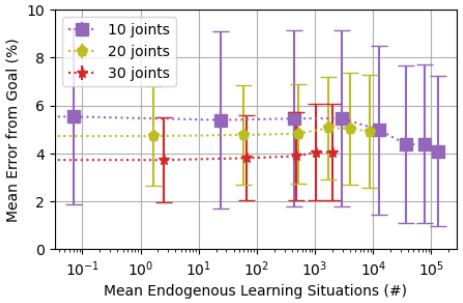


Figure 8.11 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on mean *endogenous learning situations* over robotic arms of 10, 20 and 30 joints; $p^{\mathcal{R}} = 0.03$.

The figures 8.10 and 8.11 represent the same experiment than 8.8 and 8.9 but focusing on the variation of the error according to the generated *endogenous learning situations*. With a *validity ranges precision* of 0.01 (Fig. 8.10a), for 2 and 3 joints, the more *endogenous learning situations* there are, the lower the error is. It is the same for 6 joints and a *validity ranges precision* of 0.03 (Fig. 8.10b). For 2 and 3 joints and a *validity ranges precision* of 0.03, the same behavior than the figure 8.8b is observable.

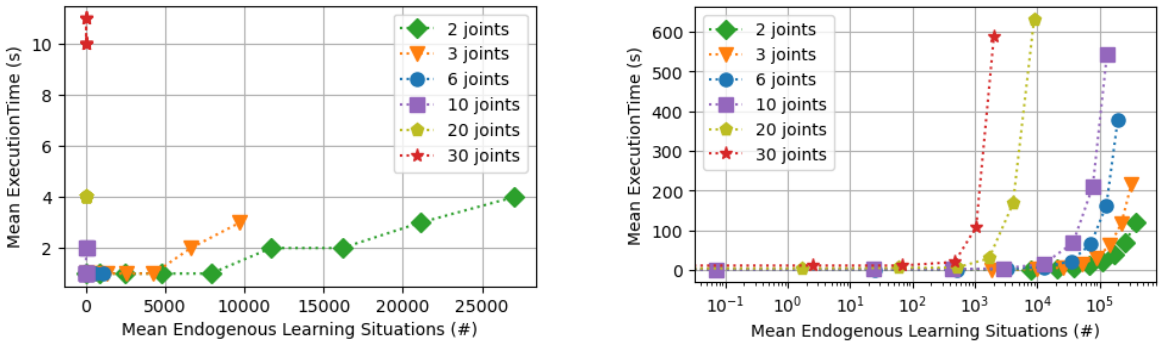
The *validity ranges precision* of 0.01 with the joints 10, 20 and 30 (Fig. 8.9a) is not represented because in this case, almost no *endogenous learning situations* were generated. For 10 joints with a *validity ranges precision* of 0.03 (Fig. 8.11), additional *endogenous learning situations* reduce the error. But for 20 and 30 joints, the *endogenous learning situations* do not reduce the goal error, they even slightly increase it.

Figures 8.8, 8.9, 8.10 and 8.11 showed that the expansion of the neighborhood with larger *validity ranges precisions* can lead to better or worse performances by generating more *endogenous learning situations*. The point of best performance is different for each joint number which shows that the neighborhood behaves differently with higher dimensions. Past this point, the error increases because the generated *endogenous learning situations* are too far from the *Best Context Agents* to bring a coherent smoothing.

At high dimensions, *endogenous learning situations* are harder to generate because of the large exploration space. This is why, for the same *neighborhood radius coefficients*, there are more *endogenous learning situations* at low dimensions. Moreover, beyond 10 joints, the performances are not affected by the *endogenous learning situations*.

Execution Time

Figure 8.12a shows that with a *validity ranges precision* of 0.01, the execution time is slightly affected by the generation of *endogenous learning situations*. However, with a *validity ranges precision* of 0.03 (Fig. 8.12b), small amounts of *endogenous learning situations* lead to the exponential increase of the execution time.



(a) $p^{\mathcal{R}} = 0.01$ (b) $p^{\mathcal{R}} = 0.03$

Figure 8.12 – Mean execution time depending on mean *endogenous learning situations* over robotic arms from 2 to 30 joints.

Extending the neighborhood to generate more *endogenous learning situations* is quickly greedy in computing time even for small amounts of situations. This shows the limitation of our approach especially since the accuracy does not improve significantly with increasing dimensions. A method is therefore needed that guarantees similar results to those of small dimensions for large dimensions.

Synthesis

The magnitude of the mean goal reaching errors of the Self-Adaptive Goal Generation RIAC algorithm (SAGG-RIAC) [Baranes and Oudeyer, 2013] is close to our results. The difference is that SAGG-RIAC uses around 10^4 micro actions for each goal to obtain comparable goal errors without the knowledge of the *Direct Kinematic Model*. Our approach instantaneously gives a set of angles to reach the goal after a training of 1000 *learning situations* but it uses the *DKM* as learning supervisor.

This experiment enabled to applied the *Context Learning* paradigm to the learning of *Inverse Kinematic Models* of robotic arms with different numbers of joints. The results showed that the generation of *endogenous learning situations* led to better performances for the positioning of the terminal tool. The generation of *endogenous learning situations* makes it possible to reduce *exogenous learning situations* as the performance improvement with *Cooperative Neighborhood Learning* attests.

The proposed approach needs to be refined in order to select the right *neighborhood radius coefficient* according to the dimension of the exploring space to maximize performance. The generation of *endogenous learning situations* at high dimensions needs also to be optimized to access to more neighbors with reasonable execution times.

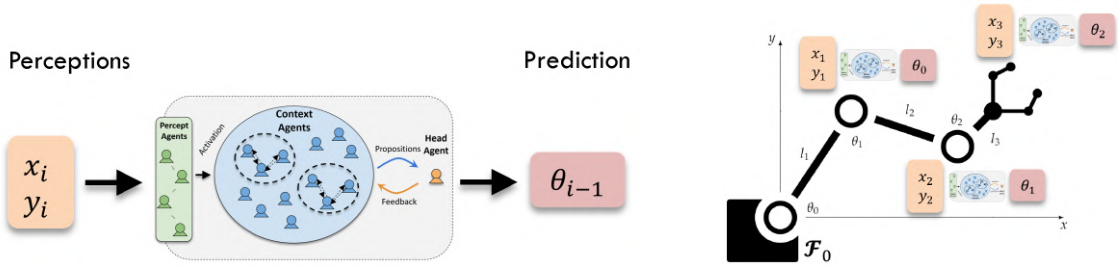
A promising lead is to decompose the learning into several local instances of the learning mechanism, one for each joint. This would reduce the high-dimensional problem into several low-dimensional problems where *Cooperative Neighborhood Learning* is more effective. It will also ensure that the performances are independent of the number of dimensions, and that the execution time is linearly dependent on the dimensions. This will be the focus of the next experiment.

8.2.3 Distributed Control

This section presents the learning of the *IKM* in a distributed setting. The implementation of the experiment is firstly described. Then, the experiment objectives and protocol are presented. The results and their analysis end this section.

Implementation

We propose here to learn the *IKM* still using the *DKM* (*Direct Kinematic Model*) but locally. The learning is exploited with a constraint propagation mechanism. This approach is independent of the joints number of the considered robots. At high dimensions, agent cooperation is computationally expensive and raises implementation issues. To cope with this problem, the learning is separated into several low dimension instances, one for each joint. This experiment involves several *ELLSA* instances for learning the *IKM* of robotic arms between 3 and 100 joints. In the case of distributed control, training and exploitation differ from the previous experiment.



(a) Perceptions and predictions (b) Instances

Figure 8.13 – *IKM* learning and exploitation implementation with 3 joints in the distributed setting.

Training. The training is local to each joint. There are as many learning instances as joints. Let $ELLSA_{\theta_i}$ the learning instance which corresponds to the control of the joint θ_i and the position of the joint θ_{i+1}: $P_{\theta_{i+1}|F_i} = x_{i+1|F_i}, y_{i+1|F_i}$ (Fig. 8.13). The arm configurations are also randomly generated. An instance learns the local angle $\theta^{rdm}_{i|F_i}$ to position its segment l_{i+1} local end $(x^{rdm}_{i+1|F_i}, y^{rdm}_{i+1|F_i})$ in the joint local frame F_i. The training is made with $x^{rdm}_{i+1|F_i}$ and $y^{rdm}_{i+1|F_i}$ as *perceptions* and $\theta^{rdm}_{i|F_i}$ as the oracle value. The local function $\mathcal{F}_{\theta_i}$ learned by each instance is:

$$\mathcal{F}_{\theta_i}(x_{i+1|F_i}, y_{i+1|F_i}) = \theta_{i|F_i} \tag{8.7}$$

The *learning situations* are defined as $\mathcal{L}_2 = [(x_{i+1|F_i}, y_{i+1|F_i}), \theta_{i|F_i}]$.

Exploration. In this case, an homogeneous exploration of space is not necessary as each joint model is learned locally. Still, the same random generation of arm configurations is used for this experiment. This distribution favors extended arm positions. This exploration removes possible biases which would be due to a non homogeneous exploration of the task space.

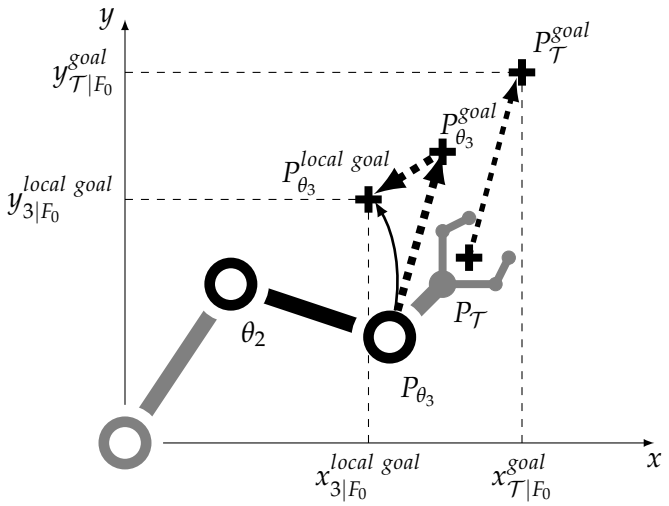


Figure 8.14 – Goal propagation mechanism; F_0 is the frame of the task space.

Exploitation During the exploitation of the learned *IKM*, the goal is to get a set of angles to position the end of the robotic arm in a point $\mathcal{P}^{goal}_{\mathcal{T}|F_0} = (x^{goal}_n, y^{goal}_n)$ (Fig. 8.3). All $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$ are randomly generated in the reachable zone of the task space. Each joint learning instance is exploited sequentially from the first to the last: it defines a goal propagation cycle. $ELLSA_{\theta_i}$ being the learning instance which corresponds to the the control of the joint θ_i and the position of the joint θ_{i+1}: $P_{\theta_{i+1}|F_i} = x_{i+1|F_i}, y_{i+1|F_i}$. Each instance enables to control the joints in local frames. To achieve a matching of the last frame (i.e. the frame of the terminal tool) with the desired goal, the goal position $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$ is propagated on all the joints local frames.

The desired position for the terminal tool $P_{\mathcal{T}|F_0}$ is $\mathcal{P}^{goal}_{\mathcal{T}|F_0}$. To approach this goal, the needed translation between the tool and the goal is $\overrightarrow{P_{\mathcal{T}|F_0}\mathcal{P}^{goal}_{\mathcal{T}|F_0}}$ (Fig 8.14). The translation is propagated to the position of each joint θ_{i+1} which gives $\overrightarrow{P_{\theta_{i+1}|F_0}\mathcal{P}^{goal}_{\theta_{i+1}|F_0}}$.

$\mathcal{P}^{goal}_{\theta_{i+1}|F_0}$ is the position that the joint θ_{i+1} must take to satisfy the goal on the terminal tool. $\mathcal{P}^{goal}_{\theta_{i+1}|F_0}$ is then moved in the reachable zone of θ_{i+1} to generate $\mathcal{P}^{local\ goal}_{\theta_{i+1}|F_0} = (x^{local\ goal}_{i+1|F_0}, y^{local\ goal}_{i+1|F_0})$. $x^{local\ goal}_{i+1|F_0}$ and $y^{local\ goal}_{i+1|F_0}$ are translated to the frame F_i. $x^{local\ goal}_{i+1|F_i}$ and $y^{local\ goal}_{i+1|F_i}$ are the *perceptions* that are given to the learning instance $ELLSA_{\theta_i}$ for the selection of the angle $\theta_{i|F_i}$. The number of goal propagation cycles is chosen by the user. An example of goal propagation on the joint θ_2 is given Fig. 8.14.

In this experiment, orientation goals $\mathcal{O}^{goal}_{\mathcal{T}|F_0}$ are also generated. If the orientation of the end of the arm is a desired constraint, orientation goals are randomly generated between 0 and 2π in addition to the position goals. During the goal propagation, the orientation of the joint θ_n is set to $\mathcal{O}^{goal}_{\mathcal{T}|F_0}$. The rest of the joints then adapt to the fixed orientation during goal propagation.

Objectives

The objectives of this experiment are the same as the previous one but in the distributed control setting. The goal of this experiment is to show that the *endogenous learning situations* generated by the *Cooperative Neighborhood Learning* mechanism enhance the learning performances. Scalability is also assessed with higher numbers of joints than the previous experiment.

Protocol

In this context, the *Self-Learning Strategy* is also used. The *Incompetence NCS* and *Model Ambiguity NCS* are not used. The focus is made on the *Cooperative Neighborhood Learning* mechanism .

The presented results are averaged over 15 learning experiences. A learning cycle corresponds to a configuration for the robotic arm. Each learning instance receives a local *learning situation* at each learning cycle. The inverse models to be learned are non linear models of high dimensions (up to 100). They are all reduced to 2-dimensional problems. We chose to stop at 100 dimensions because it is reasonable to evaluate scalability and it covers most hyper redundant robots and the dimensional complexity of a human body [Aristidou et al., 2018]. The stretched length for each arm sizes is the same as before (50 units in our simulation). Each segment of the arm is decreasingly smaller up to the end of the arm. The error margin is set to 1. For visualization needs, all angles are multiplied by 100. The *validity ranges precision* is fixed at 0.04 for all the presented results. The goal propagation cycles are set to 10. In this experiment, the generalization and experience weights for the *learning criticality* and the *exploitation criticality* are set to 0. Table 8.2 summaries the parameters used for this scenario. All other parameters are the ones presented in the table 7.1.

	Name	Notation	Constrains	Domain	Value
Exp.	*number of joints*	n	> 1	$\mathbb{N}$	$3, 5, 10, 20, 30, 50, 100$
	number of ELLSA instances	n_{ELLSA}	> 0	$\mathbb{N}$	n
	number of perceptions	$n_{perceptions}$	> 0	$\mathbb{N}$	2
	number of learning situations	$\mathcal{L}_{\#}$	> 0	$\mathbb{N}$	200
	number of exploitation situations	$\mathcal{E}_{\#}$	> 0	$\mathbb{N}$	50
	number of learning episodes	$\mathcal{E}ps_{\#}$	> 0	$\mathbb{N}$	15
U.	*validity ranges precision*	$p^{\mathcal{R}}$	$]0, 1[$	$\mathbb{R}$	0.04
	model error margin	m^{I}_{err}	> 0	$\mathbb{R}$	1
	goal propagation cycles	c_{prop}	> 0	$\mathbb{N}$	10
Designer.	*accuracy learning weight*	$w^{lrn}_{I_n}$	≥ 0	$\mathbb{R}$	1
	experience learning weight	$w^{lrn}_{c_{0,1}}$	≥ 0	$\mathbb{R}$	0
	generalization learning weight	$w^{lrn}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$	0
	proximity exploitation weight	$w^{expl}_{\mathcal{P}_n}$	≥ 0	$\mathbb{R}$	1
	experience exploitation weight	$w^{expl}_{c_{0,1}}$	≥ 0	$\mathbb{R}$	0
	generalization exploitation weight	$w^{expl}_{\mathcal{R}_n}$	≥ 0	$\mathbb{R}$	0

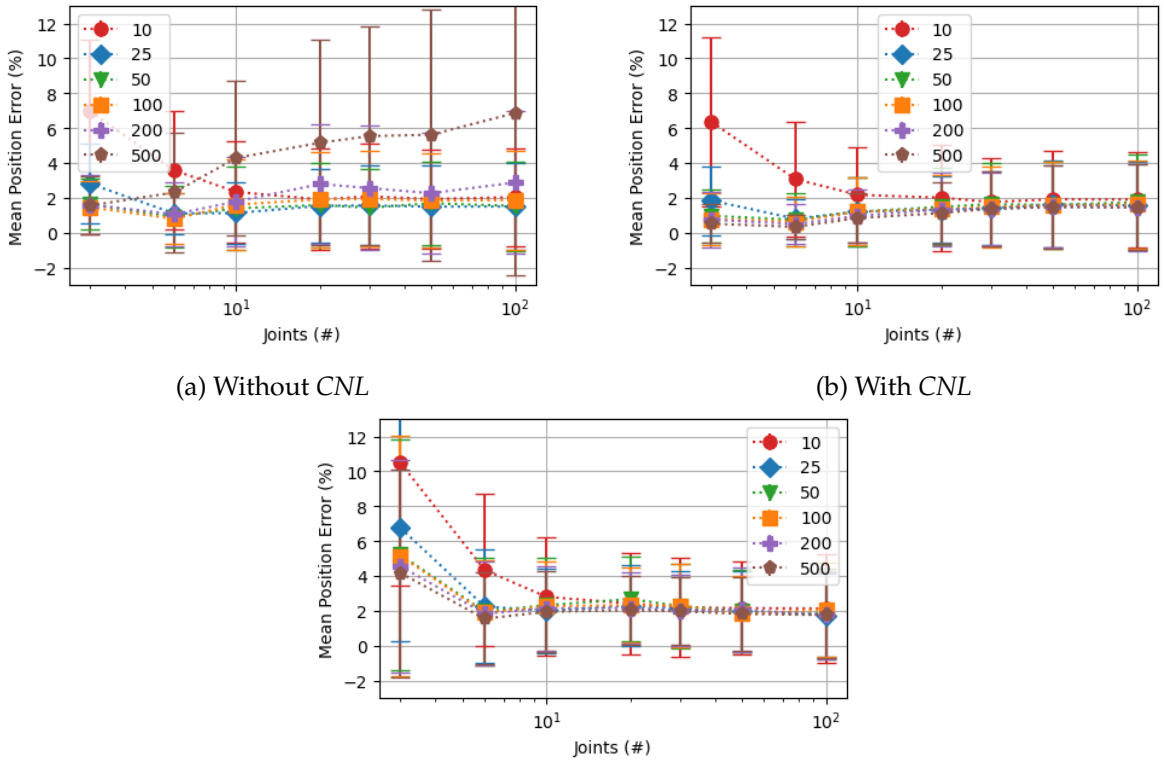
Table 8.2 – Table of experimental, user and designer parameters for the learning of the *IKM* with distributed control.

Results

This section presents the results obtained with the learning of the *IKM* in the distributed control setting.

Learning Situations and Arm Dimensions

Figure 8.15 shows the impact of learning situations on the goal position error depending on the joints number. For low *learning situations* and for small number of joints, the goal position error increases (Fig. 8.15a). For high *learning situations* and for big numbers of joints, the goal position error increases too. The addition of *Cooperative Neighborhood Learning* (Fig. 8.15b) removes the goal position error divergence for high *learning situations* and high number of joints. Starting from 25 *learning situations*, the goal position error is relatively independent of the number of joints. Fig. 8.15c shows the addition of orientation goals. This has the effect of slightly increasing the position error while keeping the same curve tendency.



