# On Learning in Collective Self-adaptive Systems: State of Practice and a 3D Framework

Mirko D'Angelo*, Simos Gerasimou†, Sona Ghahremani‡, Johannes Grohmann§, Ingrid Nunes¶, Evangelos Pournaras||, Sven Tomforde**
*Linnaeus University, Växjö, Sweden
†University of York, York, United Kingdom
‡Hasso Plattner Institute, Universität Potsdam, Potsdam, Germany
§University of Würzburg, Würzburg, Germany
¶Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
||ETH Zürich, Zürich, Switzerland
**Universität Kassel, Kassel, Germany
Email: mirko.dangelo@lnu.se, simos.gerasimou@york.ac.uk, sona.ghahremani@hpi.uni-potsdam.de, johannes.grohmann@uni-wuerzburg.de, ingridnunes@inf.ufrgs.br, epournaras@ethz.ch, stomforde@uni-kassel.de

*Abstract*—**Collective self-adaptive systems (CSAS) are distributed and interconnected systems composed of multiple agents that can perform complex tasks such as environmental data collection, search and rescue operations, and discovery of natural resources. By providing individual agents with learning capabilities, CSAS can cope with challenges related to distributed sensing and decision-making and operate in uncertain environments. This unique characteristic of CSAS enables the collective to exhibit robust behaviour while achieving system-wide and agent-specific goals. Although learning has been explored in many CSAS applications, selecting suitable learning models and techniques remains a significant challenge that is heavily influenced by expert knowledge. We address this gap by performing a multifaceted analysis of existing CSAS with learning capabilities reported in the literature. Based on this analysis, we introduce a 3D framework that illustrates the learning aspects of CSAS considering the dimensions of *autonomy*, *knowledge access*, and *behaviour*, and facilitates the selection of learning techniques and models. Finally, using example applications from this analysis, we derive open challenges and highlight the need for research on collaborative, resilient and privacy-aware mechanisms for CSAS.**

*Index Terms*—**self-adaptive systems, learning, distributed systems, autonomic systems, taxonomy, multi-agent systems**

## I. Introduction

Distributed and interconnected software systems are increasingly deployed in application domains characterised by dynamic environments, evolving requirements, and unpredictable failures. Examples of such systems include network infrastructures [1], smart cities [2], and traffic ecosystems [3] with autonomous vehicles [4]. During operation, these systems might encounter several unexpected scenarios including variations in system performance, sudden changes in system workload and component failures. Dealing effectively with such scenarios entails enhancing these systems with self-adaptive capabilities so that they can autonomously identify abnormal situations, analyse alternative adaptation options and, finally, self-adapt to a suitable new (system-wide) configuration [5]–[7].

All authors contributed equally and are listed in alphabetical order.

The feasibility of effective self-adaptation in distributed systems through centralised or hierarchical control is questionable [8]–[10]. Indeed, using a *single* autonomic manager, i.e., a single closed-loop controller such as the MAPE-K loop [11] or the Observer/Controller tandem [12], to monitor a distributed system and act upon system resources typically depends on unrealistic assumptions. These assumptions, include, but not limited to, the autonomic manager's ability to obtain a consistent and complete system-wide (i.e., global) view, and its capacity to process this information, reason, and distribute the decisions on time. Most importantly, centralised control requires trust and is susceptible to well-known problems including limited scalability and single points of failures [8].

Recent research has demonstrated that providing self-adaptive capabilities to each component of a distributed system can address the challenges faced by centralised control [9], [13]–[15]. Based on this paradigm, each system component, also known as an *agent* [16], can monitor its own environment and interact with its peers leading to an emergent behaviour of the system as a whole. This paradigm shift increases resilience, improves scalability and eliminates single points of failures [8], [17], [18]. Following Mitchell's definition [19], we refer to *collective self-adaptive systems* (CSAS) as distributed systems comprising multiple agents such that each agent: (i) can interact with other agents either directly or indirectly; (ii) does not individually possess system-wide knowledge; (iii) can exhibit learning to expand its personal knowledge; and (iv) can make decisions based on collective or aggregated knowledge from some of its peers.

Achieving effective self-adaptation in CSAS is undoubtedly complex for several reasons. First, the environment in which each agent might be situated is often highly dynamic, and significantly more complex than that of systems with centralised control. This CSAS feature makes it impossible to predict at design time all possible scenarios that can occur at runtime and provide CSAS agents with pre-specified adaptation plans. Second, system-wide knowledge is distributed among the agents, entailing that advanced mechanisms should be used for efficient knowledge sharing and acquisition between agents. Also, decision-making, both on agent and system levels, requires sophisticated techniques for effective coordination, conflict resolution, and avoidance of suboptimal behaviour [13], [20].

To address these issues, CSAS agents must be enhanced with *learning capabilities*, allowing them both to instantiate learning models using the knowledge acquired from observing the environment and their peers, and to refine these models by assessing the outcomes of their actions. These learning models provide the means to improve quality attributes (e.g., performance, cost) of an individual agent or the entire collective [21].

Although learning has been explored in many self-adaptive applications, e.g. [2], [13], [14], the use of different types of learning models in CSAS is still a handcrafted process that relies heavily on domain expertise. More specifically, the adoption of learning involves an understanding of the application particularities as well as the requirements that these particularities impose on the designed learning model. However, there is no body of knowledge about best practices in learning-enabled CSAS leaving researchers and practitioners without guidance on how to mitigate this recurrent CSAS concern. Existing surveys on multi-agent learning are restricted to reviewing alternative techniques proposed in this context [17], [22], [23]. In particular, the benefits of using self-organisation to address the challenging research issues in multi-agent systems are studied in [17]. Cooperative and competitive multi-agent techniques are investigated in [23], while [22] analyses the differences between these techniques. In contrast, *our study focuses on understanding the state of practice in developing learning-enabled CSAS.* To this end, we investigate research related to CSAS that use learning techniques and models as a means of enabling CSAS agents to adapt their behaviour when encountering scenarios unanticipated at design time.

Our investigation is made by means of a systematic literature review of 52 related research papers out of 215 candidates, selected using the research method introduced in Section II. In Section III, we analyse the selected studies from multiple perspectives, including the CSAS characteristics, application domain and type of employed agents to understand the virtue of learning in the context of CSAS. Based on this analysis, we introduce in Section IV a three-dimensional framework that consolidates key aspects (i.e., *autonomy*, *knowledge access* and *behaviour*) that must be taken into account to instantiate learning-based solutions when developing CSAS. Engineers can employ the proposed framework as a reference to reinforce their design decisions associated with learning-based agents in CSAS. Finally, we discuss related open challenges in Section V and conclude the paper in Section VI.

