A Local Active Learning Strategy by Cooperative Multi-Agent Systems

Bruno Dato, Marie-Pierre Gleizes, Frédéric Migeon

A Local Active Learning Strategy by Cooperative Multi-Agent Systems

Bruno Dato, Marie-Pierre Gleizes and Frédéric Migeon
Université Toulouse III Paul Sabatier, Toulouse, France
{*bruno.dato, marie-pierre.gleizes, frederic.migeon*}*@irit.fr*

Keywords: Multi-Agent System Learning, Local Active Learning, Context Learning

Abstract: In this paper, we place ourselves in the context learning approach and we aim to show that adaptive multi-agent systems are a relevant solution to its enhancement with local active learning strategy. We use a local learning approach inspired by constructivism: context learning by adaptive multi-agent systems. We seek to introduce active learning requests as a mean of internally improving the learning process by detecting and resolving imprecisions between the learnt knowledge. We propose a strategy of local active learning for resolving learning inaccuracies. In this article, we evaluate the performance of local active learning. We show that the addition of active learning requests facilitated by self-observation accelerates and generalizes learning, intelligently selects learning data, and increases performance on prediction errors.

1 Introduction

Every learning system has advantages and drawbacks, parts of the environment where it is accurate, where it is efficient, and parts where it is not, as Wolpert stated in his "No Free Lunch Theorem" (Wolpert and Macready, 1997). Therefore, we focus on meta-learning techniques that could benefit from the composition of specific learning systems in distinct parts of the search space as well as in their borders. Dealing with sociotechnical environments of ambient systems, an intelligent application to be designed must handle properties such as openness, heterogeneity, unpredictability and dynamics (Russell and Norvig, 2016; Abbas et al., 2018). However, as sociotechnical environments are impossible to be specified *a priori*, these systems must be able to adapt to their environment by learning from few experiences. They must learn throughout their lives without having any intrinsic knowledge of the tasks they will have to solve. Such systems are called agnostic systems (Kearns et al., 1994). To design them, it is essential to generate knowledge through processes and actions that are purely internal to the system interacting with its environment.

The proposed system enables to generate active learning requests. To deal with the previous hypotheses, we try to be agnostic regarding the learning techniques by embodying them in agents that cooperate in order to produce more knowledge. Inspired by constructivism of Piaget (Piaget, 1976) and developmental learning methods applied in robotics, we define a multi-agent system that uses a local active learning strategy to enhance the results. Each agent, each learnt model, is used locally in a small part of the search space, where it is accurate. They share local information that is internal to the system to generate active requests.

This paper presents some steps of this research goal. In the next section, we provide to the reader the background information on constructivism, multi-agent systems and how the context-based meta-learning pattern uses data to learn from the environment. This work is intended to scale up with the number of dimensions of the search space, but to be simpler, we present it as a 2D problem. Moreover, we use a default regression model to implement the embodied model in each agent but our goal remains to be independent of the type of the learning technique.

We present in the following section the existing Context Learning which is the base of our work where each agent represents a context linked to an action or answer. Then, we detail the enhancement of the context learning with the Active Context Learning which enables the system to find in itself and alone which specific learning situations to request in order to improve its learning. In the experimentations section, we describe the metrics we use and how the experiments were conducted. Finally, a discussion, related work and conclusion are presented.

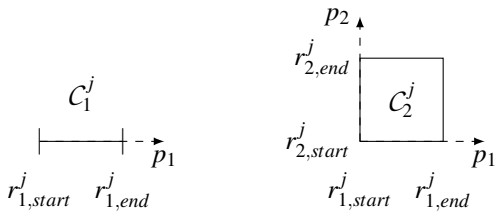
2 Background

Constructivism by schemas implemented with Multi-Agent Systems is the foundation of the Context Learning studied in this paper.

Constructivism. The work of Piaget on child development (Piaget, 1976) has inspired Constructivist Learning. It stated that knowledge construction cannot be separated from the subject's environment and the actions made on it. Called *schema*, this unit of knowledge aggregates several perceptions and several actions (Guerin, 2011). Robotic applications made use of the work of Drescher (Drescher, 1991) and Holmes (Holmes et al., 2005) to obtain self-organizing maps (SOM) (Chaput, 2004; Provost et al., 2006). Recently, it also served as a foundation on agentified model-based learning (Perotto, 2013).