(a) Without *CNL*

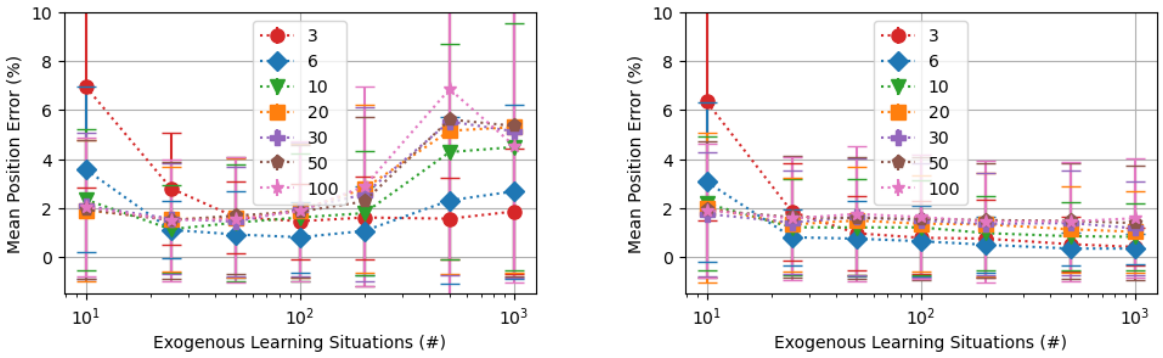
(b) With *CNL*

(c) With *CNL* and orientation goals

Figure 8.15 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on robotic arm sizes for different learning cycles (10, 25, 50, 100, 200 and 500), with and without *Cooperative Neighborhood Learning* (*CNL*).

Figure 8.16 shows the impact of joints numbers on the goal position error depending on *learning situations* with and without *Cooperative Neighborhood Learning*. One can see that

the addition of *Cooperative Neighborhood Learning* clearly decreases the goal position error in particular for high joints numbers. With *Cooperative Neighborhood Learning*, small amounts of *learning situations* is sufficient to obtain goal position errors close to the lowest obtained with high amounts of *learning situations*.



(a) Without *CNL*

(b) With *CNL*

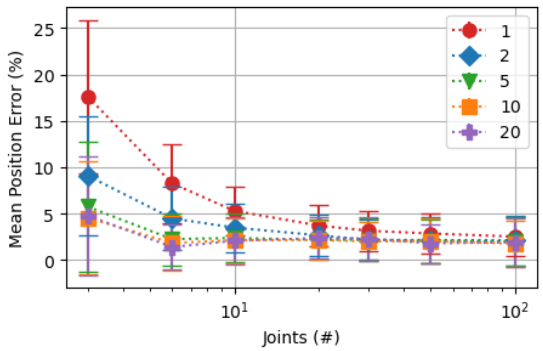
Figure 8.16 – Mean goal position errors $\mathcal{E}_{\mathcal{P}}$ depending on *learning situations* for different robotic arm sizes (3, 6, 10, 20, 30, 50 and 100), with and without *Cooperative Neighborhood Learning* (*CNL*).

Without *Cooperative Neighborhood Learning*, a worse learning is observed for high joints numbers and high *learning situations* that may be due to a bias in the used learning models. This bias is compensated by the generation of *endogenous learning situations*. Figures 8.15b and 8.16b show that with the addition of *Cooperative Neighborhood Learning*, the exploitation of the learned inverse model for position control is mostly independent of the joints numbers of the robotic arms. If a joint local learned model is not accurate enough, the goal propagation allows to transfer its incompetence to the other joints models. The collective of learning instances then allows the emergence of a correct tool position for the arm regardless of the number of joints. The only exception is for the smallest number of joints configurations and the lowest learning cycles. In this case, because of the poor redundancy, the *endogenous learning situations* cannot further decrease the error. The addition of the orientation for the goal leads to an error increase for low joints number because high redundancy is essential to control the orientation of the arm extremity. With few joints, it is harder to satisfy the position and orientation goals.

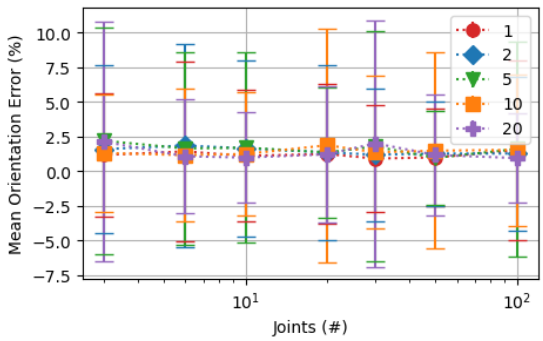
As Figure 8.16b shows it, using more than 25 *learning situations* by joint does not significantly improve the goal position error. This is also due to the redundancy of the arms coupled with the goal propagation. Small quantities of arm configurations are enough to learn the *Inverse Kinematic Model* with position errors between 1%($\pm$1) and 2%($\pm$2).

Goal Propagation

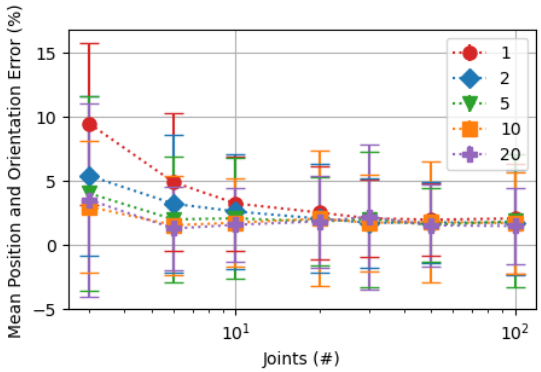
Figure 8.17 shows the impact of goal propagation cycles on the goal position error depending on the joints numbers. On figure 8.17a, the different goal propagations give similar results except for the single and twice propagations, for which, one observes worse position errors. Yet, for high joints number, the error converges towards the same value than the other propagations. The orientation error (Fig. 8.17b) is independent of the number of joints or propagation. The position and orientation errors (Fig. 8.17c) follow the same behavior than the position error but with lower error values for small joints number.



(a) Goal Position Error $\mathcal{E}_{\mathcal{P}}$



(b) Goal Orientation Error $\mathcal{E}_{\mathcal{O}}$



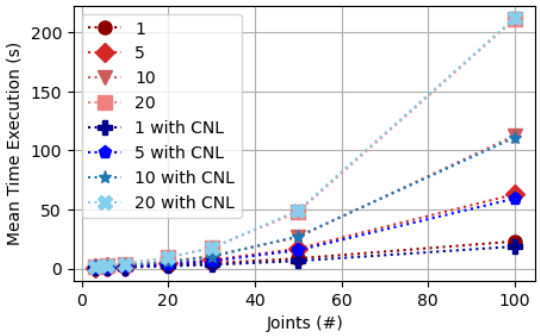
(c) Goal Position and Orientation Error $\mathcal{E}_{\mathcal{PO}}$

Figure 8.17 – Mean goal position and orientation errors depending on robotic arm sizes for different goal propagation cycles (1, 2, 5, 10 and 20), with *Cooperative Neighborhood Learning*, with orientation goal.

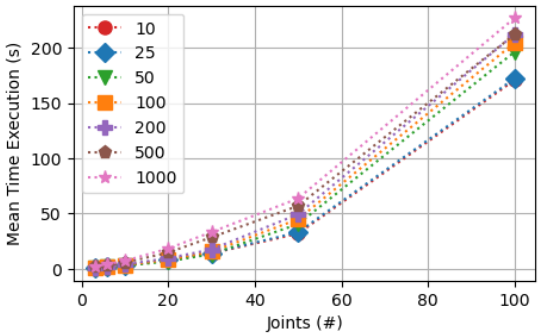
Except for the single propagation, using more than 5 propagations does not significantly improve the position and orientation errors. When goal orientation is implemented, the choice of setting the angle of the last joint at the end of propagation explains the poor results of the single propagation. Joints models do not get a second chance to adapt to the fixed orientation of the arm extremity. Indeed, the second wave of propagation significantly reduces the error. As expected, the orientation is independent of the joints number. The chaotic deviations are the consequence of goal orientations that are not reachable (mainly on the edges of the task space).

Execution Time

Figure 8.18 shows the impact of goal propagation cycles and *learning situations* on the execution time of the experiments depending on the joints number. Figure 8.18a shows that the addition of *Cooperative Neighborhood Learning* does not increase time execution. The number of goal propagation cycles has an impact. The more propagation cycles there are, the greater the execution time is. For 20 goal propagations, the trend curve is polynomial of degree 2 with a determination coefficient of 0.9997. The coefficient of degree 2 is 0.022 and the coefficient of degree 1 is -0.104. For 5 goal propagations, the trend curve is also polynomial of degree 2 with a determination coefficient of 0.9999. The coefficient of degree 2 is 0.006 and the coefficient of degree 1 is 0.012. Figure 8.18b shows that for more learning cycles, the execution time of experiences is slightly increased. Lastly, as an example, using the same experimental parameters than figure 8.18a for 10 joints and 5 goal propagation cycles, the exploitation time represents 40% of the total time execution of the experiment.



(a) For different goal propagation cycles (1, 5, 10 and 20); with and without *Cooperative Neighborhood Learning* (*CNL*).



(b) For different learning cycles (10, 25, 50, 100, 200, 500 and 1000); with *Cooperative Neighborhood Learning*; $c_{prop} = 20$.

Figure 8.18 – Mean experience time execution depending on robotic arm sizes; with orientation goal.

The execution time of the implemented approach for learning *Inverse Kinematics Models* is polynomially dependent on the number of joints. However, the highest coefficients for the trend curves are of degree 1 which shows that the linear dependency is almost reached. Moreover, 5 goal propagations is enough to obtain reasonable performances regarding the best reached errors for higher goal propagations. As can be seen on the figure 8.18b, the impact of learning cycles is weak and adding more learning situations does not change the polynomial dependence on the number of joints.

Synthesis

With close experimental conditions, for 30 joints, the Self-Adaptive Goal Generation RIAC algorithm (SAGG-RIAC) [Baranes and Oudeyer, 2013] reaches a position error of around 2.5%($\pm$2) (with the position error metric of section 8.2.1) against less than 2%($\pm$2) for our results (Figures 8.15b and 8.16b). The difference is that SAGG-RIAC uses around 10^4 micro actions for each goal to reach comparable performances. Our approach needs a few dozen learning examples and less than 10 goal propagations across the arm joints. One difference though is that the SAGG-RIAC does no relies on the *Direct Kinematic Model*.

This experiment has showed that the generation of *endogenous learning situations* makes it possible to reduce *learning situations* as the performance improvement with *Cooperative Neighborhood Learning* attests.

The several learning instances coupled with goal propagation and *Cooperative Neighborhood Learning* provided scalable performances that are independent of the number articulations of the robotic arms.

The learning architecture is not highlighted at its best with the characteristics of the use joints because they could be locally modeled by analytical techniques.

Further focus could be the implementation of this approach with more complex 3D joints or soft joints. Soft robots are known for being difficult, if not impossible to characterize. Since the implemented learning is dependent on the *Direct Kinematic Model*, future concerns could focus on how to remove this dependency. This approach also gives promising leads for implementing multi-goal situations to simulate robotic hands where each finger has its own goal.

8.3 Synthesis

This chapter proposed an application of *Endogenous Context Learning* in a robotic context. It was the learning of the *Inverse Kinematic Model* of a robotic arm. Centralized and distributed control implementations were proposed. Both configurations allowed to experience how the generation of *endogenous learning situations* enhances the prediction performances. The centralized control faced scalability issues that where solved by the distributed control. The presented implementation uses the *Direct Kinematic Model* as a supervisor during learning. One perspective would be to remove this dependency on the *Direct Kinematic Model* because robot parameters can be hard to obtain, especially with soft robots. A solution to this problem would be to learn how to perform position and orientation variations.

To finish the synthesis on the robotic experiments, the related objectives of section 1.6 are reminded and a discussion is provided with respect to them.

- ***Endogenous Feedback***. One of the focus of the robotic experiment was the impact of the *endogenous learning situations* generation. *Endogenous learning situations* enabled to enhance the prediction performances on both the centralized and distributed control. As seen in the previous chapter, higher numbers of *perceptions* require to enlarge the *neighborhood* size to consider a larger zone surrounding the *perceptions* in order to detect more *Neighbor Context Agents*. It worked up to 6 joints (7 *perceptions* for the centralized experient). Beyond 6 joints, the generated *endogenous learning situations* did not enhance the prediction performances. The *neighborhood* behavior for high dimensions needs to be further investigated so that the *Cooperative Neighborhood Learning* mechanism works whatever the number of dimensions. One alternative was to distribute the learning with several learning instances of low dimensions. In this case, the generation of *endogenous learning situations* provided better prediction performances regardless of the number of joints.
- ***Scalability***. Concerning this objective, the centralized experiment confirmed the lack of *endogenous learning situations* for high dimensions. The expansion of the *neighborhood* enabled to generate *endogenous learning situations* and to reach better prediction performances. However the resulting execution time increased exponentially with the additional situations. On the other hand, the distributed experiment showed polynomial (close to linear) time execution dependency on the number of joints.
- ***Any Data Amount***. Concerning the number of required *learning situations*, the centralized and the distributed experiments differ. The centralized experiment needs a rather large quantity of *learning situations* dispatched homogeneously in the reachable space of the robot. The distributed scenario can achieve similar position errors with very few *learning situations*. This is achieved through the propagation mechanism that allows knowledge gaps to be propagated to other joints.

Lifelong Learning by Endogenous Feedback
Application to a Robotic System

Conclusions and Perspectives

9 Conclusions and Perspectives

In this final chapter, the conclusions and perspectives are provided in relation to the initial objectives of this thesis (section 1.6). The contributions are declined in 3 categories: Context Learning, Machine Learning and Robotics. The perspectives present exploration tracks concerning additional adaptation and optimization for Context Learning, leads to advance certain challenges of Machine Learning and AI, and possible extensions of the work carried out on robotics.

Conclusions

This thesis focused on the design of *ELLSA* (*Endogenous Lifelong Learner by Self-Adaptation*. An analysis of each chapter of the manuscript is given. Then, a discussion with respect to each objective of this work is conducted.

Manuscript Analysis

Chapter 1 established the context of this work by presenting the focused environments by the designed learning mechanism. The future of robotics, industry and *AI* leads to rethinking the design of learning systems. This reflection led to consider the environment as a complex system and allowed to characterize the objectives of the work done in this thesis: *Endogenous Feedback*, *Agnosticity*, *Lifelong Learning*, *Online Learning*, *Self-Observation*, *Knowledge Generalization*, *Scalability*, *Any Data Amount* and *Explainability*.

Chapters 2, 3 and 4 described the state of the art of the learning approaches and challenges in relation with the defined objectives. Chapter 2 described the lack of *Endogenous Feedback* and *Self-Observation* in the classical learning approaches and the growing concerns of *Online Learning*, *Lifelong Learning* and *Knowledge Generalization*. Chapter 3 focused on the challenges of *Lifelong Learning* and its related learning approaches. Among the presented paradigms, *Endogenous Feedback* and *Self-Observation* appeared to be poorly addressed. Chapter 4 concentrated on learning paradigms that were using endogenous mechanism especially in robotics. The reflection was finished by the introduction of *Context Learning*, a distributed and agnostic learning approach based on *Adaptive Multi-Agent Systems*. This technique provides great malleability in the learning process to ease the implementation of the missing required properties.

Chapters 5 and 6 detailed the contributions of this thesis. The first contribution is the def-

inition of *Endogenous Learning* and its implementation in the paradigm of *Context Learning*: *Endogenous Context Learning*. The second contribution is the design of *ELLSA* (*Endogenous Lifelong Learner by Self-Adaptation*), a learning mechanism that integrates *endogenous*, *agnostic*, *lifelong*, *online*, *self-observation* and *generalization* learning properties.

Chapter 5 introduced *Endogenous Learning* by giving its motivations and definitions. In this chapter, the focus was on the *Context Agents* which represent the local learning fragments. The *neighborhood* and *influence* mechanisms were established in order to enhance communication between the knowledge fragments. The shortcomings of *Context Learning* motivated the characterization of *learning inaccuracies*. Their detection and resolution were made possible with the addition of communication between *Neighbor Context Agents*. To finish this chapter, *Cooperative Neighborhood Learning* was introduced. It is meant for the generation of *endogenous learning situations* in order to enhance the learning performances and save *exogenous learning situations*.

Chapter 6 presented *Endogenous Context Learning* from a broader point of view by describing all the mechanisms that intervene in the learning system *ELLSA*. Learning and exploitation cycles were detailed as both intervene during learning to generate *exogenous learning situations* and *endogenous learning situations*. Generalization, experience and performance were formalized with the use of a *learning criticality* and an *exploitation criticality*. A mechanism was added to update the *local models* in a lifelong setting with the goal to give weight to past *learning situations* without storing them. This mechanism was essential for *Cooperative Neighborhood Learning*. It enables *endogenous learning situations* to enhance the *local models* without completely altering their past experiences. I have proposed two learning strategies, the *Active Learning Strategy* and the *Self-Learning Strategy*. The *Active Learning Strategy* is intended for learning scenarios were specific additional *active learning situations* (also *exogenous learning situations*) can be requested to enhance the learning process. The *Self-Learning Strategy* only uses endogenous mechanisms to enhance the learning process and it uses fewer *exogenous learning situations*. These endogenous mechanisms are the generation of *endogenous exploitation situations* and *endogenous learning situations*. The chapters of contribution were finished with discussions and leads concerning related learning paradigms and their challenges. *Multi-Task Learning* is the challenge that would need the most additions to *Endogenous Context Learning* to be performed as it would involve several collectives of *Context Agents*. *Reinforcement Learning* could be achieved by using several instances of *ELLSA* and with continuous state and action spaces when most *Reinforcement Learning* techniques only use discrete state and action space. *Transfer Learning* seemed to be achievable in the *Context Learning* and is presented in the experimentations.

Chapters 7 and 8 detailed the conducted experiments. Chapter 7 focused on learning scenarios that were independent of any application domain. It enabled to assess the learning mechanism on different hidden models properties (continuity, discontinuity, linearity and non linearity) but also varying models with making as little assumptions as possible and without reprogramming *ELLSA* or resetting the learning parameters. Each of the learning objectives were evaluated quantitatively and/or qualitatively. Chapter 8 presented an application of *Endogenous Context Learning* in a concrete robotic problem: the learning of the *Inverse Kinematic Model* of a robotic arm. The experiments included a centralized and a distributed learning scenario with the *Self-Learning Strategy* where the generation of *endogenous*

learning situations enabled performance enhancements.

Objectives Conclusions

This part of the conclusions reminds the objectives defined section 1.6 and highlights the contributions for each of them.

Endogenous Feedback

The first objective of this thesis is to generate *Endogenous Feedback* to enhance a learning process. This goal was achieved with the generation of *active learning situations*, *endogenous exploitation situations* and *endogenous learning situations*. The possible learning enhancements were characterized into *learning inaccuracies*. These *learning inaccuracies* were detected and resolved using *neighborhood* mechanisms, *active learning situations* (with the *Active Learning Strategy*) and *endogenous exploitation situations* (with the *Self-Learning Strategy*). Concerning the *Self-Learning Strategy*, a second learning enhancement was to learn with fewer *exogenous learning situations* and compensate the lack of *learning situations* with the generation of *endogenous learning situations*. The experiments on mathematical models enabled to validate the generation of *active learning situations* and *endogenous exploitation situations* on several hidden models. The enhancement of learning performances was observed with the use of both strategies. The robotic experimentations also proved that the *endogenous learning situations* improved the learning performances on a concrete problem. One issue was the generation of *endogenous learning situations* with high numbers of *perceptions* where the *Context Agents* could not perceive any *Neighbor Context Agents*. Further work could focus on the behavior of *neighborhood* mechanisms for high dimension.

Agnosticity

The *Agnosticity* objective is to design a learning mechanism that makes no assumptions on the tasks it will have to learn. Instead, it focuses on the learning task. This goal was pursued all along this thesis. *Endogenous Context Learning* was design starting from the most general learning hypotheses possible. These hypotheses were then declined in the *Context Learning* paradigm to define the *Endogenous Learning* principles. The experiments conducted chapter 7 were intentionally independent of any application. They showed that all problems did not need specific tuning of the parameters to achieve similar performances. The experiments chapter 8 confirmed that a complete change of the learning parameters was not required to perform a concrete robotic learning scenario. Additional work would be to carry out several other concrete learning scenario in other domains than robotics to further evaluate the *Agnosticity* of *Endogenous Context Learning*.

Lifelong Learning

The objective of *Lifelong Learning* is to provide the learning mechanism with continuous learning properties. An incremental way of learning was achieved with the update mech-

anism that enables *learning situations* to be accepted in a lifelong way. The utility of an *exogenous learning situation* is represented by the *model prediction distance* and the *model error margin*. They determine when a new *exogenous learning situation* should be learned. *Cooperative Neighborhood Learning* brought incremental learning as *endogenous learning situations* enable to progressively enhance the *local models* by sharing their knowledge. This transfer of information was also achieved during the creation of new models to speed up the learning. *Lifelong Learning* in the sense of *Transfer Learning* focuses on the transfer of entire tasks. An experiment concerning such problem was conducted. It showed that *ELLSA* possesses great adaptive properties because it can update its collective of *Context Agents* when the learning task changes. During this update, it manages to transfer some experiences from the previously learned tasks. Especially during the creation of new knowledge fragments. Future investigations could focus on more complex learning tasks to obtain additional assessments on the transfer properties of *Endogenous Context Learning*.