The main contributions of this paper are as follows: (1) a systematic literature review that captures the state-of-practice of learning-based CSAS; (2) a three-dimensional framework for classifying CSAS based on *autonomy*, *knowledge access* and *behaviour* characteristics, that enables the selection of suitable learning techniques and models; and (3) a discussion of learning-specific open challenges for CSAS.

## II. Research Method

The adopted research method introduced in this section follows the standard practice in systematic literature reviews [24]. Since our study focuses on learning-based CSAS, the scope of the review is restricted to CSAS solutions in which agents within the collective are enhanced with learning abilities. [1]

### A. Research Questions

We use a set of research questions to steer our review. Each selected paper (representing a learning-based CSAS solution) is analysed considering the following research questions.

**RQ1**: What are the characteristics of the described CSAS?
**RQ2**: What is the purpose of learning within the CSAS?
**RQ3**: Which learning techniques are employed?
**RQ4**: What are the triggers to update the learning models?

### B. Selection Method

We specify next the strategy adopted to search the literature and the criteria used to select the analysed studies.

*1) Search Terms and Query String:* Our goal is to identify research papers describing how decentralised learning is used within a CSAS. To this end, we used these search terms and synonyms: (i) **self-adaptive**: self-adapt*, self-organi*, autonom*; (ii) **software**: application, system; and (iii) **learning**. We combined these terms and used the search string below.

```
(self-adapt* OR self-organi* OR autonom*) AND (software OR application OR system) AND learning
```

*2) Searched Databases and Venues:* We investigate how learning is used in the context of distributed collective self-adaptive and self-organising systems. We focus on reviewing advanced and high-quality studies published in the main conferences and journals in the areas of self-adaptive, self-organising, and multi-agent systems (MAS); see Table I. We exclude workshop papers as they typically report work-in-progress. The listed venues were searched using the respective databases. Figure 1 shows the adopted multi-stage search and selection process. We should emphasise that our objective is the analysis of learning-based CSAS solutions rather than the investigation of sophisticated machine learning techniques. Likewise, we focus on top venues researching autonomous and multi-agent systems (e.g., AAMAS, JAAMAS). While we do not exhaustively search the entire MAS domain, initial exploratory searches (Stage 1) indicate that these venues provide a representative number of studies related to the scope of our survey. Other top venues specialised in machine learning and not necessarily in CSAS (e.g. NIPS, ICML, etc.) are excluded to keep the analysis more focused and manageable in terms of number of reviewed papers, while our search and filter strategy (Stages 3 and 4) remains highly comprehensive within the defined scope and filters out irrelevant studies. A more extensive review with broader scope is a subject of future work.

[1]A replication package of our systematic literature review, including the list of selected studies and details of the collected and analysed data is available at https://github.com/mi-da/CSAS-learning.

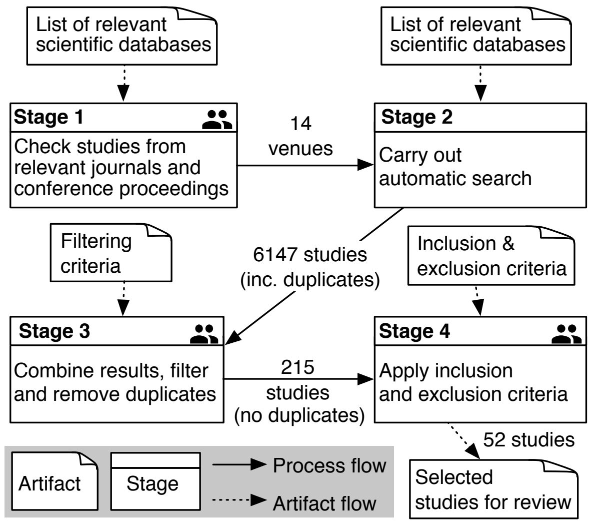
List of relevant scientific databases
List of relevant scientific databases
Stage 1
Check studies from relevant journals and conference proceedings
14 venues
Stage 2
Carry out automatic search
Filtering criteria
6147 studies (inc. duplicates)
Inclusion & exclusion criteria
Stage 3
Combine results, filter and remove duplicates
215 studies (no duplicates)
Stage 4
Apply inclusion and exclusion criteria
52 studies
Selected studies for review
Artifact
Stage
Process flow
Artifact flow

Fig. 1. Multi-stage search and selection process

We carried out automatic searches on the following databases: ACM DL, IEEE Xplore, Springer and Elsevier. To ensure that the selected studies capture recent scientific advances, we considered studies published in or after 2000.

*3) Inclusion and Exclusion Criteria:* To select studies suitable for analysis, we considered four inclusion criteria (IC) and three exclusion criteria (EC); see Table II. A study is included for review if it satisfies all IC and no EC.

*4) Study selection procedure:* The preliminary study selection is conducted by two reviewers. The resulted studies from the initial database querying (prior to any filtering by the venues) have been manually sampled. The sampled studies are analysed by the two reviewers to confirm their relevance (and of the corresponding venues) to the scope of the survey and remove irrelevant studies/venues. In case of doubt, advice from the other five reviewers resolved the dilemma. To exclude a certain venue from the search results, a full agreement between all the reviewers was required. However, a minimum of one positive vote advocating to include a certain venue resulted in its inclusion with the rationale to include as many potentially relevant venues as possible, thus, minimising false negatives. All refined (by venues) search results are manually inspected by the two reviewers to confirm their relevance to the scope of the survey. Next, the retrieved papers are equally distributed among the seven reviewers to be further investigated regarding the IC and EC (Table II). When necessary, the opinion of a second reviewer was requested to address any uncertainty (e.g., confirm if a paper is eligible or not). Finally, the remaining studies are included in the survey for further analysis.

## C. Analysis Method

We systematically analyse all selected studies to answer the research questions (cf. Section II-A). Each study is analysed by a different reviewer than the one who applied the IC and EC, thus, enhancing the validity of the review. For each study, we extract the following data items. We show in parenthesis the research question covered by each data item.

Table I. LIST OF INCLUDED VENUES

| Name | Venue | Publisher | # Studies by Venue |
|---|---|---|---|
| AAMAS | Conference | ACM | 58 |
| ASE | Conference | IEEE/ACM | 3 |
| FSE | Symposium | ACM | 4 |
| ICAC | Conference | IEEE | 28 |
| ICSE | Conference | IEEE/ACM | 22 |
| SASO | Conference | IEEE | 45 |
| SEAMS | Symposium | ACM | 7 |
| Applied Soft Comp. | Journal | Elsevier | 19 |
| ASE | Journal | IEEE | 0 |
| JAAMAS | Journal | Springer | 0 |
| TAAS | Journal | ACM | 25 |
| TLT | Journal | IEEE | 2 |
| TOSEM | Journal | ACM | 0 |
| TSE | Journal | IEEE | 2 |

- **Application Domain** (RQ1): classify the domain of the developed CSAS application;
- **Agents** (RQ1): identify agents within the collective with learning abilities;
- **Autonomy** (RQ1): determine whether agents act autonomously or are supervised by external entities;
- **Knowledge Access** (RQ1): assess the amount of information provided to an agent by its peers;
- **Behaviour** (RQ1): evaluate how an agent behaves with respect to other agents' behaviour;
- **Learning Tasks** (RQ2): establish the objectives of learning by an agent and the collective;
- **Emergent Behaviour** (RQ2): determine the behaviour that emerges from agents' interactions and is observed at the system level as system properties;
- **Learning Technique** (RQ3): classify the techniques used by agents to enable learning;
- **Learning Trigger** (RQ4): identify the triggers driving knowledge processing and refinement of learning models.