Multi-Agent Systems. In order to deal with the complexity of real world situations like in cyber-physical systems (non-linearity, dynamics, distributed information, noisy data and unpredictability), it is important to build software systems which embody as many states as the real systems do (Ashby, 1956). For this purpose, Multi-Agent Systems (Ferber, 1999), and in particular Adaptive Multi-Agent Systems (AMAS) (Georgé et al., 2011) provide adaptive properties to deal with these unexpected situations, which is appropriate for learning systems (Mazac et al., 2014; Guériau et al., 2016). AMAS are artificial complex systems with fine granularity agents enabling the emergence expected global properties. Domains like the control of biological processes, the optimal control of motors or robotics learning (Boes et al., 2015) have exhibited such properties.

Context Learning. The AMOEBA system (Agnostic MOdEl Builder by self-Adaptation) (Nigon et al., 2016) is based on Context Learning. To implement Piaget's schema in a n-dimension space, *Context Agents* are used to approximate local models. Models are based on any machine learning technique (neural networks, linear regression, SVMs, decision trees, k-means...). The global model can be described as a hidden function $\mathcal{F}(\mathcal{P}_n) = \mathcal{F}(p_1,\ldots,p_i,\ldots,p_n) = O_m$ where $O_m \in \mathbb{R}^m$ is a labeled prediction vector. **Perceptions** denote the vector of inputs $\mathcal{P}_n = [p_1,\ldots,p_n] \in \mathbb{R}^n$ and a ***learning situation*** is defined by the vector $\mathcal{L}_{n,m} = [\mathcal{P}_n, O_m]$. The goal of the Context Learning System is to give as output a *prediction vector* $O'_m \in \mathbb{R}^m$ where $O'_m = O_m$. Each *Context Agent* $\mathcal{C}_n^j$ is in charge of managing the learning model for the j^{th} pavement in dimension n which represents a part of the *schema*. The global function $\mathcal{F}$ is approximated by a local function $f_n^j(p_1,\ldots,p_i,\ldots,p_n) = o_m^j$ with $o_m^j \in \mathbb{R}^m$. The n-dimension space is reduced to a n-parallelotope for every *Context Agent* in the form of *validity ranges* $\mathcal{R}_a^j = [r_1^j,\ldots,r_i^j,\ldots,r_n^j]$ with $r_i^j = [r_{i,start}^j, r_{i,end}^j]$ for each perception p_i (Fig. 1). This is completed with a *confidence* $c^j \in \mathbb{Z}$ that enable agents to order themselves. A *Context Agent* is then defined by its validity ranges, its model and its confidence $\mathcal{C}_n^j = \{\mathcal{R}_a^j, f_n^j, c^j\}$.



(a) Dimension 1 (b) Dimension 2

Figure 1: Parallelotopes validity ranges

3 Context Learning Strategies

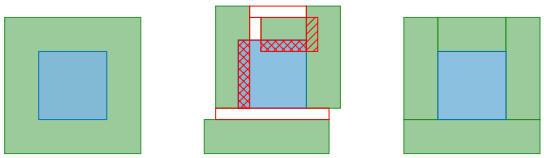
The implementation of AMOEBA is based on several simple rules and a default learning model (linear regression) that we briefly present after. In the following, a new local active Context Learning strategy is proposed which uses negotiation between agents neighboring the perceptions.

3.1 Former Context Learning Strategy

To define a good prediction, an error margin and an inaccuracy margin are used. The error margin defines a threshold above which a prediction is said bad. In the other case, the inaccuracy margin is used to address the precision needs, it is always less than the error margin. These two margins are calibrated by the user. In AMOEBA, agents are reactive. Their behavior is organized in execution cycles, which may be learning cycles or exploitation cycles.

Learning. During a learning cycle, each *Context Agent* receives the *learning situation* including $\mathcal{P}_n$. Negotiation between them, according to the validity ranges and confidence, results in the identification of the *Best Context Agent* for the current execution cycle. Though cooperative analysis, they individually realize different adaptations using the labeled prediction (Nigon et al., 2016).

Exploitation. During an exploitation cycle, only the *perceptions* $\mathcal{P}_n$ are submitted to the system. The *Best Context Agent* is designated as the one with the higher confidence and its prediction serves as the out-



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

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

put of the system. If no *Valid Context Agent* can be identified, *Closest Context Agent* to the *perceptions* is designated as the *Best Context Agent*.

Implemented Model. In this study, each *Context Agent* $\mathcal{C}_n^j$ has a local linear regression model f_n^j. In this case, the prediction vectors and the oracle predictions are a real values O_1' and O_1. A *learning situation* is then $\mathcal{L}_{n,1} = [\mathcal{P}_n, O_1]$.