Online Learning

The objective of *Online Learning* is that the designed learning mechanisms has to learn in a sequential way and new experiences should not affect the whole collective of models. This functioning was already present in classical *Context Learning* as each new *exogenous learning situation* is provided sequentially. The local properties of *AMAS* then enabled to locally modify the concerned *local models* without altering all the *Context Agents*. The addition of *neighborhoods* permits to extend and control a larger working area where *local models* can learn by sharing *endogenous learning situations*. The capacity of alternating learning and exploitation phase was validated during the *Transfer Learning* experiment. Future work could focus on defining an *Online Learning* metric to better measure this property on more adapted learning scenarios.

Self-Observation

The goal of *Self-Observation* is to make the learning system capable of self-adapting and provide feedback on its knowledge. The implementation of *learning inaccuracies* detection and resolution shows that part of this objective is reached. *ELLSA* can identify shortcomings and resolve them with or without the help of *exogenous learning situations*. One difficulty that still remains is the enhancement of discontinuities with the *Self-Learning Strategy*. The work on this problem should be continued. Tools for enabling the learning mechanism to provide feedback on its knowledge were introduced with the *normalized confidence*. It brings information on relative confidence between the *local models* in the collective and it can show if there are huge confidence differences among the *Context Agents*. One additional *learning inaccuracy* could be to add learning focus on less experimented zones where *local models* have received fewer *learning situations* and are thus less confident.

Knowledge Generalization

The objective of *Knowledge Generalization* is to increase the generalizing properties of *Context Learning*. It includes adding experience transfer over time to enhance the learning performances. The model similarity mechanisms focus on the increase of generalization by enabling *Context Agents* to merge. Even if an increase of generalization was observed with some learning scenario, for higher numbers of *perceptions*, this technique is not efficient enough. It depends on the alignment of the *Context Agents validity ranges* in the space of *perceptions* which is not controlled. Additional work concerning this objective needs to be done. To achieve better generalization, *local models* should be considered using the *validity ranges* differently. The *neighborhood* mechanism could also help reducing the number of agents by merging them when they are all *Prediction Neighbor Context Agents* for example. The experience transfer was added with the creation of new *Context Agents* using sponsors. This addition has proven itself useful during the transfer learning experiment. However, there is still room for improvement because all learning metric were not enhanced thanks to past experiences. One *learning inaccuracy* that is explored in this thesis was the incompetence but not the novelty. Seeking novelties in already learned *local models* could improve transfer properties by focusing on evolving areas of the search space.

Scalability

This objective is to manage learning with high numbers of *perceptions*. To do so, the *Percept Agents* optimized the activation process of the *Context Agents*. One of the goals of the *neighborhood* was also to consider small sets of *Neighbor Context Agents* for the resolutions of *NCS* instead of considering the entire collective of *Context Agents* at each execution cycle. The experiment assessing the execution time showed that depending on the used learning strategies, polynomial, linear and logarithmic dependencies on the number of *Context Agents* and *perceptions* were found. Assuming that it is possible, additional work needs to be done to achieve linear dependency on the worst case. The most computationally expensive processing is the active exploration of incompetent areas. It should be redesigned or combined with passive exploration. The experiments showed that with higher dimensions the *neighborhood* behaves differently and *endogenous learning situations* are no longer generated because of the non-availability of *Neighbor Context Agents*. Increasing the size of the *neighborhoods* partially resolved this problem but it generated exponentially increasing execution times. The *Cooperative Neighborhood Learning* mechanism requires additional experimentations with high dimensions. It would enable to understand how the *neighborhood* and the *influence* mechanisms could be enhanced so that they could perform well regardless of the number of *perceptions*. A solution to the *Scalability* issue with high dimensions was to distribute the learning into low *perceptions* learning instance. The proposed distributed approach for solving the *Inverse Kinematic Model* of a robotic arm showed reasonable execution times variations.

Any Data Amount

Another objective of this work is to perform learning with many *learning situations* as well as few *learning situations*. The *Lifelong Context Learning* updating mechanism enables to

keep enhancing the *local models* with growing *exogenous* and *endogenous learning situations*. The *Cooperative Neighborhood Learning* mechanism enabled to reduce the number of *exogenous learning situations* by compensating with the generation of *endogenous learning situations*. This objective was achieved for low numbers of *perceptions*. But, as previously said, for higher dimensions, *Cooperative Neighborhood Learning* needs to be refined as *endogenous learning situations* are more rare if not impossible to generate. On the other hand, the distributed robotic control scenario managed to perform learning with very few and many *learning situations* using constraint propagation between several low dimensional learning instances.

Explainability

The objective of *Explainability* is to keep full transparency in the implemented learning mechanisms so that they can be measured, observed and understood. *Context Learning* provides all this characteristics as the learning models are encapsulated into agents that represent a well defined portion of the space. The implemented user interface allowed to visualize in real time the construction of the learning fragments (i.e. the *validity ranges* and the associated *local models*). For better *Explainability*, the experiments were conducted with low dimensions that were fully visualizable. The interface enabled to visualize the redundancy of an *Inverse Kinematic Model* and validate the presence of continuities and discontinuities.

Robotic Application

A final objective of this thesis is to apply the learning mechanism to a robotic application. The chosen case study was the learning of the *Inverse Kinematic Problem* of robotic arms. It is a difficult redundant and non linear problem with increasing complexity with the number of joints. Centralized and distributed control approaches were proposed to solve this problem. Both respectively succeeded up to 30 and 100 joints with comparable performances to a developmental approach. The goal of this experiment is not to perform better than state of the art approaches but to show that *Endogenous Context Learning* enables to enhance the learning performances on a concrete and useful application. The experiments were conducted on 2-dimensional arms with the *Direct Kinematic Model* as a supervisor. Future work should focus on the extension of this technique on 3-dimensional robotic arm. Another improvement would be to remove the dependence on the *Direct Kinematic Model*. It is indeed way easier to calculate analytically but it still relies on the robot parameters which can be hard if not impossible to obtain with complex and/or soft robots.

Contributions

The contributions of this work are separated into 3 categories *Context Learning*, Machine Learning and Robotics.

Contribution to Context Learning using AMAS

The research conducted in this thesis followed a series of works on learning mechanisms involving *Context Learning*: Obsidian [Videau, 2011], AMADEUS [Guivarch et al., 2013], ESCHER [Boes, 2014], ALEX [Verstaevel, 2016] and AMOEBA [Nigon et al., 2017]. *ELLSA* introduced new concepts: *neighborhood*, *influences*, *Active Learning* and *Endogenous Learning* in the setting of *Context Learning*. The implementation of the *neighborhood* and the *influences* enabled to enhance cooperation between the *Context Agents* in order to detect and solve *Non Cooperative Situations* (*NCS*). This enhanced communication unlocked the ability for the learning mechanism to reconsider its learning representations which are the *Context Agents*. This self-reflection permitted to internally detect weaknesses in the learning models. To resolve the deficiencies, the passive learning strategy of *Context Learning* was transformed into an *Active Learning Strategy* or a *Self-Learning Strategy*. Finally, in addition to simple model performance, another advancement of this thesis is the inclusion of generalization and experience in the learning process of *Context Learning*.

Contribution to Machine Learning

All along the formalization of *Endogenous Context Learning*, a special focus was made on two important properties: the genericity of the approach and the independence of the embedded learning models. Except for model affinity and similarity that are model-dependent measures, all the mechanisms ignore the underlying models of *Context Agents*. This offers interesting learning properties as other learning techniques could be used and distributed when there are usually not. With normalized affinity and similarity measures, different learning models could be used in the same collective of *Context Agents*. *Endogenous Context Learning* offers a distributed *Meta-Learning* framework with properties of *Endogenous Feedback*, *Agnosticity*, *Lifelong Learning*, *Online Learning*, *Self-Observation*, *Knowledge Generalization*, *Scalability*, *Any Data Amount* and *Explainability*.

Contribution to Robotics

Robotic contributions are of the order of *Context Learning* architectures to perform centralized and distributed *Inverse Kinematic* control. The centralized and distributed control proposed original resolutions of the *Inverse Kinematic* problem for robotic arms using *Context Learning*. The distributed approach introduced a constraint propagation mechanism that enables the resolution of the *IKM* to be independent from the number of degrees of freedom and that reduces the required number of *learning situations*.

Perspectives

This works offers interesting perspectives that can be presented in different topics.

Cooperation and Adaptation

The introduction of *neighborhood* among the *Context Agents* enabled to add new cooperative behavior with small sets of *Context Agents*. Additional treatments could be effectuated with the *neighborhood* like supplementary *learning inaccuracies*. Like previously said, the new *learning inaccuracies* could involve novelty seeking or less confident *Context Agents*.

The *neighborhood* itself could be more adaptive by changing its size depending on the considered zone. This would enable to enhance the learning precision when it is needed. At the moment, fixed parameters are responsible for the precision and size of the *Context Agents*. The ability of the *neighborhood* to self-adapt could also provide enhanced visibility to tackle large dimensions where *Neighbor Context Agents* were difficult if not impossible to find.

One other improvement concerns the way *perceptions* are considered during the learning process. In *Context Learning* and *Endogenous Context Learning*, each perception has the same importance compared to others. It would be interesting to add weight to the *perceptions* in the case that they happen to be heterogeneous. Distances and *neighborhoods* would also be affected.

Optimization

The work on the optimization of *Context Agents* activation is a first step towards full optimization of *Context Learning*. One long term goal is to distribute and parallelize most of the operations to optimize the *Scalability* of the learning process. The objective would be to design an architecture to represent *Context Agents* spatially and speedup their interactions. Close *Context Agents* would perceive their neighbors without having to compare their *validity ranges* with all the close *Context Agents*.

Reinforcement Learning

It has been seen that *Endogenous Context Learning* offers interesting leads for performing *Reinforcement Learning* in a continuous state and action space. *Cooperative Neighborhood Learning* could enable to generate a continuous learning policy composed of a collective of *Context Agents*. This collective would then be exploited to seek better rewarded situations.

Multi-Task Learning

Another perspective of *Endogenous Context Learning* would be to learn several tasks. In the sense of *Multi-Task Learning*, it would mean that all learned tasks would communicate with each other to improve their respective performances. In *Context Learning*, a task is represented by a collective of *Context Agents*. In order for these collective to exchange useful

information, new communication mechanisms between set of *Context Agents* would need to be designed.

In robotics, learning several tasks could mean for a robot to know several skills (move, grab an object, place an object...). Each skill would be represented by a set of *Context Agents*. Estimating which skill to use depending on a specific situation would then require additional processes to seek and use the appropriate collective of *Context Agents* from a database with several sets of agents (i.e. skills).

Meta-Learning

The implemented *Meta-Learning* approach was only used with linear regression. Future work could involve testing other learning models to validate the compatibility of the design mechanisms with them. The measure of affinity and similarity would need to be specified for the selected learning model. As said before, the interface could also provide interesting representation and visualization of other learning techniques that are difficult to explain like *Deep Neural Networks*.

Robotics

The implemented robotic experiments focus on 2-dimensional robotic arms. One interesting perspective would be to extend this work to 3-dimensional robotic arms. The constraint propagation mechanism of the distributed control for the *Inverse Kinematic* problem provides promising leads for multi-goal arm positioning. Considering a robotic hand with several goal positions for the fingers ends. Each goal could be propagated to the joints were several segment meet and would then be merged to create higher level goals. This way, complex skeletons could be controlled in a distributed way.

Towards Non Finality...

This work adds a stone to the great edifice that is the *non finality* in *AI*. The challenge of *non finality* [Gleizes et al., 1999] is to design learning mechanisms that have not any prior task dependent purposes. Their goal is to find their purpose and develop skills that are adapted to the environment they are immersed in. The problems that are addressed in *Developmental Learning* join this challenge. They seek to endow learning mechanisms with autonomous knowledge generation capabilities [Cangelosi and Schlesinger, 2015]. A major focus of *Developmental Learning* is also *Intrinsic Motivation* which goal is to replicate curiosity mechanisms in order to improve the exploration and the discovery of new skills or purposes [Forestier and Oudeyer, 2016]. *Artificial General Intelligence* (*AGI*) objectives, which are infinite generality, adaptability and flexibility, are also in line with the scientific lock of *non finality* [Goertzel, 2014].

This thesis provided mechanisms for endogenous enhancement in the direction of *non finality*. This work proves that *Context Learning* and *Adaptive Multi-Agent Systems* (*AMAS*) provide generality, adaptability and flexibility to a learning mechanism, fundamentals prop-

erties for *AGI*. The distribution and the locality of *Endogenous Context Learning* enabled introspection during the learning process, an essential mechanism for *non finality*.

The future of *AI* promises interesting advances in the understanding of human cognition and in its artificial replication. One question remains, how such future artificial intelligent forms will be considered and accepted by humans and their eternal fear of the unknown. The explainability of these systems will certainly have an important role to play in this acceptance.

Appendices

Bibliography

[Aamodt and Plaza, 1994] Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59.

[Abbeel and Ng, 2011] Abbeel, P. and Ng, A. Y. (2011). Inverse reinforcement learning. In *Encyclopedia of machine learning*, pages 554–558. Springer.

[Ackley et al., 1985] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

[Aggarwal, 2013] Aggarwal, C. C. (2013). A survey of stream clustering algorithms.

[Aljundi et al., 2018] Aljundi, R., Rohrbach, M., and Tuytelaars, T. (2018). Selfless sequential learning. *arXiv preprint arXiv:1806.05421*.

[Allen, 1983] Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

[Alzetta et al., 2020] Alzetta, F., Giorgini, P., Najjar, A., Schumacher, M. I., and Calvaresi, D. (2020). In-time explainability in multi-agent systems: Challenges, opportunities, and roadmap. In *International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*, pages 39–53. Springer.

[Argall et al., 2009] Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483.

[Aristidou et al., 2018] Aristidou, A., Lasenby, J., Chrysanthou, Y., and Shamir, A. (2018). Inverse kinematics techniques in computer graphics: A survey. In *Computer Graphics Forum*, volume 37, pages 35–58. Wiley Online Library.

[Arrow, 1962] Arrow, K. J. (1962). The economic implications of learning by doing. *The review of economic studies*, 29(3):155–173.

[Asada et al., 2009] Asada, M., Hosoda, K., Kuniyoshi, Y., Ishiguro, H., Inui, T., Yoshikawa, Y., Ogino, M., and Yoshida, C. (2009). Cognitive developmental robotics: A survey. *IEEE Transactions on Autonomous Mental Development*, 1(1):12–34.

[Ashby and Goldstein, 2011] Ashby, W. R. and Goldstein, J. (2011). Variety, constraint, and the law of requisite variety. *Emergence: Complexity and Organization*, 13(1/2):190.

[Baldassarre, 2011] Baldassarre, G. (2011). What are intrinsic motivations? a biological perspective. In *Development and learning (icdl), 2011 ieee international conference on*, volume 2, pages 1–8. IEEE.

[Baranes and Oudeyer, 2013] Baranes, A. and Oudeyer, P.-Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73.

[Baresi and Ghezzi, 2010] Baresi, L. and Ghezzi, C. (2010). The disappearing boundary between development-time and run-time. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, pages 17–22.

[Bayle et al., 2003] Bayle, B., Fourquet, J.-Y., and Renaud, M. (2003). Manipulability of wheeled mobile manipulators: Application to motion generation. *The International Journal of Robotics Research*, 22(7-8):565–581.

[Beer, 2014] Beer, R. D. (2014). Dynamical systems and embedded cognition. *The Cambridge handbook of artificial intelligence*, (812):856–873.

[Bellas et al., 2010] Bellas, F., Duro, R. J., Faiña, A., and Souto, D. (2010). Multilevel darwinist brain (mdb): Artificial evolution in a cognitive architecture for real robots. *IEEE Transactions on autonomous mental development*, 2(4):340–354.

[Bendre et al., 2020] Bendre, N., Marín, H. T., and Najafirad, P. (2020). Learning from few samples: A survey. *arXiv preprint arXiv:2007.15484*.

[Bengio et al., 2017] Bengio, Y., Goodfellow, I., and Courville, A. (2017). *Deep learning*, volume 1. MIT press Massachusetts, USA:.

[Bernon et al., 2002] Bernon, C., Gleizes, M.-P., Peyruqueou, S., and Picard, G. (2002). Adelfe: A methodology for adaptive multi-agent systems engineering. In *International Workshop on Engineering Societies in the Agents World*, pages 156–169. Springer.

[Bishop, 2013] Bishop, C. M. (2013). Model-based machine learning. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20120222.

[Boes, 2014] Boes, J. (2014). *Apprentissage du contrôle de systèmes complexes par l'auto-organisation coopérative d'un système multi-agent: application à la calibration de moteurs à combustion*. PhD thesis, Université Toulouse III Paul Sabatier.

[Bondu and Lemaire, 2007] Bondu, A. and Lemaire, V. (2007). Active learning using adaptive curiosity. In *International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*.

[Bottou, 2010] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

[Breiman et al., 1984] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

[Cangelosi and Schlesinger, 2015] Cangelosi, A. and Schlesinger, M. (2015). *Developmental robotics: From babies to robots*. MIT press.

[Cao et al., 2010] Cao, B., Pan, S. J., Zhang, Y., Yeung, D.-Y., and Yang, Q. (2010). Adaptive transfer learning. In *proceedings of the AAAI Conference on Artificial Intelligence*, volume 24.

[Capera et al., 2003] Capera, D., Georgé, J.-P., Gleizes, M.-P., and Glize, P. (2003). Emergence of organisations, emergence of functions. In *AISB'03 symposium on Adaptive Agents and Multi-Agent Systems*, pages 103–108.

[Chaput, 2004] Chaput, H. H. (2004). *The constructivist learning architecture: A model of cognitive development for robust autonomous robots*. PhD thesis.

[Chen and Liu, 2018] Chen, Z. and Liu, B. (2018). Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207.

[Chen et al., 2019] Chen, Z., Wang, X., Xie, X., Wu, T., Bu, G., Wang, Y., and Chen, E. (2019). Co-attentive multi-task learning for explainable recommendation. In *IJCAI*, pages 2137–2143.

[Cheng et al., 2015] Cheng, H., Fang, H., and Ostendorf, M. (2015). Open-domain name error detection using a multi-task rnn. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 737–746.

[Chollet et al., 2018] Chollet, F. et al. (2018). *Deep learning with Python*, volume 361. Manning New York.

[Ciregan et al., 2012] Ciregan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. In *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pages 3642–3649. IEEE.

[Clarke et al., 2009] Clarke, J., Connors, J., and Bruno, E. J. (2009). *JavaFX: developing rich Internet applications*. Pearson Education.

[Creane, 1995] Creane, A. (1995). Endogenous learning, learning by doing and information sharing. *International Economic Review*, pages 985–1002.

[Daglarli, 2021] Daglarli, E. (2021). Explainable artificial intelligence (xai) approaches and deep meta-learning models for cyber-physical systems. In *Artificial Intelligence Paradigms for Smart Cyber-Physical Systems*, pages 42–67. IGI Global.

[Dato. et al., 2021a] Dato., B., Gleizes., M., and Migeon., F. (2021a). Cooperative neighborhood learning: Application to robotic inverse model. In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART,*, pages 368–375. INSTICC, SciTePress.

[Dato. et al., 2021b] Dato., B., Gleizes., M., and Migeon., F. (2021b). A local active learning strategy by cooperative multi-agent systems. In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART,*, pages 406–413. INSTICC, SciTePress.

[Davis and Marcus, 2015] Davis, E. and Marcus, G. (2015). Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103.

[Drescher, 1991] Drescher, G. L. (1991). *Made-up minds: a constructivist approach to artificial intelligence*. MIT press.

[Drogoul et al., 2013] Drogoul, A., Amouroux, E., Caillou, P., Gaudou, B., Grignard, A., Marilleau, N., Taillandier, P., Vavasseur, M., Vo, D.-A., and Zucker, J.-D. (2013). Gama: multi-level and complex environment for agent-based models and simulations. In *12th International Conference on Autonomous agents and multi-agent systems*, pages 2–p. Ifaamas.