## D. Selected Studies

We ran the search query on the scientific databases on October 24, 2018, and obtained a total of 6147 studies (including duplicates). After refining the results by the publication year, the number of studies reduced to 5405 (including duplicates). Further refinement according to the selected venues resulted in 215 studies (excluding duplicates). Each study was evaluated against the IC and EC. We show in Table II the number of studies that satisfied each criterion. In summary, 52 studies satisfy all IC and no EC, which are included for review.

Table II. INCLUSION (IC) AND EXCLUSION (EC) CRITERIA

| **Inclusion Criteria** | **#Studies satisfying each criterion** |
|---|---|
| **IC-1**: The paper describes an implemented software system | 158 |
| **IC-2**: The system involves multiple agents | 109 |
| **IC-3**: At least one agent is provided with the capability of learning | 181 |
| **IC-4**: There is no centralised learning process | 88 |
| **Exclusion Criteria** | |
| **EC-1**: The learning process is not described in the paper | 53 |
| **EC-2**: The paper is a glossary, extended abstract, tutorial, etc. | 8 |
| **EC-3**: A more complete version of the paper is selected for review | 6 |
| **Total included studies (satisfying all IC and no EC)** | **52** |

## III. Results and Analysis

Based on our analysis of the selected studies, we discuss observations regarding the research questions (cf. Section II-A).

### A. RQ1: CSAS Characteristics

*1) Application Domain and Agents:* The first step of extracting CSAS characteristics considers **application domains** and **agents** with the ability to learn. As shown in Figure 2, the most dominant applications are cyber-physical systems (CPS) such as robotics, sensor systems and smart energy applications. This is inline with the increasing interest in CPS, especially in the area of self-adaptive systems [25]. Learning agents in these studies include robots [26]–[33], smarts sensors [34]–[41] and smart grid elements [42]–[45]. Network-based applications are the second most explored domain with network controllers [46], [47] or network nodes [48]–[53] being the learning agents. CPS and network-based applications are employed in more than half of the studies (53%). This is interesting since these two interconnected domains both relate to infrastructure applications (i.e., industrial or facility-based processes that exist in the physical world [25]). Other infrastructure applications include traffic scenarios with cars [54] and traffic controllers [55]–[58] as agents, cooperative games with agents as players [59]–[63], and market applications with investors [64], sellers [65], or task allocation managers [66].

In contrast to the infrastructure applications, some studies consider more applied applications such as task allocation, scheduling and classification. In [67]–[70] abstract entities (agents) are adopted to tackle the task allocation problem in a distributed manner, while software agents are used in [71] for classification and in [72] for scheduling. Although the majority of applications belong to the domains mentioned above, the classification of the applications is not exclusive. For instance, the study in [73] introduces a CSAS solution for simulating human motor units in which agents are nodes of an artificial neural network. Finally, four studies describe the solution to learning in CSAS using abstract multi-agent organisational models but do not report any concrete applications [74]–[77].

*2) Autonomy:* We define CSAS **autonomy** to be the level of self-authorisation provided to agents within the collective. An agent is autonomous when there is no external or internal (i.e., by other agents) control over its behaviour. Most of the analysed studies assume *full autonomy* for all agents, i.e., the agents are not supervised, and all their decisions are put into action in their environment. This implies that all agents are responsible for their actions and cannot be overwritten by a hierarchically higher entity. This interesting observation can be partly explained by our search strategy which favours decentralised agents and penalises the use of a centralised learning process (cf. inclusion criterion IC-4 in Table II). We found only six studies (∼12% in Table III) explicitly modelling CSAS agents with *restricted autonomy* [34], [38], [45], [58], [66], [72]. Within a restricted autonomy setting, there is at least one agent whose actions are supervised and could be overwritten by other agents.

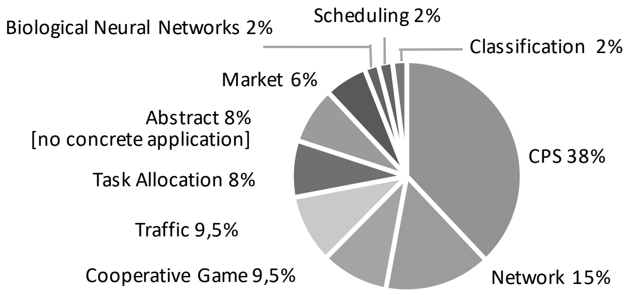

Fig. 2. Distribution of application domains among the reviewed studies

*3) Knowledge Access:* Our analysis reveals that some agent-based models explicitly exchange learning information with each other. We call this concept **knowledge access**. CSAS support the definition of different levels of knowledge access and our analysis identifies five different levels (see Table III).

Many reviewed studies (∼35%) consider *minimal* knowledge access settings, i.e., agents do not exchange information with each other. Hence, each agent should reason and act based on its own view and experiences. The opposite extreme, termed *maximal* knowledge access, is less common with only 8 studies [29], [31], [35], [62], [65], [72], [75], [76]. Here, all agents share complete knowledge with each other aiming to achieve the highest possible benefit for the collective.

Interestingly, the second largest group of studies (∼ 31%) applies a form of local or *neighbourhood*-like knowledge access. This means that, at least to a certain degree, agents share information with "nearby" agents within their neighbourhood. This type of limited knowledge access could occur because agents outside a neighbourhood are either unknown, untrustworthy or too distant (communication is infeasible).

Another group of studies [28], [33], [38], [49], [53], [73], [77] consider CSAS in which agents share *limited* information and not their full state. This limited information could take the form of selected agent configurations [73], adaptation tactics [28] or expected rewards [33], [38], [53]. Although the term can subsume many degrees of knowledge access, information sharing between agents in these studies is not enough to directly influence the adopted learning processes.

Finally, three studies [42], [45], [69] explicitly consider *tunable* degrees of knowledge access and investigate how adjusting this degree influences the learning behaviour of agents within the collective.

*4) Behaviour:* The **behaviour** of agents within a CSAS indicates whether an agent has the objective to maximise its own utility, i.e., it is *selfish*, to maximise the CSAS utility, i.e., it is *altruistic*, or adopts an intermediate role. Our analysis shows the absence of a clear-cut between selfish and altruistic agents but the existence of a continuum space comprising different modalities and combinations (see Table III).