Shortcomings. On a simple problem, with two models distributed in a 2D-space like represented in Fig. 2a with different colors, the presented learning rules do not allow the system to converge to an ideal representation (Fig. 2c), i.e. with the minimum of *Context Agents*, without any overlap or gaps. These rules do not protect against imprecise exploration (Fig. 2b). Incompetencies of the agents are not detected. Conflicts and concurrencies between the agents are not taken into account. All resolutions are made independently of the *Context Agents* neighbors. Exchanging information on the model between neighbors agents could improve the accuracy.

3.2 A New Local Active Context Learning Strategy

The Local Active Context Learning is an enhancement of Context Learning using **internal information** to requested specific *learning situations*. This mechanism is based on a collaboration inside the neighborhood of the *Context Agents*. In this section, we present the establishment of *Context Agent*'s neighborhood, the detection and the resolution of *learning inaccuracies*. The resolutions are operational regardless of the number of dimensions. However, to simplify the illustrations, only two dimensions are represented in the figures.

3.2.1 Context Agent Neighborhood

A *Context Agent* is considered as a neighbor of the *perceptions* if its validity ranges intersect a neighborhood area surrounding the current *perceptions*. The size of this area results from the *precision range* prc^{Range}, a parameter chosen by the user of the learning mechanism. For a perception p_i, there is a default radius creation for a *Context Agent* $r_i^{creation} = (p_i^{max} - p_i^{min}).prc^{Range}$ with p_i^{max} and p_i^{min} the maximum and minimum experienced values during the learning phase on the perception p_i. From this, it results the learning precision distance $d_i^{lpd} = 0.5 * r_i^{creation}$ and an approximation error distance $d_i^{aed} = 0.25.r_i^{creation}$. The neighborhood radius is $r_i^{\mathcal{N}} = 2.r_i^{creation}$.

3.2.2 Learning Inaccuracies Detection

With the previous definition of the neighborhood, six types of *learning inaccuracies* are defined. From the point of view of Adaptive Multi-Agent Systems, they are called Non Cooperative Situations (NCS):

Conflict NCS Detection (Fig. 3a). It is an ***overlap*** between *Context Agents* that have different models. For a learning problem with n dimensions, an overlap between two *Contexts Agents* is defined by n non-empty intersections of validity ranges. An overlap occurs if the intersections lengths between agents validity ranges are greater than d_i^{aed}. To differentiate models, a metric has to be defined by the user depending on the implemented learning models.

Concurrency NCS Detection (Fig. 3b). It is an ***overlap*** between two *Context Agents* that have similar models and give similar predictions. The geometry detection is the same as the Conflict NCS.

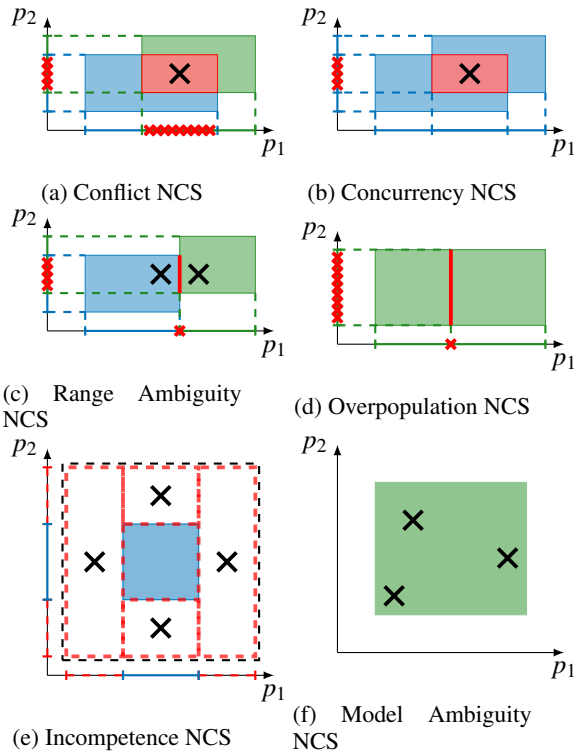
Range Ambiguity NCS Detection (Fig. 3c). It is a difference of model between two ***adjacent*** *Context Agents*. An ideal Range Ambiguity NCS is defined by $n-1$ non-empty intersections and a point intersection. In a continuous space, for 2 agents to be adjacent according to the perception p_i, the distance between their boundaries must be less than d_i^{aed}.

Overpopulation NCS Detection (Fig. 3d). It is two ***adjacent*** *Context Agents* that give similar predictions and can merge their common boundary without altering their other ranges. An ideal Overpopulation NCS is defined by $n-1$ equal ranges and a point intersection. d_i^{lpd} is used here to detect the adjacency. This NCS triggers an instantaneous fusion.