[Duan et al., 2017] Duan, Y., Andrychowicz, M., Stadie, B., Ho, O. J., Schneider, J., Sutskever, I., Abbeel, P., and Zaremba, W. (2017). One-shot imitation learning. In *Advances in neural information processing systems*, pages 1087–1098.

[Duan et al., 2016] Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. (2016). Rl2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*.

[Dudley and Kristensson, 2018] Dudley, J. J. and Kristensson, P. O. (2018). A review of user interface design for interactive machine learning. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 8(2):1–37.

[Fei-Fei et al., 2006] Fei-Fei, L., Fergus, R., and Perona, P. (2006). One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611.

[Ferber, 1999] Ferber, J. (1999). *Multi-agent systems: an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading.

[Fernandez-Gauna et al., 2018] Fernandez-Gauna, B., Osa, J. L., and Graña, M. (2018). Experiments of conditioned reinforcement learning in continuous space control tasks. *Neurocomputing*, 271:38–47.

[Finn et al., 2017] Finn, C., Abbeel, P., and Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR.

[Finn et al., 2019] Finn, C., Rajeswaran, A., Kakade, S., and Levine, S. (2019). Online meta-learning. *arXiv preprint arXiv:1902.08438*.

[Fix and Hodges Jr, 1951] Fix, E. and Hodges Jr, J. L. (1951). Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, California Univ Berkeley.

[Forestier and Oudeyer, 2016] Forestier, S. and Oudeyer, P.-Y. (2016). Curiosity-driven development of tool use precursors: a computational model. In *38th annual conference of the cognitive science society (cogsci 2016)*, pages 1859–1864.

[Freer et al., 2012] Freer, C. E., Roy, D. M., and Tenenbaum, J. B. (2012). Towards common-sense reasoning via conditional simulation: Legacies of turing in artificial intelligence. *arXiv preprint arXiv:1212.4799*.

[Gardner, 2011] Gardner, H. (2011). *Frames of mind: The theory of multiple intelligences*. Hachette UK.

[Gardner and Velinsky, 2000] Gardner, J. F. and Velinsky, S. A. (2000). Kinematics of mobile manipulators and implications for design. *Journal of Robotic Systems*, 17(6):309–320.

[Georgé et al., 2011] Georgé, J.-P., Gleizes, M.-P., and Camps, V. (2011). Cooperation. In *Self-organising Software*, pages 193–226. Springer.

[Gerostathopoulos et al., 2019] Gerostathopoulos, I., Skoda, D., Plasil, F., Bures, T., and Knauss, A. (2019). Tuning self-adaptation in cyber-physical systems through architectural homeostasis. *Journal of Systems and Software*, 148:37–55.

[Ghahramani, 2015] Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452.

[Gibney, 2016] Gibney, E. (2016). Google ai algorithm masters ancient game of go. *Nature News*, 529(7587):445.

[Gittins et al., 2011] Gittins, J., Glazebrook, K., and Weber, R. (2011). *Multi-armed bandit allocation indices*. John Wiley & Sons.

[Gleizes, 2011] Gleizes, M.-P. (2011). Self-adaptive complex systems. In *European Workshop on Multi-Agent Systems*, pages 114–128. Springer.

[Gleizes, 2012] Gleizes, M.-P. (2012). Self-adaptive Complex Systems (regular paper). In Cossentino, M., Kaisers, M., Tuyls, K., and Weiss, G., editors, *European Workshop on Multi-Agent Systems (EUMAS 2011), Maastricht, The Netherlands, 13/11/11-16/11/11*, volume 7541, pages 114–128, http://www.springerlink.com/. Springer-Verlag. (Conférencier invité).

[Gleizes et al., 1999] Gleizes, M.-P., Camps, V., and Glize, P. (1999). A theory of emergent computation based on cooperative self-organization for adaptive artificial systems. In *Fourth European Congress of Systems Science*, pages 20–24.

[Goertzel, 2014] Goertzel, B. (2014). Artificial general intelligence: concept, state of the art, and future prospects. *Journal of Artificial General Intelligence*, 5(1):1–48.

[Goldberg et al., 2011] Goldberg, A., Zhu, X., Furger, A., and Xu, J.-M. (2011). Oasis: Online active semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25.

[Granulo et al., 2019] Granulo, A., Fuchs, C., and Puntoni, S. (2019). Psychological reactions to human versus robotic job replacement. *Nature human behaviour*, 3(10):1062–1069.

[Graves et al., 2014] Graves, A., Wayne, G., and Danihelka, I. (2014). Neural turing machines. *arXiv preprint arXiv:1410.5401*.

[Graves et al., 2016] Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., et al. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471.

[Guériau, 2016] Guériau, M. (2016). *Systèmes multi-agents, auto-organisation et contrôle par apprentissage constructiviste pour la modélisation et la régulation dans les systèmes coopératifs de trafic*. PhD thesis, Université de Lyon I Claude Bernard.

[Guériau et al., 2016] Guériau, M., Armetta, F., Hassas, S., Billot, R., and El Faouzi, N.-E. (2016). A constructivist approach for a self-adaptive decision-making system: application to road traffic control. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 670–677. IEEE.

[Guerin, 2011] Guerin, F. (2011). Learning like a baby: a survey of artificial intelligence approaches. *The Knowledge Engineering Review*, 26(2):209–236.

[Gui et al., 2018] Gui, L.-Y., Wang, Y.-X., Ramanan, D., and Moura, J. M. (2018). Few-shot human motion prediction via meta-learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 432–450.

[Guivarch et al., 2013] Guivarch, V., Camps, V., and Péninou, A. (2013). Amadeus: an adaptive multi-agent system to learn a user's recurring actions in ambient systems. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 1(3):1–10.

[Hagar, 2014] Hagar, A. (2014). *Discrete or continuous?: the quest for fundamental length in modern physics*. Cambridge University Press.

[Hao et al., 2018] Hao, S., Hu, P., Zhao, P., Hoi, S. C., and Miao, C. (2018). Online active learning with expert advice. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(5):1–22.

[Hartigan and Wong, 1979] Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). Unsupervised learning. In *The elements of statistical learning*, pages 485–585. Springer.

[Henaff et al., 2011] Henaff, M., Jarrett, K., Kavukcuoglu, K., and LeCun, Y. (2011). Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, volume 11, page 2011.

[Henderson et al., 2018] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. (2018). Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

[Heuillet et al., 2020] Heuillet, A., Couthouis, F., and Díaz-Rodríguez, N. (2020). Explainability in deep reinforcement learning. *Knowledge-Based Systems*, page 106685.

[Ho et al., 2013] Ho, E. S., Shum, H. P., Cheung, Y.-m., and Yuen, P. C. (2013). Topology aware data-driven inverse kinematics. In *Computer Graphics Forum*, volume 32, pages 61–70. Wiley Online Library.

[Hoi et al., 2018] Hoi, S. C., Sahoo, D., Lu, J., and Zhao, P. (2018). Online learning: A comprehensive survey. *arXiv preprint arXiv:1802.02871*.

[Holden et al., 2016] Holden, D., Saito, J., and Komura, T. (2016). A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)*, 35(4):1–11.

[Holland, 2006] Holland, J. H. (2006). Studying complex adaptive systems. *Journal of systems science and complexity*, 19(1):1–8.

[Holland and Reitman, 1978] Holland, J. H. and Reitman, J. S. (1978). Cognitive systems based on adaptive algorithms. In *Pattern-directed inference systems*, pages 313–329. Elsevier.

[Holmes et al., 2005] Holmes, M. P. et al. (2005). Schema learning: Experience-based construction of predictive action models. In *Advances in Neural Information Processing Systems*, pages 585–592.

[Holmgren, 2014] Holmgren, J. (2014). Natural evolution and human consciousness. *Mens sana monographs*, 12(1):127.

[Hong et al., 2018] Hong, X., Wong, P., Liu, D., Guan, S.-U., Man, K. L., and Huang, X. (2018). Lifelong machine learning: outlook and direction. In *Proceedings of the 2nd International Conference on Big Data Research*, pages 76–79.

[Hootsmans and Dubowsky, 1991] Hootsmans, N. and Dubowsky, S. (1991). Large motion control of mobile manipulators including vehicle suspension characteristics. In *ICRA*, volume 91, pages 2336–2341.

[Hopfield, 1982] Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558.

[Huang et al., 2013a] Huang, J.-T., Li, J., Yu, D., Deng, L., and Gong, Y. (2013a). Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7304–7308. IEEE.

[Huang et al., 2013b] Huang, Y., Wang, W., Wang, L., and Tan, T. (2013b). Multi-task deep neural network for multi-label learning. In *2013 IEEE International Conference on Image Processing*, pages 2897–2900. IEEE.

[IFR, 2020] IFR (2020). Ifr presents world robotics report 2020. `https://ifr.org/ifr-press-releases/news/record-2.7-million-robots-work-in-factories-around-the-globe`. Accessed: 2021-04-07.

[Jamali et al., 2011] Jamali, A., Khan, R., and Rahman, M. M. (2011). A new geometrical approach to solve inverse kinematics of hyper redundant robots with variable link length. In *2011 4th International Conference on Mechatronics (ICOM)*, pages 1–5. IEEE.

[Jia and Liang, 2017] Jia, R. and Liang, P. (2017). Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

[Kansky et al., 2017] Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, S., and George, D. (2017). Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1809–1818. JMLR. org.

[Kearns et al., 1994] Kearns, M. J., Schapire, R. E., and Sellie, L. M. (1994). Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141.

[Kephart and Chess, 2003] Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1):41–50.

[Khoreva et al., 2017] Khoreva, A., Benenson, R., Ilg, E., Brox, T., and Schiele, B. (2017). Lucid data dreaming for object tracking. In *The DAVIS challenge on video object segmentation*.

[Kirkpatrick et al., 2017] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

[Kloft and Laskov, 2012] Kloft, M. and Laskov, P. (2012). Security analysis of online centroid anomaly detection. *The Journal of Machine Learning Research*, 13(1):3681–3724.

[Kohonen, 1998] Kohonen, T. (1998). The self-organizing map. *Neurocomputing*, 21(1-3):1–6.

[Krueger et al., 2020] Krueger, D., Leike, J., Evans, O., and Salvatier, J. (2020). Active reinforcement learning: Observing rewards at a cost. *arXiv preprint arXiv:2011.06709*.

[Kurzweil, 2005] Kurzweil, R. (2005). *The singularity is near: When humans transcend biology*. Penguin.

[Lake and Baroni, 2018] Lake, B. and Baroni, M. (2018). Still not systematic after all these years: On the compositional skills of sequence-to-sequence recurrent networks.

[Lake et al., 2011] Lake, B., Salakhutdinov, R., Gross, J., and Tenenbaum, J. (2011). One shot learning of simple visual concepts. In *Proceedings of the annual meeting of the cognitive science society*, volume 33.

[Lake et al., 2017] Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40.

[Lampert et al., 2009] Lampert, C. H., Nickisch, H., and Harmeling, S. (2009). Learning to detect unseen object classes by between-class attribute transfer. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 951–958. IEEE.

[Lazer et al., 2014] Lazer, D., Kennedy, R., King, G., and Vespignani, A. (2014). The parable of google flu: traps in big data analysis. *Science*, 343(6176):1203–1205.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436.

[Lemke et al., 2015] Lemke, C., Budka, M., and Gabrys, B. (2015). Metalearning: a survey of trends and technologies. *Artificial intelligence review*, 44(1):117–130.

[Lemouzy, 2011] Lemouzy, S. (2011). *Systèmes interactifs auto-adaptatifs par systèmes multi-agents auto-organisateurs: application à la personnalisation de l'accès à l'information*. PhD thesis, Université de Toulouse, Université Toulouse III-Paul Sabatier.

[Li et al., 2019] Li, A., Luo, T., Lu, Z., Xiang, T., and Wang, L. (2019). Large-scale few-shot learning: Knowledge transfer with class hierarchy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7212–7220.

[Li et al., 2009] Li, H., Liao, X., and Carin, L. (2009). Multi-task reinforcement learning in partially observable stochastic environments. *Journal of Machine Learning Research*, 10(5).

[Li et al., 2017] Li, Y.-F., Zha, H.-W., and Zhou, Z.-H. (2017). Learning safe prediction for semi-supervised regression. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[Liu et al., 2015] Liu, X., Gao, J., He, X., Deng, L., Duh, K., and Wang, Y.-y. (2015). Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 912–921.

[Lopez-Paz et al., 2017] Lopez-Paz, D. et al. (2017). Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6470–6479.

[Lu and Tang, 2015] Lu, C. and Tang, X. (2015). Surpassing human-level face verification performance on lfw with gaussianface. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

[Lucic et al., 2017] Lucic, M., Faulkner, M., Krause, A., and Feldman, D. (2017). Training mixture models at scale via coresets. *arXiv preprint arXiv:1703.08110*.

[Lughofer, 2017] Lughofer, E. (2017). On-line active learning: A new paradigm to improve practical useability of data stream modeling methods. *Information Sciences*, 415:356–376.

[Lyre, 2020] Lyre, H. (2020). The state space of artificial intelligence. *Minds and Machines*, 30(3):325–347.

[MacGlashan and Littman, 2015] MacGlashan, J. and Littman, M. L. (2015). Between imitation and intention learning. In *IJCAI*, pages 3692–3698.

[Mahdavi-Hezavehi et al., 2016] Mahdavi-Hezavehi, S., Avgeriou, P., Weyns, D., Mistrik, I., Ali, N., Kazman, R., John, G., and Schmerl, B. (2016). A classification of current architecture-based approaches tackling uncertainty in self-adaptive systems with multiple requirements. *Managing Trade-offs in Adaptable Software Architectures. Elsevier*.

[Mannucci et al., 2017] Mannucci, T., van Kampen, E.-J., de Visser, C., and Chu, Q. (2017). Safe exploration algorithms for reinforcement learning controllers. *IEEE transactions on neural networks and learning systems*, 29(4):1069–1081.

[Marcus, 2018] Marcus, G. (2018). Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*.

[Marcus, 1998] Marcus, G. F. (1998). Rethinking eliminative connectionism. *Cognitive psychology*, 37(3):243–282.

[Mazac et al., 2014] Mazac, S., Armetta, F., and Hassas, S. (2014). On bootstrapping sensorimotor patterns for a constructivist learning system in continuous environments. In *Artificial Life Conference Proceedings 14*, pages 160–167. MIT Press.

[Mazac et al., 2015] Mazac, S., Armetta, F., and Hassas, S. (2015). Approche décentralisée pour un apprentissage constructiviste en environnement continu: application à l'intelligence ambiante. In *Journées Francophones sur les Systèmes Multi-Agents (JFSMA)*. CÉPADUÈS ÉDITIONS.

[Mendez et al., 2018] Mendez, J., Shivkumar, S., and Eaton, E. (2018). Lifelong inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4502–4513.

[Metta et al., 2010] Metta, G., Natale, L., Nori, F., Sandini, G., Vernon, D., Fadiga, L., Von Hofsten, C., Rosander, K., Lopes, M., Santos-Victor, J., et al. (2010). The icub humanoid robot: An open-systems platform for research in cognitive development. *Neural Networks*, 23(8-9):1125–1134.

[Miller, 1992] Miller, A. J. (1992). Algorithm as 274: Least squares routines to supplement those of gentleman. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 41(2):458–478.

[Minato et al., 2007] Minato, T., Yoshikawa, Y., Noda, T., Ikemoto, S., Ishiguro, H., and Asada, M. (2007). Cb2: A child robot with biomimetic body for cognitive developmental robotics. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 557–562. IEEE.

[Mirolli and Baldassarre, 2013] Mirolli, M. and Baldassarre, G. (2013). Functions and mechanisms of intrinsic motivations. In *Intrinsically Motivated Learning in Natural and Artificial Systems*, pages 49–72. Springer.

[Mitchell, 1998] Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.

[Mnih et al., 2015] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

[Montazeri et al., 2011] Montazeri, H., Moradi, S., and Safabakhsh, R. (2011). Continuous state/action reinforcement learning: A growing self-organizing map approach. *Neurocomputing*, 74(7):1069–1082.

[Muccini et al., 2016] Muccini, H., Sharaf, M., and Weyns, D. (2016). Self-adaptation for cyber-physical systems: a systematic literature review. In *Proceedings of the 11th international symposium on software engineering for adaptive and self-managing systems*, pages 75–81.

[Nahavandi, 2019] Nahavandi, S. (2019). Industry 5.0—a human-centric solution. *Sustainability*, 11(16):4371.

[Ng and Russell, 2000] Ng, A. Y. and Russell, S. (2000). Algorithms for inverse reinforcement learning. In *in Proc. 17th International Conf. on Machine Learning*. Citeseer.

[Nguyen et al., 2015] Nguyen, A., Yosinski, J., and Clune, J. (2015). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436.

[Nigon, 2017] Nigon, J. (2017). *Apprentissage artificiel adapté aux systèmes complexes par auto-organisation coopérative de systèmes multi-agents*. PhD thesis, Université Toulouse III Paul Sabatier.

[Nigon et al., 2016] Nigon, J., Gleizes, M.-P., and Migeon, F. (2016). Self-adaptive model generation for ambient systems. *Procedia Computer Science*, 83:675–679.

[Nigon et al., 2017] Nigon, J., Verstaevel, N., Boes, J., Migeon, F., and Gleizes, M.-P. (2017). Smart is a matter of context. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 189–202. Springer.

[Nunes and Demiris, 2019] Nunes, U. M. and Demiris, Y. (2019). Online unsupervised learning of the 3d kinematic structure of arbitrary rigid bodies. In *Proceedings of the IEEE international conference on computer vision*, pages 3809–3817.

[Obamuyide and Vlachos, 2019] Obamuyide, A. and Vlachos, A. (2019). Meta-learning improves lifelong relation extraction. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 224–229.

[Orabona and Cesa-Bianchi, 2011] Orabona, F. and Cesa-Bianchi, N. (2011). Better algorithms for selective sampling. In *International conference on machine learning*, pages 433–440. Omnipress.

[Oreizy et al., 1998] Oreizy, P., Medvidovic, N., and Taylor, R. N. (1998). Architecture-based runtime software evolution. In *Proceedings of the 20th international conference on Software engineering*, pages 177–186. IEEE.

[Oudeyer et al., 2014] Oudeyer, P.-Y. et al. (2014). Socially guided intrinsic motivation for robot learning of motor skills. *Autonomous Robots*, 36(3):273–294.

[Palatucci et al., 2009] Palatucci, M., Pomerleau, D., Hinton, G. E., and Mitchell, T. M. (2009). Zero-shot learning with semantic output codes. In *Advances in neural information processing systems*, pages 1410–1418.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

[Papoutsakis and Argyros, 2019] Papoutsakis, K. E. and Argyros, A. A. (2019). Unsupervised and explainable assessment of video similarity. In *British Machine Vision Conference BMVC*, page 151.

[Parisi et al., 2019] Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

[Pentina et al., 2015] Pentina, A., Sharmanska, V., and Lampert, C. H. (2015). Curriculum learning of multiple tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5492–5500.

[Perles et al., 2018] Perles, A., Crasnier, F., and Georgé, J.-P. (2018). Amak-a framework for developing robust and open adaptive multi-agent systems. In *International Conference on Practical Applications of Agents and Multi-Agent Systems*, pages 468–479. Springer.

[Perotto, 2013] Perotto, F. S. (2013). A computational constructivist model as an anticipatory learning mechanism for coupled agent–environment systems. *Constructivist Foundations*, 9(1):46–56.

[Pfeifer and Bongard, 2006] Pfeifer, R. and Bongard, J. (2006). *How the body shapes the way we think: a new view of intelligence*. MIT press.

[Piaget, 1977] Piaget, J. (1977). *La naissance de l'intelligence chez l'enfant*, volume 370. Delachaux et Niestlé Neufchâtel, Switzerland.

[Piaget, 1978] Piaget, J. (1978). *La formation du symbole chez l'enfant: imitation, jeu et rêve, image et représentation*. FeniXX.

[Picard and Gleizes, 2002] Picard, G. and Gleizes, M.-P. (2002). An agent architecture to design self-organizing collectives: Principles and application. In *Adaptive agents and multi-agent systems*, pages 141–158. Springer.

[Provost et al., 2006] Provost, J., Kuipers, B. J., and Miikkulainen, R. (2006). Developing navigation behavior through self-organizing distinctive-state abstraction. *Connection Science*, 18(2):159–172.