In most studies (∼ 46%), agents are purely selfish. Kraemer et al. [26] apply independent Q-learning [78]. In this reinforcement learning technique, each agent learns values of its own actions, disregarding other learning agents in the environment. Similarly, Fang et al. [39] introduce a Bayesian-

Table III. CLASSIFICATION BASED ON AUTONOMY, KNOWLEDGE ACCESS, AND BEHAVIOUR

| **Autonomy** | **# Studies** | **Percentage** |
|---|---|---|
| Full Autonomy | 46 | 88,46% |
| Restricted Autonomy | 6 | 11,54% |
| **Knowledge Access** | **# Studies** | **Percentage** |
| Minimal | 18 | 34.62% |
| Neighborhood | 16 | 30.77% |
| Maximal | 8 | 15.38% |
| Limited | 7 | 13.46% |
| Tunable (Various evaluations) | 3 | 5.77% |
| **Behaviour** | **# Studies** | **Percentage** |
| Selfish | 24 | 46.15% |
| Selfish (collaborative) | 9 | 17.31% |
| Altruistic | 7 | 13.46% |
| Altruistic (collaborative) | 7 | 13.46% |
| Altruistic locally / Selfish globally | 3 | 5.77% |
| Both versions explored | 2 | 3.85% |

based learning algorithm in which all model inferences can be done independently for each selfish agent without any form of inter-agent communication.

Other studies ($\sim 17\%$) experiment with *selfish but collaborative* agents. This could take the form of a decentralised structural adaptation method applied locally to each selfish CSAS agent [74]. The collaborative aspect involves pairs of agents exhibiting cooperative behaviour and jointly estimating the utility of changing their learning models.

Moreover, there are studies in which agents are *altruistic* ($\sim 13\%$). Coordination between agents typically occurs through communication, e.g., using a mutual notification algorithm that is proven to converge to the system's optimal solution [33]. Other scenarios comprise agents that are *altruistic and collaborative* ($\sim 13\%$), as in [57], where a collaborative and self-organising traffic control system is designed through dynamic traffic light coordination. Finally, in $\sim 6\%$ of the studies, agents are *altruistic locally* (neighbourhood) but *globally selfish* [28], [55], [56]. We only found two studies that explore both altruistic and selfish agents [42], [59].

An interesting outcome of our review is the following. In several studies [30], [35], [36], [39], [43]–[45], [48], [49], [53], [61], [63], [69], [70], the system is designed in a way that each individual selfish agent learns a "learning task" and is not aware of the fact that it is collaborating. In these circumstances, the emergent behaviour of the system is the result of the agents' unaware collaboration and is often a system-wide collaboration scheme achieving a global goal. Moreover, our review highlights that only a few studies explore incentivisation/penalty mechanisms to promote agent collaboration [38], [44], [64], [66], [75]. In contrast, most of the studies rely on the assumption that agents follow the same goals within the collective. We highlight that such simplifying assumption limits the real-world applicability of the proposed techniques. In fact, CSAS can be deployed in open networks comprising untrusted and malicious agents that is not possible to control or trust a priori. Hence, an open CSAS challenge is the smooth integration of incentivisation and learning mechanisms.

## *B. RQ2: Learning Purpose*

The purpose of CSAS is defined by its learning tasks, which typically correspond to system-wide objectives. Besides the learning tasks, interactions between agents (intentional or inadvertent) may lead to emergent behaviour for the collective. The emergent behaviour cannot be predicted a priori as it is not a simple aggregation of the individual agents' behaviour [18], [19]. To answer the second research question, we analyse the studies based on their learning tasks (**learning purpose**) and possible emergent behaviours. The analysis of the relationship between these two aspects revealed the following observations.

A key difference between the studies is the learning task and emergent behaviour of interest. Several studies do not report any specific emergent behaviour but associate the emergent behaviour with the anticipated learning task [37], [52]. In a subset of these studies, the learning task and the associated emergent behaviour are fulfilled without requiring the agents to collaborate actively. For instance, in [28] the learning task to score a goal is fulfilled by each agent (a player of the team) individually learning adversary soccer moves that contribute to scoring a goal (emergent behaviour). Similarly, Beetz et al. [32] show that the learning task for the robots to play soccer can be achieved by each robot learning individually, without collaboration, to move towards a predefined target.

Another subset of studies with similar learning task and associated emergent behaviour have system-wide collaborative tasks [27], [37], [62], [63]. In these studies, learning techniques are directly employed to support collaborative learning, i.e., agents are aware of their collaboration. For instance, agents actively collaborate to win a cooperative game [62], [63], smart sensors cooperate to patrol an area [37], or soccer robots collaborate to form a squad and win the game [27].

Another interesting observation concerns studies in which emergent behaviour is different than the learning task. In these settings, the learning task is independent (and non-cooperative) per agent, and system-wide collaboration is realised as emergent behaviour. In [59]–[61], agents learn individually how to play the specified games and cooperate to achieve a positive outcome, i.e., win the game. Similarly, within the network domain, agents individually learn to propagate messages [50], predict the loss [48], and other individual tasks [46], [49], [53]. In this context, the emergent CSAS behaviour is the system-wide optimal dissemination of information. Other studies address learning tasks within the smart energy domain [42]–[45]; agents learn individually to solve allocation problems [42], predict energy consumption [43] or prosumers behaviour [44], which helps the team to learn system-wide energy optimisation strategies (emergent behaviour). In [35], [38], [39], [41], emergent behaviour involves agents (smart sensors) learning system-wide sensor configurations by individually learning self-configuration strategies.

In summary, we identified two main groups regarding the relation between learning tasks and emergent behaviour. One group concerns studies that associate the emergent behaviour to the exact anticipated learning task of the collective. The learning task in these studies is either a collection of individual non-collaborative tasks (e.g., each soccer agent scoring a goal on their own [27]) or a system-wide collaboration among the agents (e.g., smart sensors collaborating to patrol an area successful [37]). In the other group of studies, collaboration between the agents emerges to a different behaviour than the anticipated learning task (e.g., [42]–[44]). These studies often define learning tasks as individual goals for the agents and agents are not aware of any implicit collaboration among them. As a result, a system-wide collaboration emerges.

## C. RQ3: Learning Techniques

*Reinforcement Learning (RL)* is a widely used technique in CSAS. Referring to Table IV, the results from our review highlight that $\sim 60\%$ of the selected studies use RL. In [67] and [68] learning is used to self-organise the network. Q-learning is chosen due to its simplicity and suitability for representing the behaviour of the proposed mechanism in terms of actions and rewards [68]. In [55], the authors deal with multi-policy optimisation problems in self-organising systems using Distributed W-Learning (DWL), an RL approach inspired by W-Learning [79], where each policy is implemented by a single Q-learning process. The authors claim that Q-learning is a suitable learning and optimisation technique in situations where no pre-specified model of the environment is available. Similarly, in [26], RL is used in decentralised planning problems since the agents do not need to know a model a priori but can learn policies via repeated interaction with the environment and among each other. Multi-agent reinforcement learning (MARL) is employed in [43] to solve problems in a distributed manner in non-stationary environments when centralised control becomes infeasible. MARL is also used in [54] to tackle the complexity emerging in MAS domains, as it enables adaptive and autonomous agents to improve their learning models from experience.