Incompetence NCS Detection (Fig. 3e). The Incompetence NCS detects ***empty areas*** (red dashed zones) within the neighborhood of the current *perceptions* and *Context Agents*. An empty area is detected if its dimensions in all perceptions p_i are greater than d_i^{aed}. As few empty zones as possible are constructed.

Model Ambiguity NCS Detection. Any learning technique requires data. If the volume or quality of data is not adequate, this NCS is detected.

Once detected, *learning inaccuracies* (other than the Overpopulation NCS) are associated with a certain priority to be addressed. The order of priority from highest to lowest is: Model Ambiguity, Con-



(a) Conflict NCS (b) Concurrency NCS (c) Range Ambiguity NCS (d) Overpopulation NCS (e) Incompetence NCS (f) Model Ambiguity NCS

Figure 3: 2D *learning inaccuracies* with 1D projections. The diagonal crosses represent the *perceptions* that are used to generate active *learning situations* in order to solve the *learning inaccuracies*

flict, Concurrency, Incompetence and Range Ambiguity. Each *learning inaccuracy* leads to a future active *learning situation* that is stored in a priority queue with the area of the overlap or void if there is one. Interrogating all *Context Agents* for *learning inaccuracies* is computationally expensive for large number of agents. The detection is only performed among all *Context Agents* in the neighborhood of the considered *perceptions*.

3.2.3 New Learning Rules

Following the introduction of the neighborhood and the *learning inaccuracies*, we present here the new learning rules.

The inaccuracy margin is removed and there is a dynamic error margin for each *Context Agent*. This error allows the agent to self-assess the accuracy of its model. For *learning situations* within its validity ranges, the better its predictions, the lower its error margin; the worse its predictions, the higher its error margin. A local error margin is defined as the average of the dynamic error margins of the *Context Agents* neighboring the *perceptions*.

A good prediction of the *Best Context Agent* is when the distance to the *learning situation* is below the local error margin. It results in an increase of its confidence c^j and an update of its model with the *learning situation*. This way, the *Context Agents*' models are regularly updated to be robust to noise.

When the distance to the *learning situation* is greater than the local error margin, a *Context Agent* defines its prediction as bad. As it moves one of its ranges to exclude the current *perception*, it uses its neighborhood to destroy any overlap with another *Context Agent* and it decreases its confidence.

When there isn't any *Valid Context Agent*, the *Best Closest Context Agent* extends its ranges the cover the *perceptions*. If the creation of a new *Context Agent* is needed and if there are neighbors, they are used to initialize the properties of the new agent. If there aren't any neighbors, the created agent uses initialization values based on the perception limits of the search space and on the parameters chosen by the system user.

In relation to range extension, there is a limit beyond which a range can no longer be extended. The only way for a *Context Agent* to grow is then to merge with another *Context Agent* which doubles its confidence. This mechanism avoids oscillation behaviors for range modifications. d_i^{aed} is used as the critical validity range size for *Context Agents* subtractions.

3.2.4 The Active Strategy

The addition of the neighborhood allows to detect the *learning inaccuracies* in the learning process (section 3.1) and to define an active learning strategy that deals with the neighborhood inconsistencies. The strategy for solving the *neighborhood inaccuracies* is based on the availability of specific active *learning situations*. *Learning inaccuracies* are detected within the neighborhood at the end of learning cycles. Their associated priority defines their processing order. Each *learning inaccuracy* generates a *learning situation* that is requested to the oracle.

Conflict and Concurrency NCS Resolution. Conflict and concurrency *learning inaccuracies* generate *learning situations* in the overlap centers (figures 3a and 3b). This causes several *Valid Context Agents*. The closest *Context Agent* to the *learning situation* is the *Best Context Agent*. The other *Context Agents* reduce one of their ranges to exclude the current *perceptions* and destroy the overlap while minimizing volume loss.

Range Ambiguity NCS Resolution. An ambiguity *learning inaccuracy* generates two *learning situations* at a distance of d_i^{aed} from the borders of each *Context Agent* (Fig. 3c). These *learning situations* refine the borders.

Incompetence NCS Resolution. Each Incompetence *learning inaccuracy* generates a *learning situation*. It leads to a creation of a new *Contest Agent*. The incompetent area is filled with the dimensions of the *learning inaccuracy* void area (Fig. 3e).