[Qahtan et al., 2016] Qahtan, A., Wang, S., and Zhang, X. (2016). Kde-track: An efficient dynamic density estimator for data streams. *IEEE Transactions on Knowledge and Data Engineering*, 29(3):642–655.

[Rebuffi et al., 2017] Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

[Revzen and Koditschek, 2017] Revzen, S. and Koditschek, D. E. (2017). Why we need more degrees of freedom. *Procedia IUTAM*, 20:89–93.

[Ribeiro et al., 2016] Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM.

[Rosenblatt, 1957] Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

[Roy, 2011] Roy, D. M. (2011). *Computability, inference and modeling in probabilistic programming*. PhD thesis, Massachusetts Institute of Technology.

[Rueda et al., 2019] Rueda, F. M., Lüdtke, S., Schröder, M., Yordanova, K., Kirste, T., and Fink, G. A. (2019). Combining symbolic reasoning and deep learning for human activity recognition. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 22–27. IEEE.

[Russell and Norvig, 2016] Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

[Ruvolo and Eaton, 2013] Ruvolo, P. and Eaton, E. (2013). Ella: An efficient lifelong learning algorithm. In *International Conference on Machine Learning*, pages 507–515.

[Sabour et al., 2017] Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866.

[Sahoo et al., 2017] Sahoo, D., Pham, Q., Lu, J., and Hoi, S. C. (2017). Online deep learning: Learning deep neural networks on the fly. *arXiv preprint arXiv:1711.03705*.

[Samek and Müller, 2019] Samek, W. and Müller, K.-R. (2019). Towards explainable artificial intelligence. In *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 5–22. Springer.

[Samek et al., 2017] Samek, W., Wiegand, T., and Müller, K.-R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*.

[Saputri and Lee, 2020] Saputri, T. R. D. and Lee, S.-W. (2020). The application of machine learning in self-adaptive systems: A systematic literature review. *IEEE Access*, 8:205948–205967.

[Sayed, 2014] Sayed, A. H. (2014). Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7(ARTICLE):311–801.

[Schaul et al., 2015] Schaul, T., Horgan, D., Gregor, K., and Silver, D. (2015). Universal value function approximators. In *International Conference on Machine Learning*, pages 1312–1320.

[Schultz, 2013] Schultz, D. (2013). *A History of Modern Psychology*. Academic Press.

[Sculley et al., 2014] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., and Young, M. (2014). Machine learning: The high interest credit card of technical debt.

[Seltzer and Droppo, 2013] Seltzer, M. L. and Droppo, J. (2013). Multi-task learning in deep neural networks for improved phoneme recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6965–6969. IEEE.

[Settles, 2009] Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

[Shalev-Shwartz et al., 2011] Shalev-Shwartz, S. et al. (2011). Online learning and online convex optimization. *Foundations and trends in Machine Learning*, 4(2):107–194.

[Silver et al., 2012] Silver, D., Bagnell, J. A., and Stentz, A. (2012). Active learning from demonstration for robust autonomous navigation. In *2012 IEEE International Conference on Robotics and Automation*, pages 200–207. IEEE.

[Silver et al., 2017] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *nature*, 550(7676):354–359.

[Silver et al., 2013] Silver, D. L., Yang, Q., and Li, L. (2013). Lifelong machine learning systems: Beyond learning algorithms. In *2013 AAAI spring symposium series*.

[Skobelev and Borovik, 2017] Skobelev, P. and Borovik, S. Y. (2017). On the way from industry 4.0 to industry 5.0: From digital manufacturing to digital society. *Industry 4.0*, 2(6):307–311.

[Spelke and Kinzler, 2007] Spelke, E. S. and Kinzler, K. D. (2007). Core knowledge. *Developmental science*, 10(1):89–96.

[Stewart et al., 2020] Stewart, K., Orchard, G., Shrestha, S. B., and Neftci, E. (2020). Online few-shot gesture learning on a neuromorphic processor. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 10(4):512–521.

[Stoytchev, 2009] Stoytchev, A. (2009). Some basic principles of developmental robotics. *IEEE Transactions on Autonomous Mental Development*, 1(2):122–130.

[Sun et al., 2020] Sun, J., Lapuschkin, S., Samek, W., Zhao, Y., Cheung, N.-M., and Binder, A. (2020). Explain and improve: Cross-domain few-shot-learning using explanations. *arXiv preprint arXiv:2007.08790*, 3.

[Sun et al., 2019] Sun, Q., Liu, Y., Chua, T.-S., and Schiele, B. (2019). Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 403–412.

[Sutton, 1996] Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in neural information processing systems*, pages 1038–1044.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

[Thrun and Mitchell, 1995] Thrun, S. and Mitchell, T. M. (1995). Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46.

[Thrun and Pratt, 2012] Thrun, S. and Pratt, L. (2012). *Learning to learn*. Springer Science & Business Media.

[Thuruthel et al., 2016] Thuruthel, T. G., Falotico, E., Cianchetti, M., and Laschi, C. (2016). Learning global inverse kinematics solutions for a continuum robot. In *Symposium on Robot Design, Dynamics and Control*, pages 47–54. Springer.

[Torresen, 2018] Torresen, J. (2018). A review of future and ethical perspectives of robotics and ai. *Frontiers in Robotics and AI*, 4:75.

[Turing, 1950] Turing, A. (1950). Computing machinery and intelligence. *Mind*, 59:433–460.

[Unzueta et al., 2008] Unzueta, L., Peinado, M., Boulic, R., and Suescun, Á. (2008). Full-body performance animation with sequential inverse kinematics. *Graphical models*, 70(5):87–104.

[Vanschoren, 2019] Vanschoren, J. (2019). Meta-learning. In *Automated Machine Learning*, pages 35–61. Springer, Cham.

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

[Vernon et al., 2011] Vernon, D., Von Hofsten, C., and Fadiga, L. (2011). *A roadmap for cognitive development in humanoid robots*, volume 11. Springer Science & Business Media.

[Verstaevel, 2016] Verstaevel, N. (2016). *Self-organization of robotic devices through demonstrations*. PhD thesis, Université Toulouse III Paul Sabatier.

[Verstaevel et al., 2017] Verstaevel, N., Boes, J., Nigon, J., d'Amico, D., and Gleizes, M. P. (2017). Lifelong machine learning with adaptive multi-agent systems. In *ICAART (2)*, pages 275–286.

[Videau, 2011] Videau, S. (2011). *Contrôle de processus dynamiques par systèmes multi-agents adaptatifs: application au contrôle de bioprocédés*. PhD thesis, Toulouse, INSA.

[Vinyals et al., 2016] Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638.

[Vlassis et al., 2012] Vlassis, N., Ghavamzadeh, M., Mannor, S., and Poupart, P. (2012). Bayesian reinforcement learning. In *Reinforcement Learning*, pages 359–386. Springer.

[Wang et al., 2018] Wang, N., Wang, H., Jia, Y., and Yin, Y. (2018). Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 165–174.

[Wang et al., 2016] Wang, S., Wan, J., Zhang, D., Li, D., and Zhang, C. (2016). Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. *Computer Networks*, 101:158–168.

[Wang et al., 2020] Wang, Y., Yao, Q., Kwok, J. T., and Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34.

[Watkins and Dayan, 1992] Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3-4):279–292.

[Weber and Wermter, 2020] Weber, T. and Wermter, S. (2020). Integrating intrinsic and extrinsic explainability: The relevance of understanding neural networks for human-robot interaction. *arXiv preprint arXiv:2010.04602*.

[Wei et al., 2020] Wei, K., Deng, C., and Yang, X. (2020). Lifelong zero-shot learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 551–557.

[Weiss et al., 2016] Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big data*, 3(1):9.

[Weng et al., 2014] Weng, C., Yu, D., Watanabe, S., and Juang, B.-H. F. (2014). Recurrent deep neural networks for robust speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 5532–5536. IEEE.

[Weng et al., 2001] Weng, J., McClelland, J., Pentland, A., Sporns, O., Stockman, I., Sur, M., and Thelen, E. (2001). Autonomous mental development by robots and animals. *Science*, 291(5504):599–600.

[Wertheimer and Hariharan, 2019] Wertheimer, D. and Hariharan, B. (2019). Few-shot learning with localization in realistic settings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6558–6567.

[West et al., 2007] West, J., Ventura, D., and Warnick, S. (2007). Spring research presentation: A theoretical foundation for inductive transfer. *Brigham Young University, College of Physical and Mathematical Sciences*, 1:32.

[Weyns, 2018] Weyns, D. (2018). Engineering self-adaptive software systems–an organized tour. In *2018 IEEE 3rd International Workshops on Foundations and Applications of Self* Systems (FAS* W)*, pages 1–2. IEEE.

[Wickens et al., 2015] Wickens, C. D., Hollands, J. G., Banbury, S., and Parasuraman, R. (2015). *Engineering psychology and human performance*. Psychology Press.

[Wong et al., 2021] Wong, T., Wagner, M., and Treude, C. (2021). Self-adaptive systems: A systematic literature review across categories and domains. *arXiv preprint arXiv:2101.00125*.

[Wu et al., 2017] Wu, Q., Wu, H., Zhou, X., Tan, M., Xu, Y., Yan, Y., and Hao, T. (2017). Online transfer learning with multiple homogeneous or heterogeneous sources. *IEEE Transactions on Knowledge and Data Engineering*, 29(7):1494–1507.

[Wu et al., 2019] Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., and Fu, Y. (2019). Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382.

[Xian et al., 2017] Xian, Y., Schiele, B., and Akata, Z. (2017). Zero-shot learning-the good, the bad and the ugly. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4582–4591.

[Yegnanarayana, 2009] Yegnanarayana, B. (2009). *Artificial neural networks*. PHI Learning Pvt. Ltd.

[Yoon et al., 2018] Yoon, J., Kim, T., Dia, O., Kim, S., Bengio, Y., and Ahn, S. (2018). Bayesian model-agnostic meta-learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7343–7353.

[Zednik, 2019] Zednik, C. (2019). Solving the black box problem: A normative framework for explainable artificial intelligence. *Philosophy & Technology*, pages 1–24.

[Zhan et al., 2017] Zhan, Y., Ammar, H. B., and Taylor, M. E. (2017). Scalable lifelong reinforcement learning. *Pattern Recognition*, 72:407–418.

[Zhang et al., 2018] Zhang, C., Zhao, P., Hao, S., Soh, Y. C., Lee, B. S., Miao, C., and Hoi, S. C. (2018). Distributed multi-task classification: a decentralized online learning approach. *Machine Learning*, 107(4):727–747.

[Zhang and Yang, 2017] Zhang, Y. and Yang, Q. (2017). A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*.

[Zhang et al., 2014] Zhang, Z., Luo, P., Loy, C. C., and Tang, X. (2014). Facial landmark detection by deep multi-task learning. In *European conference on computer vision*, pages 94–108. Springer.

[Zlatev and Balkenius, 2001] Zlatev, J. and Balkenius, C. (2001). Introduction: Why epigenetic robotics?

List of Figures

1.1 Industrial Revolutions 8

2.1 Learning Mechanism Inputs and Outputs. 15

3.1 Traditional *Machine Learning* VS *Transfer Learning* 25
3.2 *Transfer Learning* Formalism 25
3.3 *Transfer Learning* Strategies 26

4.1 Context Agents *validity ranges* 41
4.2 Context Agents linear regression models 42
4.3 *Context Learning* with *AMOEBA* 42
4.4 Simple learning problem of 2 linear models symbolized by different colors . . 47

5.1 *Exogenous Learning* VS *Endogenous Learning* 51
5.2 Schemas of ideal and imperfect explorations. The imperfect exploration shows all possible exploration inaccuracies in *Context Learning*. Different colors represent different *local models* 54
5.3 Schemas of ideal and imprecise learning for linear *local models*. The search space is supposed entirely explored and represented by *Context Agents validity ranges*. The gray thick dashes represent the *hidden function* $\mathcal{F}$, and the black line are the learned *local models* $f_1^{j_i}$. 55
5.4 Schema of the *neighborhood* of the *perceptions* $\mathcal{P}_2^{c_l}$ and the *Context Agents influences* for 2 perceptions p_1 and p_2.The *neighborhood* is represented with thick dashes and *influences* are represented with thin dashes. 58
5.5 Schema of *Prediction Neighbor Context Agents* for one perception p_1 and one prediction output o_1. $o_1^{j_1,last}$, $o_1^{j_2,last}$ and $o_1^{j_3,last}$ are the last predictions of the *Context Agents* $\mathcal{C}_1^{j_1}$, $\mathcal{C}_1^{j_2}$ and $\mathcal{C}_1^{j_3}$ represented by the local models $f_1^{j_1}$, $f_1^{j_2}$ and $f_1^{j_3}$. $\mathcal{C}_1^{j_1}$ and $\mathcal{C}_1^{j_2}$ are prediction neighbors whereas $\mathcal{C}_1^{j_3}$ is not. 58
5.6 Behavior comparison of $2-$combinations C_2^n with linear and degree 2 polynomial functions 59

5.7 *Conflict NCS* Detection. . . . 62
5.8 *Concurrency NCS* Detection. . . . 62
5.9 *Complete Redundancy NCS* Detection. . . . 63
5.10 *Partial Redundancy NCS* Detection. . . . 63
5.11 *Range Ambiguity NCS* Detection. . . . 64
5.12 *Incompetence NCS* Detection. . . . 65
5.13 Possible configurations when comparing one-dimensional *empty areas* (dotted line) with one-dimensional *validity ranges*. . . . 66
5.14 Recursive *incompetent volume* generation for two perceptions and one *Neighbor Context Agent*. . . . 66
5.15 *Model Ambiguity NCS* detection for 1 perception. . . . 67
5.16 *Model Ambiguity NCS* Detection for 2 perceptions. . . . 67
5.17 Continuity and Discontinuity among *Neighbor Context Agents*. . . . 68
5.18 *Conflict NCS* and *Concurrency NCS* Resolutions. . . . 69
5.19 *Range Ambiguity NCS* and *Model Discontinuity NCS* Resolution. . . . 70
5.20 *Incompetence NCS* Resolution. . . . 71
5.21 *Complete Redundancy NCS* Resolution. . . . 72
5.22 *Partial Redundancy NCS* Resolution. . . . 72
5.23 *Model Ambiguity NCS* Resolution for 2 perceptions. . . . 73
5.24 Schema of *Cooperative Neighborhood Learning* aims. The search space is supposed entirely explored and represented by *Context Agents validity ranges*. The gray thick dashes represent the *hidden function* $\mathcal{F}$, and the black lines are the learned *local models* $f_1^{j_i}$. . . . 75
5.25 *Cooperative Neighborhood Learning*. The *endogenous learning situations* are generated in the hatched zones. The *Best Context Agent* ${}^{\mathcal{B}}\mathcal{C}_2^{j_1}$ is receiving the *endogenous learning situations*. The *Prediction Neighbor Context Agents* ${}^{\mathcal{PN}}\mathcal{C}_2^{j_2}$ and ${}^{\mathcal{PN}}\mathcal{C}_2^{j_3}$ are each providing an *endogenous learning situation*. . . . 76

6.1 *Context Learning* VS *Endogenous Context Learning* . . . 80
6.2 *Valid Context Agents* and *Neighbor Context Agents* activation with and without all *perceptions*. Dashes represent the *neighborhood* area and dots represent the influence zones of *Context Agents*. . . . 84
6.3 *Bad Prediction NCS* Resolutions. . . . 87
6.4 *Uselessness NCS* Resolution. . . . 88
6.5 *Unproductivity NCS* expansion resolutions. The *neighborhood* is represented by dashes. The *Good Context Agent* is boxed. For the illustrations, influences are not used. . . . 89

6.6 *Unproductivity NCS* creation resolutions. The *neighborhood* is represented by dashes. The *Good Context Agents* are boxed. For the illustrations, influences are not used. . . . 90
6.7 Normalized *confidence* between 0 and 1. . . . 98
6.8 Distance to a *learning situation* with 2 perceptions. . . . 99
6.9 Probability density for artificial *perceptions* generation . . . 104
6.10 *Active Learning Strategy* . . . 106
6.11 *Self-Learning Strategy* . . . 108
6.12 Context Domain Adaptation . . . 111
6.13 Context Inductive Transfer . . . 112
6.14 Context Multi-Task Learning . . . 113
6.15 Context Reinforcement Learning . . . 114

7.1 Screen-shots of the user interface *AMAKFX*. . . . 127
7.2 2 Perceptions Linear Toy Problem. . . . 128
7.3 Screenshots of *Context Agents* after 500 training cycles with different *NCS* resolution incrementally added. Each color is a different linear model. . . . 129
7.4 Metrics results on the toy learning problem with the *Active Learning Strategy*. The *learning inaccuracies* NCS are incrementally added. The parameters are set to the values of the tables 7.2 and 7.1. . . . 130
7.5 Metrics results on the toy learning problem with the *Active Learning Strategy* with different *validity ranges precisions* $p^{\mathcal{R}} = \{0.05; 0.1; 0.15; 0.2\}$. The other parameters are set to the values of the tables 7.2 and 7.1. . . . 132
7.6 Metrics results on the toy learning problem with the *Active Learning Strategy* with different learning and exploitation concerns for the *learning criticality* and the *exploitation criticality*: all, performance, generalization and experience. The other parameters are set to the values of the tables 7.2 and 7.1. . . . 133
7.7 Captures of the 2 Perceptions Continuous Non Linear Problem . . . 134
7.8 Metrics results on the non linear continuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.3 and 7.1. . . . 136
7.9 Impacted metrics results on the non linear continuous learning problem with the *Active Learning Strategy* (AL) and the *Self-Learning Strategy* (SL) with different *model error margins* $m^f_{err} = \{0.5; 1.0; 1.5\}$. The other parameters are set to the values of the tables 7.3 and 7.1. . . . 137
7.10 Impacted metrics results on the non linear continuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy* with different *discontinuity detection probabilities* $pb^{disc} = \{0.01; 0.05; 0.1\}$. The other parameters are set to the values of the tables 7.3 and 7.1. . . . 137

7.11 Captures of the 2 Perceptions Discontinuous Non Linear Problem 139

7.12 Metrics results on the non linear discontinuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.4 and 7.1. 140

7.13 Metrics results on the toy learning problem with the *Active Learning Strategy* with different *model similarity thresholds* $t^{f}_{sim} = \{0.001; 0.01; 0.1\}$. The other parameters are set to the values of the tables 7.4 and 7.1. 141

7.14 Captures of the 2 Perceptions Multi-Model Problem 143

7.15 Metrics results on the non linear discontinuous learning problem with the *Active Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.4 and 7.1. 144

7.16 Captures of the 2 Perceptions Noisy Model Problem 146

7.17 Metrics results on the noisy learning problem with the *Active Learning Strategy*, the *Active Cooperative Learning Strategy* and the *Self-Learning Strategy*. The parameters are set to the values of the tables 7.6 and 7.1. 148

7.18 Metrics results on the noisy learning problem with the *Active Learning Strategy* (AL), the *Active Cooperative Learning Strategy* (ACL) and the *Self-Learning Strategy* (SL) with different *noise deviation* $2\sigma = \{0; 1; 10\}$. The other parameters are set to the values of the tables 7.6 and 7.1. 149

7.19 Metrics results on the noisy learning problem with the *Active Learning Strategy* (AL), the *Active Cooperative Learning Strategy* (ACL) and the *Self-Learning Strategy* (SL) with different *validity ranges precisions* $p^{\mathcal{R}} = \{0.02; 0.04; 0.06\}$. The other parameters are set to the values of the tables 7.6 and 7.1. 149

7.20 Metrics results on the noisy learning problem with the *Active Learning Strategy* (AL), the *Active Cooperative Learning Strategy* (ACL) and the *Self-Learning Strategy* (SL) with different *perceptions generation coefficients* $\alpha^{\mathcal{P}_{gen}} = \{0.1; 0.5; 1; 2\}$. The other parameters are set to the values of the tables 7.6 and 7.1. 150

7.21 Capture of the 2 Perceptions Lifelong Exploitation Problem 151

7.22 Metrics results on the lifelong exploitation learning problem *Self-Learning Strategy* with different numbers of *exploitation situations* $\mathcal{E}^{lifelong}_{\#} = \{0; 1000; 10000; 20000; 100000\}$. The parameters are set to the values of the tables 7.7 and 7.1. 152

7.23 Prediction error $\mathcal{O}_{Err}$ (%) with the *Self-Learning Strategy* with several *validity ranges precisions* $p^{\mathcal{R}}$ and different numbers of *exploitation situations* $\mathcal{E}^{lifelong}_{\#} = \{0; 1000; 10000; 20000; 100000\}$. The parameters are set to the values of the tables 7.7 and 7.1. 153