The wide adoption of RL in CSAS lies in its simple and straightforward resemblance between the actions and rewards of its theoretical model and the behaviour of agents in the employed application domains (cf. Section III-A) [79]. Most importantly, RL techniques (e.g., Q-learning, W-learning) do not necessarily need a model of the environment and can learn directly from raw experience. In fact, in large-scale decentralised application domains of interacting agents (e.g., CPS with many robots and smart IoT devices), building a complete model of the environment is a challenging and often an impossible task. Since self-adaptive systems aim at tackling scenarios unpredicted at design time, such models of the environment are not only hard to realise, but also need to change in a timely manner together with the CSAS variability. This is a primary factor driving the use of model-free learning approaches for decentralised CSAS [75]. The use of model-based RL techniques for CSAS is also an open research area.

Since RL techniques are widely used for enhancing CSAS with learning abilities, some of the reviewed studies focus on improving RL [26], [29], [43], [69]. The need for additional exploration in decentralised RL settings entails that the collective should converge fast to its decision. Driven by research in [67], which shows that self-organisation leads to faster convergence of the learning task, [69] proposes a hierarchical self-organising framework to coordinate agent exploration and shows that coordinated exploration activities of agents can lead to faster learning convergence.

Table IV. CLASSIFICATION BASED ON LEARNING TECHNIQUE

| Learning Technique | # Studies | Percentage |
|---|---|---|
| Reinforcement Learning | 31 | 59,62% |
| Game Theory | 5 | 9,62% |
| Supervised Learning | 5 | 9.62% |
| Probabilistic | 4 | 7.69% |
| Statistics | 2 | 3.85% |
| Swarm System | 2 | 3,85% |
| Applied Logic | 1 | 1.92% |
| Evolutionary Process | 1 | 1.92% |
| Game Theory and RL | 1 | 1.92% |

When CSAS agents are homogeneous (i.e., they perform the same learning task) building a fully observable [29] or a hierarchically observable context [69], in situations where agents have partial knowledge access to the environment and their peers, helps to speed up decentralised learning tasks. This context can be a collective memory (i.e. an environment holding a centralised socially-shared memory) which serves as a blackboard for all the agents (analogous to a shared past) [62]. Even though this centralisation of knowledge may be a viable approach for some CSAS, this is not always the case. Hence, an interesting research area for CSAS using RL is investigating in-depth the different levels of knowledge access and their role in improving the speed of convergence of learning tasks, considering their practical applicability in target CSAS applications.

*Game Theory (GT)* is used in $\sim 10\%$ of the reviewed studies. In [63], a consensus algorithm is introduced to establish cooperation among the agents. Similarly, [49] promotes emergent coordination among agents using a game theoretical approach based on an incentivisation mechanism that assigns penalties through a dynamic payoff matrix. A GT-based approach is also used in [61], by means of evolutionary and social learning, to promote cooperative behaviour among distributed agents and solve an N-player prisoner's dilemma. Finally, GT is combined with RL in [41] to facilitate cooperation among agents in a distributed wireless sensor network.

Our survey shows that GT approaches are mainly used in application domains requiring collective and coordinated actions by the agents. However, we highlight that some of the considered domains lack real-world applicability (e.g., [61]). A research area that needs further investigation is how to promote and rely on collective and collaborative actions in an open network of (untrusted) agents. In fact, as shown in Section III-A4, incentivisation mechanisms promoting collaboration, which address the free-rider problem in a decentralised network of selfish agents [80], are rarely used.

*Supervised Learning (SL)* techniques are applied in $\sim 10\%$ of the studies. In [42], support vector data description (SVDD), a variant of support vector machines (SVM), is used to solve a system-wide energy optimisation problem in a distributed way. The proposed trust- and cooperation-based algorithm is trained offline with a big set of previously observed values. The authors claim that their approach can be applied to generic self-organising hierarchical system structure domains. SVM is used in [44] to regulate the behaviour of prosumers in a smart energy scenario (i.e., produce or consume). In [52], different supervised learning approaches (i.e., neural network, support vector regression, and regression decision trees) are used for modelling the performance of total order broadcast protocols.

Most of the reviewed studies in this group, guide the *online* agents' behaviour based on a model that has been trained *offline* using SL techniques. A recurrent challenge for offline model training, e.g., [81], is identifying *when*, i.e., determine effective stopping criteria, that enable an agent to generalise from the training data to unseen situations while reducing the risk of overfitting [82]. We did not find any study based on SL dealing with this problem in CSAS.

Our review highlights that $\sim 8\%$ of the studies uses *probabilistic* approaches (e.g., Bayesian Dynamic Linear Models [39], Bayesian Learning [28], [50] and Gaussian Mixture Models [34]), while two other studies use *applied logic* [76] and *evolutionary processes* [73]. Finally, *swarm systems* are used for collective self-organisation in two studies [40], [65].

In conclusion, learning techniques range from fit-for-purpose statistics-based approaches to more sophisticated concepts, e.g., combining RL with neural networks to make informed decisions [47]. Some approaches use additional (external) information to guide the learning process (e.g. detect mutual information [83] or build trust relationships [84], and augment the condition part of RL). Despite its high relevance, existing CSAS solutions neglect privacy-aware global modelling/learning concepts such as federated learning [85].

### D. RQ4: Triggers for Model Learning and Refinement

Beyond the adopted learning techniques, the process employed for learning new or updating existing models is an equally important factor affecting the ability of agents within CSAS to operate in uncertain environments. The distributed nature of most CSAS solutions entails that revising learnt models by accessing and analysing knowledge while agents operate might be inhibited due to physical limitations or reduced computational resources. Thus, the successful operation of CSAS depends also on determining effective **triggers** that enable learning new or revising existing agents' models.

Our analysis of the selected CSAS studies targeted the identification of *when* gathered knowledge is initially exploited and *when* learning models are updated. Table V summarises our analysis. As expected, the initial trigger that enables building the first version of learning models is predominantly associated with launching the CSAS solution to perform its learning task (cf. Section III-B). The majority of the reviewed studies initialises these models randomly, i.e., without any initial knowledge. This is an effective means of reducing bias and enabling effective evaluation of the CSAS solution. When peers have some initial knowledge, which is accessible to other agents, this could trigger the instantiation of learning models driven by knowledge extracted from peers [26], [48], [71]. Other studies (e.g., [62], [72], [75]) use domain knowledge for setting boundaries on what agents can learn at runtime and for accelerating learning, while in [28] the initial trigger and knowledge are provided through human demonstration.