Model Ambiguity NCS Resolution. In this case, the linear regression demands a certain quantity of *learning situations* to complete the model. As many *learning situations* as needed are generated at random positions within the validity ranges of the requesting *Context Agent* (Fig. 3f).

4 Experimentations

In this section, we compare learning without any NCS detection and resolution with the results taking into account only one NCS at a time, and then with all NCS together. Each learning experience is stopped after 1000 cycles because it is enough to explore the entire space. The learning mechanism receives a *learning situation* at each cycle. The metrics are averaged over 10 learning experiences. The models to be learnt are two linear models whose spatial distribution is as shown in Fig. 2a. On average, the duration of an experiment is about 3 seconds with the features[1] of the computer we are using. This work is planned to scale up with the number of dimensions of the search space, it is more demonstrative to present it as a 2D problem.

4.1 Metrics

We present here the metrics used during the experiments. For simplicity, we remove the n indices that symbolize the space dimension.

Generalization. To evaluate the ability of the mechanism to generalize, we look at the number of *Context Agents* n_{Ctxt} used to represent the space.

Exploration of the search space. We are interested in different types of volumes that measure the exploration of the search space.

- $\mathcal{V}_{Ctxt} = (\sum_j \mathcal{V}_{Cj} - \mathcal{V}_{Cflt} - \mathcal{V}_{Conc})/\mathcal{V}_{total}$ It is the normalized volume explored by all the *Context Agents*.
- $\mathcal{V}_{Inac} = \mathcal{V}_{Cflt} + \mathcal{V}_{Conc} + \mathcal{V}_{Inc}$ It is the volume of *learning inaccuracies*.
- $\mathcal{V}_{Cflt} = (\sum_{(j,k),1\leqslant j<k\leqslant n_{Ctxt}, f^j \neq f^k} \mathcal{V}_{C^j \cap C^k})/\mathcal{V}_{total}$ It is the normalized volume of conflicts.
- $\mathcal{V}_{Conc} = (\sum_{(j,k),1\leqslant j<k\leqslant \mathcal{N}_{ctxt}, f^j \approx f^k} \mathcal{V}_{C^j \cap C^k})/\mathcal{V}_{total}$ It is the normalized volume of concurrencies.
- $\mathcal{V}_{Inc} = 1 - \mathcal{V}_{Ctxt}$ It is the normalized volume of the unexplored search space.

$\mathcal{V}_{total} = \prod_i (p_i^{max} - p_i^{min})$ is the volume of the search space with p_i^{min} and p_i^{max} the minimum and maximum perceptions experienced.

Learning Data. We are interested in the number of *learning situations* randomly supplied and actively requested.

- $\mathcal{L}_{Rdm}$ are randomly generated *learning situations* from a uniform distribution in the interval $[-100;100]$ for each perception p_i.
- $\mathcal{L}_A$ represents all the active *learning situations*.
- $\mathcal{L}_A^{Model}$, $\mathcal{L}_A^{Cflt}$, $\mathcal{L}_A^{Conc}$, $\mathcal{L}_A^{Inc}$ and $\mathcal{L}_A^{Rge}$ are all the different kinds of active *learning situations* (Model Ambiguity, Conflict, Concurrency, Incompetence and Range Ambiguity).

Prediction. To evaluate the proposals of the learning mechanism, we calculate the normalized prediction error $\mathcal{E}_O$ according to the oracle predictions: $\mathcal{E}_O = |O - O'|/|O|$. The prediction metric is calculated over 250 exploitation random cycles.

4.2 Results

In this section, we present the results obtained by comparing the resolutions of the different *learning inaccuracies*. The metric values we give here are the averages available on T. 1.

4.2.1 Results without NCS Resolution

When no NCS is taken into account, it is visually noticeable (Fig. 4a) that the volume of *learning inaccuracies* is significant (42.28%). 159 *Context Agents* cover only 80.23% of the space and there is an error of 55.4% on the predictions.

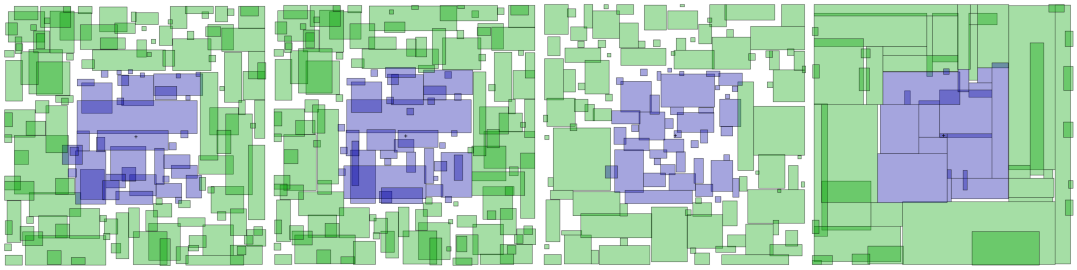
4.2.2 Results with NCS Resolution Alone

For each of the following cases, we compare the metric values obtained with the previous case as a reference.