7.24 Prediction error $\mathcal{O}_{Err}$ (%) with the *Self-Learning Strategy*; sensibility to the *endogenous learning weight* w^{endo}_{lrn}, the *neighborhood radius coefficient* $\alpha^{\mathcal{N}}$ and the *influence radius coefficient* $\alpha^{\mathcal{I}}$; *exploitation situations* $\mathcal{E}^{lifelong}_{\#} = 10000$. The other parameters are set to the values of the tables 7.7 and 7.1. 153

7.25 Captures of the 2 perceptions non linear continuous (NLC) and discontinuous (NLD) problems and their learning with 500 *learning situations* with the *Naive Learning Strategy* and the *Self-Learning Strategy*. 155

7.26 Metrics results on the non linear continuous (NLC) and discontinuous (NLD) learning problems with few *learning situations*; The *Self-Learning Strategy* and *Naive Learning Strategy* are compared. The other parameters are set to the values of the tables 7.8 and 7.1. 156

7.27 Prediction error $\mathcal{O}_{Err}$ (%) with the *Self-Learning Strategy*; sensibility to the *validity ranges precision* $p^{\mathcal{R}}$, the *neighborhood radius coefficient* $\alpha^{\mathcal{N}}$ and the *influence radius coefficient* $\alpha^{\mathcal{I}}$, the *endogenous learning weight* w_{lrn}^{endo} and the *exogenous learning weight*. The other parameters are set to the values of the tables 7.8 and 7.1. 157

7.28 Captures of learned *validity ranges* for the toy learning problem 159

7.29 Metrics results on the square problems with the *Self-Learning Strategy* and several *perceptions*. $p^{\mathcal{R}} = 0.06$. The other parameters are set to the values of the tables 7.9 and 7.1. 160

7.30 Execution time sums results on the square problem depending on the number of *Context Agents* with the *Self-Learning Strategy* and several *perceptions* $n = \{2; 3; 5; 10\}$. $p^{\mathcal{R}} = 0.02$. The other parameters are set to the values of the tables 7.9 and 7.1. 161

7.31 Execution time sums results on the square problem depending on the number of *perceptions* with the *Self-Learning Strategy* and different *validity ranges precisions* $p^{\mathcal{R}} = \{0.02; 0.06; 0.1\}$. The other parameters are set to the values of the tables 7.9 and 7.1. 161

7.32 Metrics results on the square problems with the *Active Learning Strategy* and several *perceptions*. $p^{\mathcal{R}} = 0.1$ The other parameters are set to the values of the tables 7.9 and 7.1. 162

7.33 Execution time sums results on the square problem depending on the number of *Context Agents* with the *Active Learning Strategy* and several *perceptions* $n = \{2; 3; 5\}$. $p^{\mathcal{R}} = 0.06$. The other parameters are set to the values of the tables 7.9 and 7.1. 162

7.34 Execution time sums results on the square problem depending on the number of *perceptions* with the *Active Learning Strategy* and different *validity ranges precisions* $p^{\mathcal{R}} = \{0.02; 0.06; 0.1\}$. The other parameters are set to the values of the tables 7.9 and 7.1. 163

7.35 Captures of learned *validity ranges* of 2 *perceptions Transfer Learning* problems. 165

7.36 Metrics results on the *Transfer Learning* problems with the *Active Learning Strategy*. The parameters are set to the values of the tables 7.10 and 7.1. 166

8.1 Robotic arm example in a 2D task space with 3 joints. $\mathcal{P}_{\mathcal{T}|F_0} = (x_3, y_3, 0)$, $\mathcal{O}_{\mathcal{T}|F_0} = (\theta_{2|F_0}, 0, 0)$ and $\mathcal{J}_3 = (\theta_0, \theta_1, \theta_2)$. 172

8.2 Direct Kinematic and Inverse Kinematic Models for control. 172
8.3 Goal position and orientation error in a 2-dimensional task space. 175
8.4 Example of an explored reachable space by a simulated robotic arm in a 2-dimensional task space. 175
8.5 *IKM* learning implementation with 3 joints in the centralized setting. 176
8.6 *Validity ranges* for the *IKM* with 2 joints in a 2-dimensional task space. *Perceptions* $\mathcal{P}_3 = (x_2, y_2, \theta_0)$ and *prediction* $\mathcal{O}_1 = \theta_1$. 178
8.7 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on the number of joints without *Cooperative Neighborhood Learning*. 178
8.8 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on *neighborhood radius coefficients* over robotic arms of 2, 3 and 6 joints. 179
8.9 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on *neighborhood radius coefficients* over robotic arms of 10, 20 and 30 joints. 180
8.10 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on mean *endogenous learning situations* over robotic arms of 2, 3 and 6 joints. 180
8.11 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on mean *endogenous learning situations* over robotic arms of 10, 20 and 30 joints; $p^\mathcal{R} = 0.03$. 180
8.12 Mean execution time depending on mean *endogenous learning situations* over robotic arms from 2 to 30 joints. 181
8.13 *IKM* learning and exploitation implementation with 3 joints in the distributed setting. 183
8.14 Goal propagation mechanism; F_0 is the frame of the task space. 184
8.15 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on robotic arm sizes for different learning cycles (10, 25, 50, 100, 200 and 500), with and without *Cooperative Neighborhood Learning* (*CNL*). 186
8.16 Mean goal position errors $\mathcal{E}_\mathcal{P}$ depending on *learning situations* for different robotic arm sizes (3, 6, 10, 20, 30, 50 and 100), with and without *Cooperative Neighborhood Learning* (*CNL*). 187
8.17 Mean goal position and orientation errors depending on robotic arm sizes for different goal propagation cycles (1, 2, 5, 10 and 20), with *Cooperative Neighborhood Learning*, with orientation goal. 188
8.18 Mean experience time execution depending on robotic arm sizes; with orientation goal. 189

List of Tables

2.1 Synthesis on classical learning approaches with respect to the thesis objectives. Legend: + positive match ; - + in development or relative match; - negative match. 22

3.1 Synthesis on *Lifelong Learning* approaches with respect to the thesis objectives. Legend: + positive match ; - + in development or relative match; - negative match. 33

4.1 Synthesis on literature learning approaches with respect to the thesis objectives. Legend: + positive match ; - + in development or relative match; - negative match. 46

5.1 Table of user and designer parameters and distances for the configuration of the *neighborhood* and the *prediction neighborhood*. 57

5.2 13 possible relations between one dimension *validity ranges* 60

5.3 Table of parameters, distances and metrics for the *learning inaccuracies* detection mechanisms. 61

5.4 *NCS* priorities from the highest to the lowest. 73

5.5 Designer parameter for the *Cooperative Neighborhood Learning* mechanism. . . 76

5.6 Table of user and designer parameters, distances and metrics for the *Endogenous Context Learning* principles. 77

6.1 Table different types of *Context Agents* during learning and exploitation cycles. 82

6.2 Table of user and designer parameters and distances involved in the activation of *Valid Context Agents* and *Neighbor Context Agents*. 84

6.3 Table of user and designer parameters, distances, and criticalities involved in learning cycles. 92

6.4 Table of user and designer parameters involved in exploitation cycles. 94

6.5 Table of *confidence* dynamics through the different experiences of *Context Agents* 97

6.6 Table of designer parameters, metrics and criticalities involved in learning and exploitation criticalities. 101
6.7 Table of designer parameters involved in *Lifelong Context Learning*. 105
6.8 Table of parameters, distances, metrics and criticalities involved in *Endogenous Context Learning*. 117

7.1 Table of experimental, user and designer parameters. If not specified, these are the default parameters for this chapter. 122
7.2 Table of experimental parameters for the validation of *learning inaccuracies NCS*.129
7.3 Table of experimental parameters for the experiments on continuous non linearity. 135
7.4 Table of experimental parameters for the experiments on discontinuous non linearity. 140
7.5 Table of experimental parameters for the mutli-model experiments. 144
7.6 Table of experimental parameters for the noisy experiment. 147
7.7 Table of experimental parameters for the transfer experiment. 151
7.8 Table of experimental parameters for the few learning experiment. 156
7.9 Table of experimental parameters to evaluate scalability. 159
7.10 Table of experimental parameters for the transfer experiment. 166

8.1 Table of experimental, user and designer parameters for the learning of the *IKM* with centralized control. 178
8.2 Table of experimental, user and designer parameters for the learning of the *IKM* with distributed control. 185

List of Algorithms

6.1 Execution cycle c_l of *ELLSA* 82
6.2 Execution cycle c_l of a *Percept Agent* 83
6.3 Learning cycle c_l for the *Head Agent* 86
6.4 Exploitation cycle c_l for the *Head Agent* 93

Glossary

Acronyms

AGI *Artificial General Intelligence*. It is theoretical numerical version of the human brain with infinite generality, adaptability and flexibility. Opposed to usual *Narrow AI* that are specific behaviors in specific contexts. 29, 30, 45, 51, 203, 204

AI *Artificial Intelligence*. It is the implemented theories and techniques in order to create machines capable of simulating human intelligence. 1, 3, 8, 9, 15, 20, 22, 23, 29, 30, 32, 35, 36, 38, 44, 45, 46, 51, 79, 111, 112, 113, 116, 195, 203, 204, 233, 234, 235

AMAKFX It is the designed user interface in this theses which provides understandable feedback on *Context Learning* with explainable and graphical information. 127, 225

AMAS *Adaptive Multi-Agent System*. Is is a *MAS* that promotes the emergence of expected global properties using cooperative behaviors between the agents. 40, 44, 45, 46, 52, 59, 124, 198, 203, 233, 234, 242

AMOEBA *Agnostic MOdEl Builder by self Adaptation*. It is distributed *Context Learning* mechanism designed with the *AMAS* theory for continuous environments. 41, 42, 43, 79, 80, 223, 233

AVT *Adaptive Value Tracker*. It is a tool to discover a real dynamic value with successive returns. It rules the *validity ranges* modification increments. 87, 89

CNL *Cooperative Neighborhood Learning*. It is the implemented mechanism within *ELLSA* to generate *Endogenous Learning* using cooperation between *Neighbor Context Agents*. 75, 97, 109, 113, 151, 169, 186, 187, 189, 228, 243, 244

DKM *Direct Kinematic Model*. It calculates the position and orientation of a robotic arm end tool from its joints states. 172, 174, 182, 183

ELLSA *Endogenous Lifelong Learner by Self-Adaptation*. It is the learning mechanism designed in this thesis. It is an enhancement of *AMOEBA* that adds *Endogenous Learning* to the *Context Learning* paradigm. 79, 80, 81, 82, 92, 93, 94, 98, 105, 106, 108, 111, 112, 115, 116, 121, 123, 124, 125, 126, 139, 145, 151, 165, 167, 169, 176, 177, 178, 183, 185, 195, 196, 198, 201, 231, 233, 241

IKM *Inverse Kinematic Model*. It calculates between one and all possible configurations of a robotic arm joints in order to reach a desired position and orientation of its end tool. 172, 173, 174, 176, 177, 178, 183, 184, 185, 186, 201, 228, 230

MAS *Multi-Agent System*. It is an *AI* approach that distributes the resolution of problems into several intelligent and autonomous entities called agents. 40, 44, 56, 79, 127, 233

NCS *Non Cooperative Situations*. They are abnormal situations in an *AMAS* that require special treatments. The design, detection and resolution of these situations are the basis of cooperation within *AMAS*. 43, 47, 52, 59, 64, 67, 69, 70, 73, 74, 81, 82, 86, 87, 91, 93, 107, 108, 121, 125, 128, 129, 130, 131, 132, 133, 135, 136, 140, 141, 144, 148, 152, 155, 156, 160, 162, 163, 166, 167, 199, 201, 225, 229, 230, 238, 239, 242, 243, 244

SAS *Self-Adaptive Systems*. They are systems that can adapt to their environment and its changes. 8, 9

Machine Learning

Active Learning *Active Learning* is a learning technique where the learner selects the labeled data (*exogenous learning situations*). 17, 18, 22, 27, 28, 40, 46, 201

Constructivist Learning *Constructivist Learning* is a theory that states that learning is an incremental construction based on observation and action one an environment. 35, 36, 40, 234

Context Learning *Context Learning* is a learning technique inspired from *Constructivist Learning* that distributes the learning process using the *AMAS* theory. 1, 2, 3, 35, 40, 41, 42, 44, 45, 46, 47, 52, 53, 54, 56, 77, 79, 80, 81, 87, 91, 111, 112, 115, 116, 124, 127, 167, 168, 169, 178, 182, 195, 196, 197, 198, 199, 200, 201, 202, 203, 223, 224, 233, 234, 235

Deep Learning *Deep Learning* is a machine learning technique that uses network architectures inspired from the biological brain to extract high-level features from raw input. 2, 23, 24, 27, 29, 30, 31, 33, 38, 39, 40, 46

Developmental Learning *Developmental Learning* is an application of *Constructivist Learning* in machine learning especially in *Developmental Robotics*. 2, 35, 36, 37, 38, 45, 46, 203

Endogenous Context Learning *Endogenous Context Learning* is an enrichment of *Context Learning* with the addition of *Endogenous Learning*. 1, 2, 51, 52, 53, 77, 80, 112, 113, 116, 117, 191, 196, 197, 198, 200, 201, 202, 204, 224, 229, 230

Endogenous Learning *Endogenous Learning* represents the action of learning with situations which are internal to a learning mechanism (*endogenous learning situations*). 1, 20, 27, 45, 51, 52, 75, 79, 105, 115, 196, 197, 201, 223, 233, 234, 241

Exogenous Learning *Exogenous Learning* represents the action of learning with situations which are external to a learning mechanism (*exogenous learning situations*). 51, 105, 223

Few Shot Learning *Few Shot Learning* bases its learning on just one or a few training examples using prior knowledge. 2, 20, 27, 35, 38, 39, 40, 45, 46

Genetic Algorithms *Genetic Algorithms* is a biologically inspired learning technique that uses a fitness function and natural selection heuristics to optimize the learning process. 18, 19

Intrinsic Motivation *Intrinsic Motivation* is a concept introduced by *Developmental Robotics* that aims to internally guide exploration by focusing on curiosity mechanisms to improve exploration in the learning process. 37, 45, 203

Inverse Reinforcement Learning *Inverse Reinforcement Learning* is an enhancement of *Reinforcement Learning* for which the reward function is extracted from observed behaviors and not design *a priori*. 18, 19, 20, 22, 37, 46

Learning by Demonstrations *Learning by Demonstrations* is a *Supervised Learning* technique that extracts learning policies from expert demonstrations. 19, 20, 22, 46

Intention Learning *Intention Learning* is a *Supervised Learning* technique that extracts the motivations behind the expert demonstrations to generate the learning policy. 20, 22, 37, 46

Lifelong Context Learning *Lifelong Context Learning* is an enrichment of *Context Learning* which enables a dynamic update of *local models* without storing all the *learning situations*. 80, 102, 105, 116, 121, 146, 169, 199, 230

Lifelong Learning *Lifelong Learning* aims to overcome the isolated learning paradigm and uses knowledge acquired for one task to solve related ones. It is defined by three key characteristics: continuous learning process, explicit knowledge retention and accumulation, and use of previously learned knowledge. 2, 16, 17, 19, 20, 22, 23, 24, 26, 27, 28, 29, 31, 32, 33, 37, 38, 40, 45, 80, 102, 115, 124, 125, 146, 151, 168, 195, 198, 229

Machine Learning *Machine Learning* or *Artificial Learning* is a field of *AI* that uses mathematical and statistical approaches to give computers the ability to learn from data. 2, 3, 15, 16, 18, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30, 37, 39, 41, 44, 45, 51, 67, 111, 165, 195, 223

Meta-Learning *Meta-Learning* aims at collecting meta-data about prior experiences, models and parameter, and to extract and transfer knowledge from it to guide the learning process of new tasks. 2, 27, 35, 38, 39, 40, 45, 46, 201, 203

Multi-Task Learning *Multi-Task Learning* focuses on learning simultaneously related tasks and using their relatedness to achieve better performances on each task. 2, 16, 20, 23, 24, 26, 27, 28, 32, 33, 38, 39, 44, 45, 46, 111, 112, 116, 196, 202

Offline Learning *Offline Learning* is a learning paradigm where all the training data is provided at once. New training data requires re-training the models on all the available data. 24, 28

Online Learning *Online Learning* is a learning paradigm where training data is sequentially provided to the learning mechanism. 2, 16, 19, 22, 23, 24, 28, 29, 31, 32, 33, 37, 38, 40, 41, 45, 46, 124, 125, 195, 198

Probabilistic Learning *Probabilistic Learning* approaches are machine learning techniques that enable to represent and manipulate uncertainty. 20, 21

Reinforcement Learning *Reinforcement Learning* is a learning approach that uses a reward function to enable learning by acting in an environment and receiving positive and negative feedback. 2, 18, 19, 20, 22, 23, 24, 28, 30, 32, 40, 44, 46, 81, 111, 113, 114, 116, 196, 202, 235

Semi-Supervised Learning *Semi-Supervised Learning* is a learning technique that uses labeled data (*exogenous learning situations*) and unlabeled data (*exploitation situations*). 17, 18, 22, 28, 46

Supervised Learning *Supervised Learning* is a learning technique that only uses labeled data (*exogenous learning situations*). 16, 17, 18, 19, 22, 27, 28, 29, 39, 44, 46, 51, 79, 235

Transfer Learning *Transfer Learning* is a research problem that focuses on using knowledge gained while solving one problem and applying it to a different but related problem. 2, 8, 16, 23, 24, 25, 26, 27, 32, 33, 37, 38, 39, 45, 46, 81, 111, 112, 116, 121, 165, 166, 169, 196, 198, 223, 227

Unsupervised Learning *Unsupervised Learning* are data aggregation and dimension reduction techniques that only use unlabeled data (*exploitation situations*). 16, 17, 22, 27, 28, 32, 45, 46

Thesis Objectives

Agnosticity The learning system is agnostic towards the task it has to learn. It can learn any task using the same approach without making preliminary assumptions. It has to focus on the task of learning itself. 3, 11, 16, 20, 22, 32, 33, 37, 40, 45, 46, 121, 123, 124, 168, 195, 197, 201

Any Data Amount Learning must be performed in every situation the learning system is facing, regardless of the amount of learning examples it has access to. 3, 12, 16, 22, 32, 33, 38, 45, 46, 121, 123, 125, 169, 191, 195, 201

Endogenous Feedback The mechanism learns from exogenous and endogenous experiences. Knowledge must also be generated through processes and interactions that are internal to the system through active learning or self-learning. 1, 3, 11, 12, 16, 17, 18, 19, 20, 22, 27, 29, 31, 32, 33, 37, 38, 40, 44, 45, 46, 121, 123, 168, 191, 195, 197, 201

Explainability The reason that brought the learning mechanism to choose a behavior instead of another one during an exploitation is explainable and observable. 3, 12, 16, 17, 18, 19, 20, 22, 27, 29, 31, 32, 33, 38, 40, 44, 45, 46, 121, 123, 126, 169, 195, 200, 201

Knowledge Generalization The learning mechanism recognizes if new experiences are similar to previous ones and it establishes connections between them permitting it to improve previously learned tasks and to learn faster the new ones. 3, 12, 16, 17, 18, 20, 22, 26, 29, 31, 32, 33, 37, 39, 40, 41, 44, 45, 46, 121, 123, 168, 169, 195, 199, 201

Lifelong Learning The system learns in an lifelong and incremental way to be able to improve its previous knowledge. If tasks are related, the learning system may share knowledge between these tasks to improve learning performance. 3, 11, 16, 17, 18, 22, 33, 44, 45, 46, 76, 121, 123, 168, 195, 197, 201

Online Learning The learning process can sequentially receive new data and be asked for decision making. The process of this new data does not interfere with the exploitation of previously learned tasks. 3, 12, 16, 22, 33, 44, 45, 46, 121, 123, 168, 195, 198, 201

Scalability Learning must be applicable to large-scale complex learning systems. Its complexity must be linearly dependent on the number of sensors or actuators. 3, 12, 16, 17, 18, 19, 20, 22, 26, 27, 29, 31, 32, 33, 38, 40, 44, 45, 46, 64, 121, 123, 169, 191, 195, 199, 201, 202