Table V. CLASSIFICATION BASED ON TRIGGERS

| **Initial Trigger** | **# Studies** | **Percentage** |
|---|---|---|
| No initial knowledge (random) | 33 | 63.46% |
| Not mentioned | 9 | 17.31% |
| Domain knowledge / humans | 7 | 13.46% |
| From peers and other agents | 3 | 5.77% |
| **Trigger Update** | **# Studies** | **Percentage** |
| Periodic | 14 | 26.92% |
| Action (load/message/decision/step) | 13 | 25.00% |
| Learning task threshold achieved | 8 | 15.38% |
| Task/Episode | 7 | 13.46% |
| Not mentioned | 6 | 11.54% |
| Social interaction | 4 | 7.69% |

Triggers for refining learnt models reveal a more balanced distribution that can be partitioned further into time-based and event-based triggers. Time-based triggers represent $\sim 27\%$ of the reviewed studies and activate model refinement periodically (i.e., at regular intervals). This group assumes that sufficient knowledge has been gathered, the analysis of which enables effective model refinement. In contrast, event-based triggers can be partitioned further considering the type of event. These triggers range from action-based refinement (i.e., upon making a decision [28], or performing an action [67], [69]) to achieving a learning-task specific threshold [44], [56] and from episodic updates [33], [59], [60], [76] to refinements upon interacting with peers [27], [37], [50], [51].

In summary, the trigger for instantiating the initial learning model typically occurs upon launching a new learning task, whereas we observe an equal spread between time-based and event-based triggers for updating the models. Selecting effective update triggers capable of improving model accuracy and driving efficient achievement of the CSAS objectives remains an open challenge. In fact, this challenge resembles the exploration-exploitation dilemma in RL [79].

## IV. A 3D Framework of CSAS

The analysis of the selected literature, focusing on CSAS applications with decentralised learning, enables us to identify the key characteristics of the learning process in CSAS agents. These characteristics are the basis for a framework comprising three *dimensions*. Similarly to the framework proposed in [86], which is in the context of norms in MAS, we explore CSAS with varying combinations of dimension values. We discuss the impact of dimensions on the learning process and highlight application classes that can be investigated in future research.

### A. Framework Dimensions

Figure 3 depicts the proposed three-dimensional framework. Driven by the investigated data items (Section II-C), we identified characteristics of CSAS agents that influence design

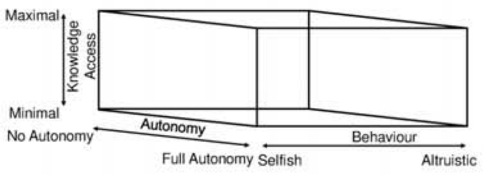

Fig. 3. The 3D framework for CSAS

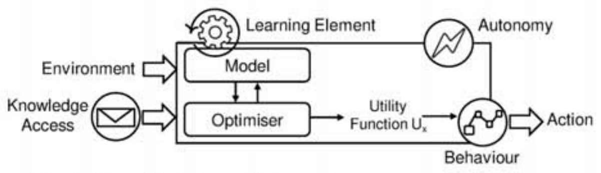

Fig. 4. Canonical view of the learning process of a CSAS agent

decisions on how agents can learn. This led us to the following dimensions: *autonomy*, *knowledge access*, and *behaviour*. Each dimension is orthogonal to the others, thus forming a continuous 3D space in which agents can be located.

**Autonomy**, the first dimension, refers to the ability of an agent to have full control over its own state and behaviour. When an *autonomous* agent learns, it has complete autonomy over its learning process, meaning that it is responsible for creating learning models, adjusting parameters, etc. In the other extreme, agents with *no autonomy* have their learning model updated externally. This can be done by a peer or supervisory agent that undertakes the model updating activity and the agent uses the updated model (upon receiving it) to guide its actions. Although this compromises the standard definition of an agent, it can be used in particular situations, e.g., to build a learning model with more data than what the agent can access. *Restricted autonomy* corresponds to CSAS instances in which external entities have only partial direct influence on an agent's process for updating its learning model.

The second dimension is **knowledge access**. Although CSAS agents are located in and can perceive an environment, the degree of perception can range from a limited part of the environment, where the agent is situated, to the whole environment. Moreover, agents can collaborate, which entails that an agent may have access not only to the information it perceives from the environment, but also to knowledge provided by other agents, including their internal information. Hence, our notion of knowledge access also includes the degree of information sharing between agents. However, our focus is on the value of information available to the agent and not the means used for obtaining the information. Knowledge access thus ranges from *minimal knowledge*, in which agents do not exchange any information, to *maximal knowledge*, in which information is fully shared among agents leading to a complete perception of the environment.

Finally, the third dimension is **behaviour**. A CSAS agent is *selfish* when its behaviour is driven by its own goals, rather than the global system (or society) goals. Therefore, if an agent is selfish, it aims to maximise its own utility while learning. In contrast, if an agent is *altruistic*, it has the goal of maximising the utility of the system as a whole or its society. Note that a selfish agent is not necessarily harmful or malicious. Also, a selfish agent might cooperate with other agents if adopting this behaviour is useful for the agent to achieve its own goals.

These dimensions provide a clear understanding of the assumptions that can be made about agents that are part of CSAS. The dimensions also help to make design decisions on how agents should learn. Having a good understanding of the implications associated with specific dimension values enables to provide learning solutions that are applicable to a class of CSAS with similar characteristics. For example, if knowledge access is minimal, all observations from the environment are made by the agent itself and, thus, the agent must explore its environment sufficiently to learn. Implications on learning can also be connected with multiple dimensions. For instance, in a CSAS comprising selfish and autonomous agents with minimal knowledge access, the adopted learning algorithms should converge fast because no information is shared and agents want to maximise their own utility. The investigated studies provide limited information for learning-related design decisions and do not permit deriving founded learning implications. Our framework consolidates knowledge regarding the learning aspects of CSAS and aims to encourage the development of such a body of knowledge.

Next, we describe a canonical view [87] of the learning process of individual agents, aiming to provide additional insights regarding possible links between our 3D framework and decentralised learning in CSAS.

### *B. Canonical View of the Learning Process of a CSAS Agent*

A learning-based CSAS agent is designed to improve its ability to fulfil the learning task. Irrespective of the CSAS characteristics analysed in Section III, i.e., learning purpose, learning techniques and triggers used for model learning and refinement, each agent within its self-adaptation loop (e.g., MAPE-K loop [8]) follows a standard learning process.