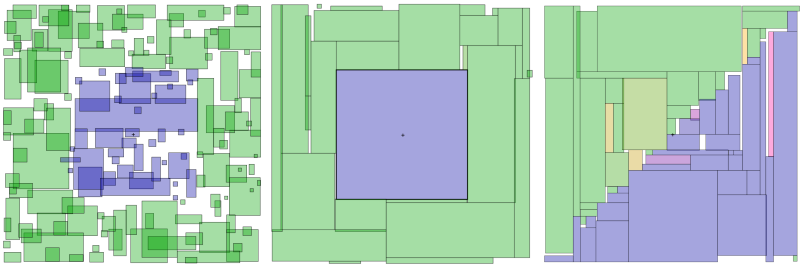
Conflict NCS. The treatment of conflicts does not bring any obvious visual difference (blue/green intersections Fig. 4b). 15 active *learning situations* are requested to reduces the volume of conflicts from 0.7% to 0.12%.

Concurrency NCS. The treatment of concurrencies brings an exploration almost without any overlap (blue/blue and green/green intersections Fig. 4c). 136 active *learning situations* are required to reduce the volume of concurrency from 21.81% to 1.38%.

[1] Intel(R) Core(TM) i7-6600U CPU 2.20 GHz 2.81 GHz, RAM 32 GB



(a) None (b) Conflicts (c) Concurrencies (d) Incompetencies



(e) Ambiguities (f) All (g) Dynamic Model

Figure 4: Screen shots of learning experiences after 1000 training cycles with different NCS resolution. Each color is a different linear model.

Table 1: Metrics of exploration, learning data, generalization and prediction averaged over 10 episodes of 1000 learning cycles for different NCS resolutions (presented section 4.1). Red values represent worse results than reference case (No NCS), green values represent better results than the reference case and bold values represent the impacted learning data for each case. The dynamic column represents an additional experiment made with 1000 more learning cycles on a different disposition of the models after a classical learning episode with all the NCS

Metrics		No NCS		Conflicts		Concurrencies		Incompetences		Ambiguities		All NCS		Dynamic	
$\mathcal{V}_{Ctxt}$	(%)	80.23	± 1.38	78.4	± 1.11	**73.36**	± 1.96	**97.97**	± 0.42	**75.71**	± 0.98	**97.23**	± 0.58	92.21	± 4.77
$\mathcal{V}_{Inac}$	(%)	42.28	± 1.44	43.39	± 1.65	**28.26**	± 1.94	**12.7**	± 2.61	42.94	± 0.92	**3.67**	± 0.57	12.92	± 5.53
$\mathcal{V}_{Cflt}$	(%)	0.7	± 0.27	**0.12**	± 0.08	**0.24**	± 0.15	0.82	± 0.56	**0.19**	± 0.18	**0.13**	± 0.12	4.9	± 3.59
$\mathcal{V}_{Conc}$	(%)	21.81	± 2.24	21.67	± 1.89	**1.38**	± 0.2	**9.85**	± 2.59	18.46	± 0.91	**0.76**	± 0.4	0.23	± 0.22
$\mathcal{V}_{Inc}$	(%)	19.77	± 1.38	21.6	± 1.11	**26.64**	± 1.96	**2.03**	± 0.42	**24.29**	± 0.98	**2.77**	± 0.58	7.79	± 4.77
$\mathcal{L}_{Rdm}$	(#)	923	± 4	908	± 4	786	± 6	581	± 45	792	± 17	220	± 55	127	± 37
$\mathcal{L}_{A}$	(#)	77	± 4	**93**	± 4	**214**	± 6	**419**	± 45	**208**	± 17	**780**	± 55	874	± 37
$\mathcal{L}_{A}^{Model}$	(#)	77	± 4	78	± 3	78	± 3	**41**	± 9	77	± 4	**43**	± 8	**8**	± 4
$\mathcal{L}_{A}^{Cflt}$	(#)	0	± 0	**15**	± 3	0	± 0	0	± 0	0	± 0	15	± 6	26	± 9
$\mathcal{L}_{A}^{Conc}$	(#)	0	± 0	0	± 0	**136**	± 4	0	± 0	0	± 0	**71**	± 11	58	± 15
$\mathcal{L}_{A}^{Inc}$	(#)	0	± 0	0	± 0	0	± 0	**378**	± 50	0	± 0	**415**	± 38	185	± 31
$\mathcal{L}_{A}^{Rge}$	(#)	0	± 0	0	± 0	0	± 0	0	± 0	**132**	± 17	**237**	± 34	598	± 68
n_{Ctxt}	(#)	159	± 7	165	± 6	**125**	± 10	**53**	± 6	159	± 7	**31**	± 5	59	± 10
$\mathcal{E}_{O}$	(%)	55.4	± 28.85	51.29	± 19.51	44.04	± 19.07	**2.68**	± 1.67	53.51	± 18.82	**1.81**	± 3.03	3.02	± 3.99