Self-Observation The system has the capability of self-observation to self-adapt. It is able to provide feedback on what it knows, how precisely it knows it and what it could know better. 3, 12, 16, 17, 18, 19, 20, 22, 27, 29, 31, 32, 33, 37, 38, 40, 44, 45, 46, 121, 123, 124, 168, 195, 198, 201

Learning Hypothesis

Agnosticity Hypothesis Learning is agnostic. It adapts to new needs and it is always open for new experiences as different as they can be. It does not need any prior knowledge to gather experience. 53, 55

Continuity Hypothesis The world, at its macroscopic scale is continuous. It is humans, through their perception of it, that make it discrete. Learning is seeking discontinuities in a continuum. 53, 54, 55, 68, 75

Curiosity Hypothesis A learner internally seeks novelty and explores new experiences. It looks for contradictions to develop better understanding of its environment. 53, 54

Generalization Hypothesis To learn is to generalize thanks to several experiences. Generalization is made across time and space as experiences can be related through history and context. 53, 54

Agents

Best Context Agent The *Best Context Agent* is the *Context Agent* for which the *local model* is considered the best among the collective during an execution cycle according to the *perceptions*, the *neighborhood/influences* and the *learning criticality* or the *exploitation criticality*. 43, 44, 75, 237, 76, 81, 82, 85, 86, 88, 93, 94, 95, 96, 97, 107, 109, 132, 177, 179, 181, 224, 246

Context Agent A *Context Agent* is an intelligent autonomous agent that locally represents a part of a learned global function $\mathcal{F}$ with a *local model* f_n^j, a spatial representation that are

the *validity ranges* $\mathcal{R}_n^j$ and a *confidence* c^j. 41, 42, 43, 44, 46, 47, 52, 53, 54, 55, 56, 57, 58, 59, 60, 62, 63, 64, 65, 68, 69, 70, 71, 72, 73, 74, 75, 76, 79, 80, 81, 82, 83, 84, 85, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 100, 102, 103, 104, 105, 107, 108, 109, 111, 112, 113, 115, 116, 118, 121, 124, 125, 127, 128, 129, 130, 131, 132, 133, 135, 136, 137, 140, 141, 142, 144, 145, 146, 147, 148, 149, 150, 152, 155, 156, 157, 159, 160, 161, 162, 163, 166, 167, 168, 169, 179, 196, 197, 198, 199, 201, 202, 203, 223, 224, 225, 227, 229, 237, 238, 239, 241, 242, 243, 245, 246, 247, 248

Good Context Agent *Good Context Agents* are *Context Agents* for which the *model prediction distance* is lesser than the *model error margin*. 85, 86, 88, 89, 90, 91, 94, 224, 225, 239

Head Agent The *Head Agent* communicates with the *Valid Context Agents* to retrieve their model propositions according to the *perceptions* and provide the final output *prediction vector*. 43, 79, 81, 82, 85, 86, 93, 116, 160, 161, 162, 163, 231, 238

Neighbor Context Agent *Neighbor Context Agents* are *Context Agents* that are activated by the *Percept Agents* using the *neighborhood* of the *perceptions* $\mathcal{P}_n$ and the influences of *Context Agents*. 56, 57, 59, 64, 65, 66, 68, 70, 71, 74, 75, 76, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 93, 94, 97, 100, 105, 109, 115, 116, 127, 135, 155, 160, 163, 169, 191, 196, 197, 199, 202, 224, 229, 233, 238, 239, 241, 242, 243, 244, 246

Percept Agent The *Percept Agents* are responsible for the activation of the *Valid Context Agents* and the *Neighbor Context Agents*. There is one *Percept Agent* $\mathcal{P}ct_i$ by perception p_i. 43, 79, 80, 81, 82, 83, 84, 93, 116, 160, 161, 162, 163, 199, 231, 238, 246

Prediction Neighbor Context Agent *Prediction Neighbor Context Agents* are *Neighbor Context Agents* which *local prediction vectors* are in *prediction neighborhoods*. 58, 68, 75, 76, 82, 109, 199, 223, 224, 239, 243, 246

Valid Context Agent *Valid Context Agents* are *Context Agents* that are activated by the *Percept Agents* using the *perceptions* $\mathcal{P}_n$. They provide *local prediction vectors* $\mathcal{O}_m^j$ to the *Head Agent*. 43, 44, 79, 81, 82, 83, 84, 85, 86, 87, 88, 89, 91, 93, 94, 107, 177, 224, 229, 238, 239, 246

Learning Incacuracies NCS

Complete Redundancy NCS This *NCS* is detected when two adjacent *Context Agents* with similar *local models* ($d_{sim}^f < t_{sim}^f$) can merge their common boundary without altering their other *validity ranges*. It is revolved by extending the *validity ranges* of one of the two *Context Agents* to the maximum of the both *Context Agents validity ranges*. 62, 63, 71, 72, 73, 86, 91, 93, 125, 131, 141, 142, 145, 152, 162, 224

Concurrency NCS This *NCS* is detected when two *Context Agents validity ranges* overlap and their *local models* are similar ($d_{sim}^f < t_{sim}^f$). Depending on the *learning strategy*, the two *Context Agents* negotiate which one should retract itself to suppress the overlapping area. 62, 63, 69, 73, 74, 86, 88, 93, 94, 97, 107, 109, 123, 131, 142, 145, 152, 162, 224, 248

Conflict NCS This *NCS* is detected when two *Context Agents validity ranges* overlap and their *local models* are different ($d^f_{sim} > t^f_{sim}$). Depending on the *learning strategy*, the two *Context Agents* negotiate which one should retract itself to suppress the overlapping area. 62, 69, 73, 74, 86, 87, 88, 93, 94, 97, 107, 109, 123, 130, 135, 142, 145, 147, 152, 156, 160, 167, 224, 248

Incompetence NCS This *NCS* is detected when *incompetent volumes* are found inside the *neighborhood* of the current *perceptions* at the end of an execution cycle c_l. It is resolved by creating new *Context Agents* using the *incompetent volumes* during future execution cycles. 64, 65, 70, 71, 73, 74, 90, 94, 107, 109, 123, 131, 135, 152, 160, 164, 177, 185, 224, 241, 244, 248

Model Ambiguity NCS This *NCS* is detected when the volume or quality of data is not adequate for a *Context Agent local model*. It is resolved by providing additional *learning situations* to the *local model*. 67, 73, 74, 102, 104, 107, 108, 123, 130, 131, 135, 136, 145, 152, 160, 161, 167, 177, 185, 224

Model Discontinuity NCS This *NCS* is detected when two *Context Agents* are *Neighbor Context Agents* but not *Prediction Neighbor Context Agents*. It is resolved with the *Range Ambiguity NCS*. 68, 70, 73, 224, 239

Partial Redundancy NCS This *NCS* is detected when two adjacent *Context Agents* with similar *local models* ($d^f_{sim} < t^f_{sim}$) and can restructure one of their *validity ranges* to maximize the volume of one of the *Context Agents*. It is resolved by extending the *validity range* the *Context Agent* that can maximize its volume.. 63, 64, 72, 73, 86, 91, 93, 125, 131, 141, 142, 145, 152, 162, 224

Range Ambiguity NCS This *NCS* is detected when two adjacent *Context Agents* have different *local models* ($d^f_{sim} > t^f_{sim}$) and a *Model Discontinuity NCS* is detected between them. It is resolved by generating *active learning situations* or *endogenous exploitation situations* close their frontier. 64, 70, 73, 74, 107, 109, 124, 131, 136, 137, 138, 141, 142, 147, 152, 160, 224, 239

Other NCS

Bad Prediction NCS This *NCS* is detected when a *Valid Context Agent* has a bad *local model* ($d^{f_j}_{\mathcal{L}_{n,m}} > m^f_{err}$). It is resolved by moving one of the *Valid Context Agent validity ranges* to exclude the current *perceptions*. 86, 87, 88, 93, 107, 224

Unproductivity NCS This *NCS* is detected when there aren't any *Valid Context Agents*. It is resolved by expanding the closest *Good Context Agent validity ranges* to aim at including the new *perceptions*. If the expansion failed or there is not any *Good Context Agent*, a new *Context Agent* is created. 86, 89, 90, 93, 94, 107, 109, 224, 225

Uselessness NCS This *NCS* is detected when one of a *Context Agent validity ranges* is lesser than the *minimum range distance*. It is resolved by subtracting such *Context Agents* from the learning mechanism. 86, 88, 93, 224

Learning and Exploitation Situations

active learning situation An *active learning situation* is a vector of *endogenous perceptions* $\mathcal{P}_n^{endo}$ combined with an *exogenous prediction vector* $\mathcal{O}_m^{exo}$: $\mathcal{L}_{n,m}^{act} = [\mathcal{P}_n^{endo}, \mathcal{O}_m^{exo}]$. 106, 107, 108, 110, 123, 130, 131, 167, 168, 177, 196, 197, 239, 243, 245, 246, 247

endogenous exploitation situation An *endogenous exploitation situation* is a vector of *endogenous perceptions* $\mathcal{P}_n^{endo}$: $\mathcal{E}_n^{endo} = [\mathcal{P}_n^{endo}]$. 86, 108, 109, 110, 123, 124, 135, 147, 151, 152, 156, 168, 196, 197, 239, 244, 245, 246, 247

endogenous learning situation An *endogenous learning situation* is a vector of or *endogenous perceptions* $\mathcal{P}_n^{endo}$ combined with an *endogenous prediction vector* $\mathcal{O}_m^{endo}$: $\mathcal{L}_{n,m}^{endo} = [\mathcal{P}_n^{endo}, \mathcal{O}_m^{endo}]$. 46, 51, 52, 75, 76, 77, 79, 80, 97, 104, 105, 109, 115, 116, 117, 123, 124, 135, 138, 147, 149, 151, 152, 158, 163, 168, 174, 175, 177, 180, 181, 182, 185, 187, 190, 191, 196, 197, 198, 199, 200, 224, 228, 234, 241, 244, 247

exogenous learning situation An *exogenous learning situation* is a vector of *exogenous perceptions* $\mathcal{P}_n^{exo}$ or *endogenous perceptions* $\mathcal{P}_n^{endo}$ combined with an *exogenous prediction vector* $\mathcal{O}_m^{exo}$: $\mathcal{L}_{n,m}^{pass} = [\mathcal{P}_n^{exo} / \mathcal{P}_n^{endo}, \mathcal{O}_m^{exo}]$. 43, 44, 51, 52, 79, 81, 86, 87, 91, 94, 95, 96, 97, 104, 105, 106, 107, 109, 117, 123, 135, 138, 142, 145, 147, 149, 150, 154, 158, 168, 182, 196, 197, 198, 200, 234, 236, 242, 244, 247

exploitation situation An *exploitation situation* is a vector of or *perceptions* $\mathcal{P}_n$: $\mathcal{E}_n = [\mathcal{P}_n]$. 43, 81, 93, 106, 108, 118, 123, 151, 152, 153, 154, 177, 226, 236, 245, 246

learning situation A *learning situation* is a vector of *perceptions* $\mathcal{P}_n$ combined with a *prediction vector* $\mathcal{O}_m$: $\mathcal{L}_{n,m} = [\mathcal{P}_n, \mathcal{O}_m]$. 1, 2, 41, 42, 43, 44, 51, 52, 56, 67, 69, 73, 75, 76, 77, 79, 85, 86, 91, 92, 94, 95, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108, 109, 110, 111, 112, 116, 117, 118, 121, 123, 124, 125, 130, 132, 135, 138, 140, 145, 146, 147, 150, 155, 156, 157, 160, 161, 162, 163, 167, 168, 169, 176, 177, 182, 183, 185, 186, 187, 189, 190, 191, 196, 197, 198, 199, 200, 201, 225, 227, 228, 235, 239, 241, 245, 247

passive learning situation A *passive learning situation* is a vector of *exogenous perceptions* $\mathcal{P}_n^{exo}$ combined with an *exogenous prediction vector* $\mathcal{O}_m^{exo}$: $\mathcal{L}_{n,m}^{pass} = [\mathcal{P}_n^{exo}, \mathcal{O}_m^{exo}]$. 43, 106, 108, 110, 123, 129, 130, 135, 155, 243, 244, 245, 247

Perceptions and Predictions

endogenous perceptions The *endogenous perceptions* is a vector of inputs p_i^{endo} provided by learning mechanism itself: $\mathcal{P}_n^{endo} = [p_1^{endo}, \ldots, p_i^{endo}, \ldots, p_n^{endo}] \in \mathbb{R}^n$. 52, 69, 70, 73, 74, 75, 79, 80, 81, 86, 88, 91, 106, 107, 108, 109, 240, 247

endogenous prediction vector An *endogenous prediction vector* is a *prediction vector* provided by the learning mechanism itself. 52, 75, 76, 94, 97, 108, 109, 240, 247

exogenous perceptions The *exogenous perceptions* is a vector of inputs p_i^{exo} provided by an entity external to the learning mechanism: $\mathcal{P}_n^{exo} = [p_1^{exo}, \ldots, p_i^{exo}, \ldots, p_n^{exo}] \in \mathbb{R}^n$. 52, 79, 80, 106, 108, 240, 247

exogenous prediction vector A *exogenous prediction vector* is a *prediction vector* provided by an entity external to the learning mechanism (labeled data in machine learning). 52, 57, 79, 81, 82, 85, 94, 95, 106, 107, 108, 240, 247

local prediction vector A *local prediction vector* is the *output prediction vector* $\mathcal{O}_m^j$ of the *Context Agent local model* f_n^j. 41, 42, 43, 81, 82, 85, 86, 93, 98, 238, 247

output prediction vector The *output prediction vector* is the vector of outputs ${o_k}'$ of the learning mechanism: $\mathcal{O}'_m = [o'_1, \ldots, o'_k, \ldots, o'_m] \in \mathbb{R}^m$. 81, 82, 85, 86, 93, 115, 241, 248

perceptions The *perceptions* is the vector of inputs p_i of the learning mechanism: $\mathcal{P}_n = [p_1, \ldots, p_i, \ldots, p_n] \in \mathbb{R}^n$. 41, 42, 43, 44, 52, 56, 57, 58, 61, 62, 64, 71, 74, 79, 81, 82, 83, 84, 85, 87, 88, 89, 92, 93, 94, 95, 96, 98, 99, 100, 103, 104, 107, 108, 109, 111, 112, 113, 114, 115, 121, 122, 123, 124, 125, 127, 128, 129, 133, 134, 135, 139, 140, 143, 144, 147, 149, 151, 156, 159, 160, 161, 162, 163, 165, 166, 168, 169, 177, 178, 179, 183, 184, 185, 191, 197, 199, 200, 202, 223, 224, 225, 227, 228, 237, 238, 239, 240, 242, 243, 247

prediction vector The *prediction vector* is the vector representing the labels o_k of *learning situations*: $\mathcal{O}_m = [o_1, \ldots, o_k, \ldots, o_m] \in \mathbb{R}^m$. 41, 42, 43, 52, 57, 93, 99, 102, 103, 104, 108, 123, 147, 169, 176, 178, 238, 240, 241, 248

Endogenous Context Learning Formalism

confidence The *confidence* $c^j \in \mathbb{Z}$ of *Context Agents* is an experience measure that enables them to evaluate themselves in relation to others. 41, 43, 44, 71, 72, 86, 87, 88, 90, 94, 95, 96, 97, 98, 168, 225, 229, 238, 243, 246

Cooperative Neighborhood Learning It is the implemented mechanism within *ELLSA* to generate *Endogenous Learning* using cooperation between *Neighbor Context Agents*. 2, 51, 52, 75, 76, 77, 80, 85, 105, 107, 115, 123, 136, 146, 150, 160, 169, 175, 177, 178, 179, 182, 185, 186, 187, 188, 189, 190, 191, 196, 198, 199, 200, 202, 224, 228, 229, 241

cooperative neighbors The *cooperative neighbors* are the *Neighbor Context Agents* that provided *endogenous learning situations* during learning and exploitation with the *Cooperative Neighborhood Learning* mechanism. 136, 140, 141, 144, 148, 152, 156, 160, 162, 166

exploitation criticality The *exploitation criticality* measure the relevance of a *Context Agent* during an exploitation cycle according to its spatial proximity (*range perceptions proximity distance*), its experience (*normalized confidence*) and its generalization (*validity ranges* volume). 2, 79, 93, 94, 95, 97, 101, 109, 116, 117, 132, 133, 177, 185, 196, 225, 237, 244, 245, 246

hidden function The *hidden function* is a global function that the learning mechanism *ELLSA* locally estimates with the *Context Agents*. 41, 55, 67, 75, 124, 128, 134, 139, 143, 147, 151, 155, 159, 165, 223, 224, 242, 246

incompetent volume The *incompetent volumes* are the empty area that are found in the *neighborhood* during the *Incompetence NCS*. 64, 65, 66, 70, 74, 90, 107, 109, 224, 239, 248

influence The *influence* is a spatial area around a *Context Agent* that is proportional to its *validity ranges*. Associated with the *neighborhood* it enables *Context Agents* to locally perceive themselves by activating the *Neighbor Context Agents*. 57, 58, 75, 82, 83, 84, 115, 118, 163, 196, 199, 201, 223, 237

influence radius The *influence radius* sets the influence length for a perception p_i and a *Context Agent* C_n^j. 57, 77, 84, 117, 248

learning criticality The *learning criticality* measure the relevance of a *Context Agent* during a learning cycle according to its prediction performance (*model prediction distance*), its experience (*normalized confidence*) and its generalization (*validity ranges* volume). 2, 79, 85, 86, 92, 95, 97, 101, 107, 116, 117, 132, 133, 176, 177, 185, 196, 225, 237, 244, 246

learning inaccuracy The *learning inaccuracies* are learning deficiencies characterized by the learning hypothesis section 5.1, they are categorized into *NCS* using the *AMAS* theory first. 2, 51, 52, 54, 59, 60, 61, 67, 69, 70, 73, 74, 77, 79, 80, 81, 83, 85, 86, 88, 91, 93, 106, 107, 108, 110, 115, 116, 118, 121, 123, 124, 128, 129, 130, 132, 133, 135, 136, 137, 140, 141, 144, 147, 148, 149, 150, 151, 152, 155, 156, 160, 162, 166, 167, 168, 196, 197, 198, 199, 202, 225, 229, 230, 243

local model A *local model* is a learning model that locally represents a part of the *hidden function*. Its spatial representation in the space of *perceptions* is given by the *validity ranges*. 41, 42, 43, 53, 54, 55, 59, 75, 76, 81, 82, 85, 86, 88, 90, 91, 92, 93, 94, 95, 96, 97, 98, 100, 102, 103, 104, 107, 108, 109, 111, 112, 115, 116, 117, 124, 125, 127, 128, 130, 131, 132, 134, 135, 139, 142, 143, 145, 146, 151, 155, 156, 162, 167, 168, 169, 176, 196, 198, 199, 200, 223, 224, 235, 237, 238, 239, 241, 242, 244, 245, 247, 248

maximum range radius The *maximum range radiuses* is the maximal *validity range* radius length below which the expansion is allowed. 89, 92, 117, 245, 248

minimum range distance The *minimum range distance* defines the minimal precision distance for the *validity ranges*. 61, 62, 64, 77, 88, 117, 239, 245, 246

model prediction distance The *model prediction distance* measure the affinity of a *local model* with an *exogenous learning situation*. 85, 87, 88, 92, 95, 98, 99, 101, 117, 198, 238, 242, 244, 245, 246

model similarity distance The *model similarity distance* measure the similarity between two *local models*. 61, 62, 63, 64, 70, 77, 88, 97, 98, 100, 101, 117, 245, 246

neighborhood The *neighborhood* is a spatial area around the current *perceptions* of a cycle $\mathcal{P}_n^{c_l}$. Associated with the influence of *Context Agents*, it enables *Context Agents* to locally perceive themselves by activating the *Neighbor Context Agents*. 2, 51, 52, 56, 57, 58, 64, 65, 66, 70, 74, 75, 76, 77, 79, 80, 81, 82, 83, 84, 242, 88, 89, 90, 109, 115, 118, 127, 131, 154, 163, 169, 191, 196, 197, 198, 199, 201, 202, 223, 224, 225, 229, 237, 238, 239, 241, 242, 243, 244, 245

neighborhood radius The *neighborhood radius* sets the *neighborhood* length for a perception p_i. 56, 57, 77, 83, 84, 117, 248

normalized confidence The *normalized confidence* is a normalization of *confidence* between 0 and 1 to enable it to be comparable with other positive metrics. 95, 96, 101, 117, 198, 241, 242, 244, 246

prediction neighborhood The *prediction neighborhood* adds the prediction dimensions to the *neighborhood* to measure if *Neighbor Context Agents* are *Prediction Neighbor Context Agents*. 57, 76, 82, 85, 109, 229, 238, 243

prediction neighborhood radius The *prediction neighborhood radius* sets the *prediction neighborhood* length for a prediction o_k. 56, 57, 58, 68, 77, 117, 248

range creation radius The *range creation radius* is the default *validity range* radius length on the perceptions p_i during the creation of a new *Context Agent*. 56, 57, 61, 77, 89, 90, 117, 248

range perceptions proximity distance The *range perceptions proximity distance* measure the proximity between the *validity ranges* in the space of *perceptions*. 89, 92, 95, 96, 97, 100, 101, 117, 241, 245, 246

range similarity distance The *range similarity distance* defines the maximal distance between two *validity ranges* borders to be similar. 61, 63, 64, 77, 117, 245, 246

validity range The *validity ranges* are the spatial representation of *Context Agents* in the space of *perceptions*. There is one *validity range* by perception p_i. 41, 42, 43, 47, 53, 54, 55, 56, 57, 59, 60, 61, 62, 63, 64, 65, 66, 69, 70, 71, 72, 73, 74, 75, 76, 81, 82, 83, 87, 88, 89, 90, 91, 92, 94, 95, 96, 97, 98, 100, 101, 102, 103, 105, 107, 111, 112, 115, 117, 118, 127, 128, 129, 130, 131, 133, 135, 149, 150, 151, 159, 165, 167, 168, 178, 179, 199, 200, 202, 223, 224, 227, 228, 229, 233, 238, 239, 241, 242, 243, 244, 245, 248