Figure 4 illustrates a canonical view of the learning process in such a self-adaptation loop. The learning element takes environmental and internal observations as inputs and generates an action (leading to CSAS reconfiguration) as output. Internally, a utility function $U_x$ is employed to characterise the desirability of the agent's behaviour [88]. Furthermore, the learning element contains a model (e.g., a classifier) and an optimisation (or controller) component that decides about the actions optimising the employed utility function.

The three dimensions of the introduced framework (Section IV-A) support the canonical view as follows: (1) Depending on the level of *autonomy*, the agent's decisions may be overruled, changed, or blocked by external intervention; (2) Depending on the level of *knowledge access*, information from other agents may be available in addition to the local view of internal and environmental information; (3) Depending on the *behaviour* type, the definition of a utility function $U_x$ may vary between favouring agent and/or system-wide utility.

A CSAS agent can achieve the ability to fulfil the learning task at *design-time*, at *runtime*, or in a *hybrid* manner. Learning at *design-time* requires a large amount of sample data to train a classifier or a regression model using supervised or unsupervised learning techniques (e.g., [89]). The produced knowledge (i.e., the pre-trained model) is exploited at runtime to steer the CSAS. To learn at *runtime*, a utility function continuously assesses the merits of the CSAS based on the obtained utility values. The employed learning model is incrementally refined by exploiting the runtime data. Such data is collected through the adopted learning technique (e.g., RL [56]) and peer-provided knowledge (depending on the level of knowledge access). A *hybrid* learning scheme uses pre-collected sample data to train a learning model at design-time and uses runtime data to refine the model (e.g. [90]).

Refinements of the learning model (taking the features of the learning task into account) fall in one of the following combinations (listed in order of increasing complexity).

1) *The learning task and the learning model both have static features*. This results in the design-time selection of a pre-defined configuration. Such characteristics disable the CSAS agent to react to unanticipated events and do not allow for runtime learning. In this setting, the CSAS agent learns only at design time by employing only sample data while excluding any runtime information [64].
2) *The learning model is dynamic, and the learning task has static features*. This allows for updating the learning model at runtime in response to knowledge collected (at unexpected times) from the environment or agent's peers (e.g., the decision boundary in a classifier). Learning techniques (e.g., RL) are used at runtime to update the model [29], [30]. This is the most prominent learning technique in the analysed studies (cf. Section III).
3) *The learning task has dynamic features, and the model is static*. This setting can be the result of varying availability of input information (e.g., changing sensor patterns). This setting requires either a set of pre-trained models [52], switching quickly between them at runtime, or advanced techniques to interpret the model using limited input.
4) *Both the learning task and the model have dynamic features*. These features can be the result of runtime activation, addition, deactivation or removals of sensors (even intentionally as in [91], [92]). In this setting, the dynamic model refers to runtime consideration of information (e.g., through RL).

### C. Example Applications

After introducing the 3D framework and analysing the impact of its dimensions on the learning process of individual agents, we present in Table VI examples corresponding to CSAS applications located on the corners of our 3D framework. These examples represent classes of applications that can be investigated to address learning-specific challenges, taking into account design decisions such as those exemplified in Section IV-A. In our systematic analysis, we observed the availability of applications in most of the extreme cases (corners) of our 3D framework. Figure 5 shows a scatter plot indicating how the CSAS applications are spread over the dimensions of the proposed 3D framework. Agents are typically considered selfish while varying levels of knowledge access

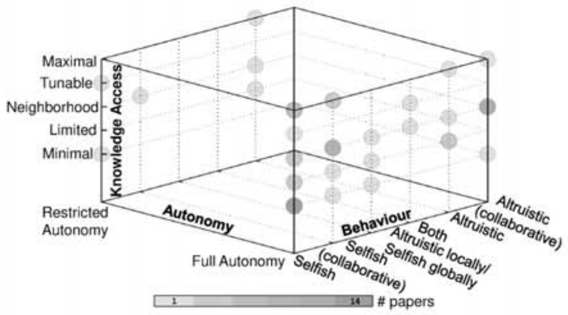

Fig. 5. Analysed applications within the 3D framework.

are commonly explored. There is, however, limited work exploring agents with restricted autonomy. We only found six studies, located in corners, where agents have restricted autonomy. There is no study exploring selfish agents with maximal knowledge access or altruistic agents with minimal knowledge access. The lack of research in these areas of our 3D framework is interesting since these settings can naturally occur in real-world applications (cf. Table VI).

Finally, we highlight the importance of studying intermediate cases or cases where a CSAS comprises agents situated in different areas of our 3D framework. For instance, a CSAS may comprise multiple societies and the relationship among agents within a society is different from the relationship between societies (correspondingly, different to agents in other societies). Using the proposed framework, the impact of potential (runtime) variation in a dimension can also be evaluated. This is the case in recent research exploring variation in *behaviour* [42], [59] and *knowledge access* [42], [45], [69].

**Threats to Validity.** We limit *construct validity threats* due to the adopted multi-stage search and selection process by following protocols used in similar studies [24], [25]. We limit *internal validity threats* that could produce incorrect analysis data and lead to constructing a biased 3D framework by having each selected paper analysed by at least two reviewers. We mitigate *external validity threats* that could limit the generalisation of our findings by investigating 52 CSAS studies from 14 relevant venues. Extending the search string and running automatic searches on entire scientific databases will enable to evaluate further the generality of the 3D framework.

## V. Implications and Open Challenges

The dimensions of the 3D framework have implications for the effectiveness of adopted learning techniques. For instance, using a cooperative learning algorithm in a CSAS of selfish agents shall result in slow or no convergence, requirement for trust models and inefficiencies, i.e., unnecessary communication and computation cost. Similarly, learning techniques for autonomous CSAS should be designed to be robust against malicious nodes and free riders. Active research combining distributed consensus mechanisms in blockchain technologies and artificial intelligence aims to address this challenge [93].

Significant implications are also observed when moving to higher levels of knowledge access. Minimal knowledge access promotes autonomy and imposes learning via the

Table VI. EXAMPLE APPLICATIONS LOCATED IN THE CORNERS OF THE 3D FRAMEWORK

| Autonomy | Behaviour | Knowledge | Application Description |
|---|---|---|---|
| Full Autonomy | Selfish | Minimal | Autonomous cars deciding which route to take based on individual goals and *only* observing neighbours. |
| | | Maximal | Autonomous cars sharing information about destination, speed, etc. but each car pursues its own goals. |
| | Altruistic | Minimal | Software components processing data stream owned by a single provider, (restricted knowledge access due to bandwidth/scalability). |
| | | Maximal | Traffic lights aiming to optimise the traffic flow taking other traffic lights into account (e.g., signalisation, control of green light duration, routing, coordination). |
| No Autonomy | Selfish | Minimal | Bike sharing system with agents acting on behalf of humans, utilising the bike sharing stations. |
| | | Maximal | Self-adapting software components (considering other components data) that contribute to a global behaviour (e.g. self-healing) through prioritising their own goals. |
| | Altruistic | Minimal | Car sharing system where software controls cars to serve other users, maximising the interest of car owners. |
| | | Maximal | Robots jointly monitoring and splitting an area for distributed tasks such as surveillance, cleaning, etc. |

agents' environment that is feasible, though expensive, using general-purpose RL techniques (e.g., Q-learning). However, models easily become outdated when the environment changes radically. In time-critical applications that can experience unexpected events, limited knowledge access can result in significant inefficiencies and risks. In contrast, maximal knowledge access provides a larger spectrum of learning techniques (cf. Section III-C). However, this is not the norm in distributed systems such as CSAS. Robust CSAS usually have by design some partial knowledge access (limited or neighbourhood on our 3D framework). This entails using more complex learning techniques, specialised in coordination processes between agents and collective intelligence. Recent work towards this direction can be found in [13], [94], [95].