Fewer *Context Agents* (125) cover a smaller part of the search space 73.36%

Incompetence NCS. Incompetency processing results in an exploration with almost no gaps (white areas Fig. 4d). 378 active *learning situations* allow to decrease the void volume (2.03%), the concurrency volume (9.85%), and the Model Ambiguity NCS (41). 53 *Context Agents* cover 97.97% of the search space and the prediction error is reduced to 2.68%.

Range Ambiguities NCS. The treatment of ambiguities alone gives a similar exploration if not worse to the case without NCS treatment (Fig. 4e).

4.2.3 Results with All NCS Resolutions

When all NCS are processed during training Fig. 4f, exploration is close to the expected ideal Fig. 2c. 31 *Context Agents* cover 97.23% of the space. The volume of all *learning inaccuracies* drops from 42.28% to 3.67%, the lowest of all cases. Prediction error is the lowest with 1.81%.

4.2.4 Dynamic Model

Fig. 4g experiment tests the ability of the learning mechanism to reuse its knowledge and to adapt to a

variation. From an experiment of 1000 cycles with all NCS (Fig. 4f), a second experiment is conducted with 1000 learning cycles on a new model. T. 1 shows that the exploration performances are slightly worse but the quantity of model *learning situations* $\mathcal{L}_A^{Model}$ is much lower.

4.3 Discussion

When none of the *learning inaccuracies* are addressed, the metric results are far from ideal. The prediction error is very bad, which is explained by the very large amount of non-exploitable space. Processing NCS alone allows to validate the expected behaviors but the associated explorations do not converge towards the ideal exploration. When all NCS are treated, the number of *Context Agents* and the prediction error are the lowest of all the encountered cases, but there are still minute *learning inaccuracies*. The number of *Context Agents* still does not reach the optimum. To reach it, an additional treatment is missing that would allow the *Context Agents* to reorganize their validity ranges, i.e. to split if it allows a merge. The dynamic experiment shows the capacity of the learning mechanism to adapt its exploration and to reuse already known models to accelerate learning. The learning case that we present here is a simplified case to validate our approach. Further evaluations will focus on exploring more complex search spaces with non-linear models to be learnt which are closer to real complex systems.

5 Related work

The approach of integrating a local model within each *Context Agent* differs from classical regression and classification approaches. Piecewise learning, agentified and distributed, online and dynamic, allows adding internal mechanisms that guide learning by making it active, independently of the learning technique. In this section, we present similar methods and highlight their similarities and differences.

Data Selection. The approach presented in this paper is comparable to active learning, which consists of deciding which data should be labeled from a set of unlabeled data (Settles, 2009). It also reminds semi-supervised learning, which seeks to achieve better performance by combining unlabeled and labeled data (Li et al., 2017). Our work focuses on how multi-agent systems allow to actively request labeled data online with a generic approach independent of the learning models.

Domain, dimension and distribution/locality. Transfer learning uses related domains or tasks to enhance learning (Pan and Yang, 2009). Unsupervised learning uses only unlabeled data for data aggregation or dimensional reduction problems (Hastie et al., 2009). Distributed learning divides the training data into subsets before processing them (Lin et al., 2017). We find local learning work for classification based on generalization, continuity in prediction and scalability (Zantedeschi, 2018). Our approach is intended for different online learning problems of classification or regression depending on the implemented models with fixed dimensions. The dynamic experience shows that transfer learning is effectuated when *Context Agents* reorganize themselves to match the new geography.