Learning Strategies

Active Cooperative Learning Strategy The *Active Cooperative Learning Strategy* is a learning scenario where *passive learning situations* and *active learning situations* are used with *neighborhood* mechanisms and *CNL*. 146, 147, 148, 149, 150, 226

Active Learning Strategy The *Active Learning Strategy* is a learning scenario where *passive learning situations* and *active learning situations* are used with *neighborhood* mechanisms but without *CNL*. 3, 69, 79, 81, 86, 106, 107, 108, 110, 116, 121, 123, 128, 130, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 159, 161, 162, 163, 165, 166, 168, 169, 196, 197, 201, 225, 226, 227

Naive Learning Strategy The *Naive Learning Strategy* is a learning scenario where only *passive learning situations* are used without any *neighborhood* mechanisms. The neighborhood mechanisms include the *learning inaccuracies NCS*, the creations of *Context Agents* with *Neighbor Context Agents* and *CNL*. 134, 135, 138, 139, 140, 141, 142, 143, 144, 145, 155, 156, 157, 227

Self-Learning Strategy The *Self-Learning Strategy* is a learning scenario where *passive learning situations* and *endogenous exploitation situations* are used with *neighborhood* mechanisms and *CNL*. 3, 69, 79, 81, 86, 106, 107, 108, 109, 110, 116, 121, 123, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 163, 164, 168, 169, 177, 185, 196, 197, 198, 201, 225, 226, 227, 244

Learning Parameters

accuracy learning weight The *accuracy learning weight* is a designer parameter to set the weight of the *model prediction distance* during the calculation of the *learning criticality*. 95, 101, 117, 122, 178, 185, 248

bootstrap cycles number The *bootstrap cycles number* is a user parameter to set the number of learning cycles during which no *NCS* are detected and resolves. 84, 86, 93, 117, 122, 246

creation neighbors number The *creation neighbors number* is a designer parameter to set the number of necessary *Neighbor Context Agents* to detect an *Incompetence NCS* during the *Self-Learning Strategy*. 94, 109, 117, 122, 247

discontinuity detection probability The *discontinuity detection probability* is a user parameter to set the importance of discontinuity detection during learning. 70, 77, 117, 122, 134, 137, 138, 225, 246

endogenous learning weight The *endogenous learning weight* is a user parameter to set the weight of *endogenous learning situations* on the *local models*. 76, 77, 105, 117, 122, 153, 157, 226, 227, 249

exogenous learning weight The *exogenous learning weight* is a user parameter to set the weight of *exogenous learning situations* on the *local models*. 105, 117, 122, 157, 227, 249

experience exploitation weight The *experience exploitation weight* is a designer parameter to set the weight of the *normalized confidence* during the calculation of the *exploitation criticality*. 96, 101, 117, 122, 178, 185, 249

experience learning weight The *experience learning weight* is a designer parameter to set the weight of the *normalized confidence* during the calculation of the *learning criticality*. 95, 101, 117, 122, 178, 185, 249

generalization exploitation weight The *generalization exploitation weight* is a designer parameter to set the weight of the *validity ranges* volumes during the calculation of the *exploitation criticality*. 96, 101, 117, 122, 178, 185, 249

generalization learning weight The *generalization learning weight* is a designer parameter to set the weight of the *validity ranges* volumes during the calculation of the *learning criticality*. 95, 101, 117, 122, 178, 185, 249

generalization score Generalization Score. 125, 129, 130, 131, 132, 133, 135, 136, 138, 140, 141, 142, 144, 145, 147, 148, 149, 150, 152, 156, 166, 167, 168, 248

influence radius coefficient The *nfluence radius coefficient* is a designer parameter to set the reach of the *Context Agents* influences. 57, 77, 84, 117, 122, 153, 157, 158, 226, 227, 246

maximum range radius coefficient The *maximum range radius coefficient* is a designer parameter to set the *maximum range radius*. 89, 92, 117, 122, 246

minimum range coefficient The *minimum range coefficient* is a designer parameter to set the *minimum range distance*. 61, 77, 117, 122, 246

model error margin The *model error margin* is a user parameter to set the *model prediction distance* below which *Context Agents local models* are considered good. 85, 87, 88, 92, 117, 122, 124, 134, 136, 137, 138, 140, 144, 147, 151, 156, 159, 166, 178, 185, 198, 225, 238, 247

model similarity threshold The *model similarity threshold* is a designer parameter to set the *model similarity distance* below which two *local models* are considered similar. 100, 101, 117, 122, 139, 141, 142, 226, 248

neighborhood radius coefficient The *neighborhood radius coefficient* is a designer parameter to set the size of the *neighborhood* area. 56, 57, 77, 84, 117, 122, 153, 157, 158, 179, 180, 181, 182, 226, 227, 228, 246

number of exploitation situations It is the number of *exploitation situations* during the exploitation phase of a *learning episode*. These situations are used to measure the prediction performance. 122, 123, 129, 135, 140, 144, 147, 151, 156, 159, 166, 178, 185, 247

number of learning episodes It is the number of *learning episodes* during an experiment. A *learning episode* is a learning phase followed by an exploitation phase. 122, 129, 135, 140, 144, 147, 151, 156, 159, 166, 178, 185, 247

number of learning situations It is the number of the accumulated *passive learning situations*, *active learning situations* or *endogenous exploitation situations* during the learning phase of a *learning episode*. 122, 125, 129, 135, 140, 144, 147, 151, 156, 159, 166, 178, 185, 247

perceptions generation coefficient The *perceptions generation coefficient* is a designer parameter to set the *validity range* ratio for generating artificial *learning situations*. 103, 105, 117, 122, 149, 150, 226, 246

proximity exploitation weight The *proximity exploitation weight* is a designer parameter to set the weight of the *range perceptions proximity distance* during the calculation of the *exploitation criticality*. 96, 101, 117, 122, 178, 185, 249

range similarity coefficient The *range similarity coefficient* is a designer parameter to set the *range similarity distance*. 61, 77, 117, 122, 246

validity ranges precision The *validity ranges precision* is a user parameter to set the precision of *Context Agents validity ranges*. 56, 57, 61, 77, 84, 117, 122, 125, 128, 131, 132, 133, 135, 140, 144, 147, 148, 149, 150, 151, 153, 154, 156, 157, 158, 159, 161, 163, 166, 178, 179, 180, 181, 185, 225, 226, 227, 248

Notations

$\mathcal{C}_n$ All *Context Agents*. 82

$^{\mathcal{N}}\mathcal{C}_n$ All *Neighbor Context Agents*. 82, 83, 86, 93

$^{\mathcal{PN}}\mathcal{C}_n$ All *Prediction Neighbor Context Agents*. 82

$^{\mathcal{V}}\mathcal{C}_n$ All *Valid Context Agents*. 82, 83

$^{\mathcal{B}}\mathcal{C}_n$ *Best Context Agent*. 82, 85, 86, 93

$\mathcal{C}_n^j$ *Context Agent* j. 41, 42, 56, 57, 81, 83, 97, 98, 242, 246, 247, 248

$^{\mathcal{N}}\mathcal{C}_n^j$ *Neighbor Context Agent* j. 94

$\mathcal{P}ct_i$ *Percept Agent* i. 83, 238

$^{\mathcal{V}}\mathcal{C}_n^j$ *Valid Context Agent* j. 86

$\alpha^{\mathcal{I}}$ *Influence radius coefficient*. 57, 77, 84, 117, 122, 153, 157, 226, 227

$\alpha^{\mathcal{R}_{max}}$ *Maximum range radius coefficient*. 89, 92, 117, 122

$\alpha^{\mathcal{R}_{min}}$ *Minimum range coefficient*. 61, 77, 117, 122

$\alpha^{\mathcal{N}}$ *Neighborhood radius coefficient*. 56, 57, 77, 84, 117, 122, 153, 157, 226, 227

$\alpha^{\mathcal{P}_{gen}}$ *Perceptions generation coefficient*. 246, 103, 105, 117, 122, 149, 150, 226

$\alpha^{\mathcal{R}_{sim}}$ *Range similarity coefficient*. 61, 77, 117, 122

c^j *Confidence* of the *Context Agent* $\mathcal{C}_n^j$. 41, 43, 86, 246, 97, 238, 241

$c_{0,1}^j$ *Normalized confidence* of the *Context Agent* $\mathcal{C}_n^j$. 95, 96, 97, 101, 117

$Crit^{expl}$ *Exploitation criticality*. 93, 94, 95, 101, 117

$Crit^{lrn}$ *Learning criticality*. 85, 86, 92, 95, 101, 117

c_{boot} *Bootstrap cycles number*. 84, 86, 93, 117, 122

c_l Execution *cycle* l of the learning mechanism. 43, 81, 82, 83, 86, 93, 102, 231, 239, 247, 248

pb^{disc} *Discontinuity detection probability*. 70, 77, 117, 122, 137, 225

$d_i^{\mathcal{R}_{min}}$ *Minimum range distance*. 61, 62, 64, 70, 77, 246, 88, 117

$d_{\mathcal{L}_{n,m}}^{f_j}$ *Model prediction distance*. 85, 86, 87, 88, 91, 92, 95, 98, 117, 239

d_{sim}^{f} *Model similarity distance*. 61, 62, 63, 64, 70, 77, 88, 98, 100, 117, 238, 239

$d_{\mathcal{P}_n}^{\mathcal{R}_n^j}$ *Range perceptions proximity distance*. 92, 95, 96, 100, 101, 117

$d_i^{\mathcal{R}_{sim}}$ *Range similarity distance*. 61, 63, 64, 77, 117

$\mathcal{E}_n^{endo}$ *Endogenous exploitation situation*. 108, 123, 240

$\mathcal{E}_n$ *Exploitation situation*. 43, 81, 93, 240

$\mathcal{F}$ *Hidden function*. 41, 55, 75, 128, 134, 139, 143, 223, 224, 237

$\mathcal{L}_{n,m}^{act}$ *Active learning situation*. 106, 123, 240

$\mathcal{L}_{n,m}^{endo}$ *Endogenous learning situation*. 52, 75, 97, 123, 240

$\mathcal{L}_{n,m}^{exo}$ *Exogenous learning situation*. 52, 79, 81, 91, 95, 97, 123

$\mathcal{L}_{n,m}$ *Learning situation*. 41, 52, 85, 94, 240

$\mathcal{L}_{n,m}^{c_l}$ *Learning situation* of the cycle c_l. 82, 85, 86

$\mathcal{L}_{n,1}$ *Learning situation* with linear regression *local model*. 42, 43, 99

$\mathcal{L}_{n,1}^{c_l}$ *Learning situation* with linear regression *local model* for the cycle c_l. 43

$\mathcal{L}_{n,m}^{pass}$ *Passive learning situation*. 106, 108, 123, 240

f_n^j *Local model* of the *Context Agent* $\mathcal{C}_n^j$. 41, 42, 82, 85, 86, 98, 99, 237, 241

m_{err}^f *Model error margin*. 85, 86, 87, 88, 91, 92, 117, 122, 137, 140, 144, 147, 151, 156, 159, 166, 178, 185, 225, 239

$n^{creation}$ *Creation neighbors number*. 94, 117, 122

n_{Ctxt} Number of *Context Agents*. 125, 130, 132, 136, 137, 140, 141, 144, 148, 149, 150, 152, 156, 160, 162, 166

$\mathcal{E}ps_{\#}$ *Number of learning episodes*. 122, 129, 135, 140, 144, 147, 151, 156, 159, 166, 178, 185

$\mathcal{E}_{\#}^{endo}$ Number of *endogenous exploitation situations*. 123

$\mathcal{E}_{\#}$ *Number of exploitation situations*. 122, 129, 135, 140, 144, 147, 151, 156, 159, 166, 178, 185

$\mathcal{L}_{\#}^{act}$ Number of *active learning situations*. 123, 130, 132

$\mathcal{L}_{\#}^{endo}$ Number of *endogenous learning situations*. 123, 175

$\mathcal{L}_{\#}$ *Number of learning situations*. 122, 129, 135, 140, 144, 147, 151, 156, 157, 159, 166, 178, 185

$\mathcal{L}_{\#}^{pass}$ Number of *passive learning situations*. 123, 130, 132

$\mathcal{P}_n^{endo}$ *Endogenous perceptions*. 52, 69, 74, 75, 247, 86, 88, 91, 106, 108, 123, 240

$\mathcal{P}_n^{exo}$ *Exogenous perceptions*. 52, 79, 106, 108, 123, 240

$\mathcal{P}_n$ *Perceptions*. 41, 42, 43, 52, 56, 81, 82, 85, 93, 98, 99, 238, 240, 241, 247

$\mathcal{P}_n^{c_l}$ *Perceptions* of cycle c_l. 43, 81, 82, 85, 86, 89, 93, 94, 242

p_i^{max} Maximum experienced value on the perception p_i. 56, 83, 84, 125

p_i^{min} Minimum experienced value on the perception p_i. 56, 83, 84, 125

p_i Single perception i of the vector of *perceptions* $\mathcal{P}_n$. 41, 56, 57, 61, 64, 247, 65, 83, 100, 247, 238, 241, 242, 243, 247, 248

$\mathcal{O}_m^{endo}$ *Endogenous prediction vector*. 52, 75, 76, 94, 123, 240

$\mathcal{O}_m^{exo}$ *Exogenous prediction vector*. 52, 57, 79, 81, 85, 106, 108, 123, 240

$\mathcal{O}_m^{j,last}$ Last *local prediction vector* of a *Context Agents* $\mathcal{C}_n^j$. 57, 85, 86, 93

$\mathcal{O}_m^j$ *Local prediction vector* of a *Context Agents* $\mathcal{C}_n^j$. 41, 82, 238, 241

$\mathcal{O}_1^j$ *Local prediction vector* of a *Context Agents* $\mathcal{C}_n^j$ with linear regression *local model*. 42, 98, 99

$\mathcal{O}_1^{j,c_l}$ *Local prediction vector* of a *Context Agents* $\mathcal{C}_n^j$ with linear regression *local model* for the cycle c_l. 43

$\mathcal{O}'_m$ *Output prediction vector*. 41, 93, 241

$\mathcal{O}'_1$ *Output prediction vector* with linear regression *local models*. 42, 99

$\mathcal{O}_1^{c_l\prime}$ *Output prediction vector* with linear regression *local models* for the cycle c_l. 44

$\mathcal{O}_{Err}$ Prediction error. 123, 130, 132, 133, 136, 137, 140, 141, 144, 148, 149, 150, 152, 153, 156, 157, 160, 162, 166, 226, 227

$\mathcal{O}_m$ *Prediction vector*. 41, 52, 240, 241, 248

$\mathcal{O}_1$ *Prediction vector* with linear regression *local model*. 42, 43, 99

$\mathcal{O}_1^{c_l}$ *Prediction vector* with linear regression *local model* for the cycle c_l. 43

$\mathcal{O}_m^{max}$ Vector of maximum experienced prediction values. 85, 86, 93

$\mathcal{O}_m^{min}$ Vector of minimum experienced prediction values. 85, 86, 93

o_k^{max} Maximum experienced value on the prediction o_k. 56, 57, 85

o_k^{min} Minimum experienced value on the prediction o_k. 56, 57, 85

o_k Single prediction k of the *prediction vector* $\mathcal{O}_m$. 57, 241, 243, 248

$r_i^{creation}$ *Range creation radius*. 56, 57, 61, 77, 89, 90, 117

$r_{j,i}^{\mathcal{I}}$ *Influence radius* on the perception p_i for the *Context Agent* $\mathcal{C}_n^j$. 57, 77, 83, 84, 117

r_i^{max} *Maximum range radius*. 89, 92, 117

$r_i^{\mathcal{N}}$ *Neighborhood radius* on the perception p_i. 56, 57, 77, 83, 84, 117

$r_{o_k}^{\mathcal{N}}$ *Prediction neighborhood radius* on the prediction o_k. 56, 57, 68, 77, 117

r_i^j *Validity range* of the *Context Agent* $\mathcal{C}_n^j$ on the perception p_i. 41, 56, 57, 248, 83, 87, 88, 96, 100

$\mathcal{R}_n^j$ *Validity ranges* of the *Context Agent* $\mathcal{C}_n^j$. 41, 82, 238

$p^{\mathcal{R}}$ *Validity ranges precision*. 56, 57, 61, 77, 84, 92, 117, 122, 132, 135, 140, 144, 147, 149, 151, 153, 156, 157, 159, 160, 161, 162, 163, 166, 178, 179, 180, 181, 185, 225, 226, 227, 228

$\mathcal{G}^{scr}$ *Generalization score*. 125, 130, 132, 133, 136, 140, 141, 144, 148, 149, 150, 152, 156, 166

t_{sim}^f *Model similarity threshold*. 100, 101, 117, 122, 141, 226, 238, 239

$\mathcal{V}_{Conc}$ Volume of *Concurrency NCS*. 124, 125, 130, 132, 133, 136, 137, 140, 141, 144, 148, 149, 150, 152, 156, 166

$\mathcal{V}_{Cflt}$ Volume of *Conflict NCS*. 124, 125, 130, 132, 133, 136, 137, 140, 141, 144, 148, 149, 150, 152, 156, 166

$\mathcal{V}_n^j$ Volume of a *Context Agent* $\mathcal{C}_n^j$. 95, 96, 101, 117

$\mathcal{V}_{Ctxt}$ Volume explored by all *Context Agents*. 124, 125, 160, 162

$\mathcal{V}_{Inc}$ Volume of *Incompetence NCS*. 125, 130, 132, 133, 136, 137, 140, 141, 144, 148, 149, 150, 152, 156, 166

$\mathcal{V}_n^{inc}$ *Incompetent volume*. 64, 65

$w_{f_n}^{lrn}$ *Accuracy learning weight*. 95, 101, 117, 122, 132, 176, 178, 185

w_{lrn}^{endo} *Endogenous learning weight*. 76, 77, 105, 117, 122, 153, 157, 226, 227

w_{lrn}^{exo} *Exogenous learning weight*. 105, 117, 122, 157

$w_{c_{0,1}}^{expl}$ *Experience exploitation weight*. 95, 96, 101, 117, 122, 132, 177, 178, 185

$w_{c_{0,1}}^{lrn}$ *Experience learning weight*. 95, 101, 117, 122, 132, 176, 178, 185

$w_{\mathcal{R}_n}^{expl}$ *Generalization exploitation weight*. 95, 96, 101, 117, 122, 132, 177, 178, 185

$w_{\mathcal{R}_n}^{lrn}$ *Generalization learning weight*. 95, 101, 117, 122, 132, 176, 178, 185

$w_{\mathcal{P}_n}^{expl}$ *Proximity exploitation weight*. 95, 96, 101, 117, 122, 132, 177, 178, 185