Intrinsic Motivation. Beyond inverse reinforcement learning and learning by intention, in the fields of robotics, intrinsic motivation (Oudeyer et al., 2007; Bondu and Lemaire, 2007) is a way to guide exploration by focusing on the learning process itself. Intrinsic motivation is independent of the task, it promotes hierarchical learning of knowledge and the reuse of skills (Mirolli and Baldassarre, 2013). It is particularly found in the field of developmental robotics and focuses on the development of robots with infantile mental capacities (Cangelosi et al., 2015). This approach is intended for blank systems that learn from scratch. In the literature, there are mostly architectures that use different learning mechanisms (Bellas et al., 2010; Vernon et al., 2011). We take inspiration from this work but we are not yet experimenting on blank robotic systems.

Among these learning methods, our work falls within the framework of several learning domains. Our interest here is in adding internal mechanisms to our context-based meta-learning approach and not in comparing with other isolated learning methods that do not match all of our concerns.

6 Conclusion and Perspectives

As multi-agent systems are well suited to cope with the constraints of the sociotechnical world, we have proposed a learning paradigm based on the extension of the contextual learning pattern. This extension is based on the use of information shared in the neighborhood of a learning situation. We proposed a taxonomy of these situations and a solution based on agents cooperation. Self-observation of self-adaptive multi-agent systems allows to detect *learning inaccuracies* in the local models. The local active *learning situations* bring improved prediction performance by

anticipating ambiguous situations and solving them. All the behaviors of the cooperative agents are designed generic in order to be agnostic from the application domain, the learning technique and the number of dimensions. We are currently working in the field of robotics on multi-articulated robotic arms in order to learn their inverse kinematics model. Further work will also focus on the comparison with other learning techniques.

REFERENCES

Abbas, H., Shaheen, S., Elhoseny, M., Singh, A. K., and Alkhambashi, M. (2018). Systems thinking for developing sustainable complex smart cities based on self-regulated agent systems and fog computing. *Sustainable Computing: Informatics and Systems*, 19:204–213.

Ashby, W. R. (1956). Cybernetics and requisite variety. *An Introduction to Cybernetics*.

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.

Boes, J., Nigon, J., Verstaevel, N., Gleizes, M.-P., and Migeon, F. (2015). The self-adaptive context learning pattern: Overview and proposal. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 91–104. Springer.

Bondu, A. and Lemaire, V. (2007). Active learning using adaptive curiosity. In *International Conference on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems*.

Cangelosi, A., Schlesinger, M., and Smith, L. B. (2015). *Developmental robotics: From babies to robots*. MIT Press.

Chaput, H. H. (2004). *The constructivist learning architecture: A model of cognitive development for robust autonomous robots*. PhD thesis.

Drescher, G. L. (1991). *Made-up minds: a constructivist approach to artificial intelligence*. MIT press.

Ferber, J. (1999). *Multi-agent systems: an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading.

Georgé, J.-P., Gleizes, M.-P., and Camps, V. (2011). Cooperation. In *Self-organising Software*, pages 193–226. Springer.

Guerin, F. (2011). Learning like a baby: a survey of artificial intelligence approaches. *The Knowledge Engineering Review*, 26(2):209–236.

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.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). Unsupervised learning. In *The elements of statistical learning*, pages 485–585. Springer.

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.

Kearns, M. J., Schapire, R. E., and Sellie, L. M. (1994). Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141.

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*.

Lin, S.-B., Guo, X., and Zhou, D.-X. (2017). Distributed learning with regularized least squares. *The Journal of Machine Learning Research*, 18(1):3202–3232.

Mazac, S., Armetta, F., and Hassas, S. (2014). On bootstrapping sensori-motor patterns for a constructivist learning system in continuous environments. In *Artificial Life Conference Proceedings 14*, pages 160–167. MIT Press.

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.

Nigon, J., Gleizes, M.-P., and Migeon, F. (2016). Self-adaptive model generation for ambient systems. *Procedia Computer Science*, 83:675–679.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. (2007). Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2):265–286.

Pan, S. J. and Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Perotto, F. S. (2013). A computational constructivist model as an anticipatory learning mechanism for coupled agent–environment systems. *Constructivist Foundations*, 9(1):46–56.

Piaget, J. (1976). *Piaget's theory*. Springer.

Provost, J., Kuipers, B. J., and Miikkulainen, R. (2006). Developing navigation behavior through self-organizing distinctive-state abstraction. *Connection Science*, 18(2):159–172.

Russell, S. J. and Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Vernon, D., Von Hofsten, C., and Fadiga, L. (2011). *A roadmap for cognitive development in humanoid robots*, volume 11. Springer Science & Business Media.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Zantedeschi, V. (2018). *A Unified View of Local Learning: Theory and Algorithms for Enhancing Linear Models*. PhD thesis.