Our findings can help researchers to make design choices when developing CSAS with learning capabilities. For instance, consider an application scenario that requires minimal knowledge access between agents (e.g., privacy policies). This requirement acts as a filter within the 3D framework that enables choosing among autonomy and behaviour options. As such, the 3D framework highlights the current state of practice that meets this requirement and provides design intuition based on systematically collected and analysed empirical evidence.

### A. Resilience and Fault-tolerance

CSAS are typically employed in CPS and network application domains (cf. Section III-A) that frequently experience several uncertainties such as network latency, component failures and limited resources, i.e., battery lifetime [17], [19]. Several applications discussed in our study fall in this class, i.e., traffic systems, sensor networks, smart grids etc. These uncertainty types can disrupt the learning process, limit the data required for training or even invalidate data on which learning is performed. Self-organisation is often proposed as the means to cope with such uncertainties. It can come, however, with prohibitive computational and communication costs as it usually requires continuous proactive agent interactions. It also perplexes the design of new learning techniques, i.e., learning to self-organise vs. self-organise via learning, that is the subject of active research [96], [97]. An alternative approach is the introduction of learning mechanisms that are by design self-adaptive to network uncertainties. Self-adaptation may refer to the localisation of the learning process, applying the concept of dropout in the communication network to improve performance, and balance exploration and exploitation [92].

### B. Privacy and Accountability

Learning with personal and privacy-sensitive data poses several challenges and threats [98]. Privacy violation, profiling actions over users' activities can undermine users' trust in learning systems, and enable discriminatory data analytics and nontransparent recommender systems [99]. Recent research in the area introduces novel learning algorithms that employ privacy-preserving techniques such as differential privacy [100], homomorphic encryption [101] and secure data management via distributed ledgers [102]. CSAS with limited/neighbourhood-style knowledge access are by design more effective in privacy-preservation as data is not aggregated in single locations. Since CSAS management is diffused among all agents, it is computationally harder for single agents to manipulate the system as a whole. In contrast, holding an agent accountable for specific CSAS is not straightforward and democratic governance mechanisms could be useful [103].

## VI. Conclusion

Our systematic review of learning-enabled CSAS shows that behavioural and collaboration modalities are perplexed when learning is required. This reveals that learning tasks, emergent behaviour, learning techniques and triggers play a key role when designing learning-enabled CSAS. Minimal knowledge access, high autonomy and the prevalence of reinforcement learning are some key characteristics that we observe, together with their use in CPS. Based on these findings, we introduce a 3D framework capturing the characteristics of learning-enabled CSAS. Given the dimensions of *autonomy*, *knowledge access* and *behaviour*, we present a canonical view of the learning process with which applications and learning implications can be classified, and discuss how these implications are different to those in centralised-controlled CSAS. Open challenges such as the design of collaborative learning techniques, privacy and accountability in open environments without trusted third parties, the resilience, fault-tolerance and provision of assurances [104], [105] of learning in distributed environments are highlighted as opportunities for future work.

## Acknowledgement

This study was financed in part by CAPES - Brasil - Finance Code 001 and by the Swedish Knowledge Foundation, Grant No. 20150088: *Software Technology for Self-Adaptive Systems*.

## References

[1] V. Cardellini, M. D'Angelo, V. Grassi, M. Marzolla, and R. Mirandola, "A decentralized approach to network-aware service composition," in *European Conference on Service-Oriented and Cloud Computing*. Springer, 2015, pp. 34–48.

[2] M. U. Iftikhar, G. S. Ramachandran, P. Bollansée, D. Weyns, and D. Hughes, "Deltaiot: A self-adaptive internet of things exemplar," in *SEAMS'17*, May 2017, pp. 76–82.

[3] H. Prothmann, S. Tomforde, J. Branke, J. Hähner, C. Müller-Schloer, and H. Schmeck, "Organic Traffic Control," in *Organic Computing'11*, 2011, pp. 431–446.

[4] M. Bojarski *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[5] C. Müller-Schloer and S. Tomforde, *Organic Computing – Techncial Systems for Survival in the Real World*, 2017.

[6] J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[7] R. Calinescu, M. Autili, J. Cámara, A. Di Marco, S. Gerasimou *et al.*, "Synthesis and verification of self-aware computing systems," in *Self-Aware Computing Systems*. Springer, 2017, pp. 337–373.

[8] R. De Lemos *et al.*, "Software engineering for self-adaptive systems: A second research roadmap," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 1–32.

[9] R. Calinescu, S. Gerasimou, and A. Banks, "Self-adaptive software with decentralised control loops," in *FASE'15*, 2015, pp. 235–251.

[10] D. Weyns *et al.*, "On patterns for decentralized control in self-adaptive systems," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 76–107.

[11] B. H. C. Cheng *et al.*, *Software Engineering for Self-Adaptive Systems: A Research Roadmap*. Springer, 2009, pp. 1–26.

[12] S. Tomforde *et al.*, "Observation and Control of Organic Systems," in *Organic Computing'11*, 2011, pp. 325 – 338.

[13] E. Pournaras, P. Pilgerstorfer, and T. Asikis, "Decentralized collective learning for self-managed sharing economies," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 13, no. 2, p. 10, 2018.

[14] M. Caporuscio, M. D'Angelo, V. Grassi, and R. Mirandola, "Reinforcement learning techniques for decentralized self-adaptive service assembly," in *Service-Oriented and Cloud Computing*, 2016, pp.53–68.

[15] D. Weyns, S. Malek, and J. Andersson, "On decentralized self-adaptation: lessons from the trenches and challenges for the future," in *SEAMS'10*. ACM, 2010, pp. 84–93.

[16] N. R. Jennings, "An agent-based approach for building complex software systems," *CACM*, vol. 44, no. 4, pp. 35–41, Apr. 2001.

[17] D. Ye, M. Zhang, and A. V. Vasilakos, "A survey of self-organization mechanisms in multiagent systems." *IEEE Trans. Systems, Man, and Cybernetics: Systems*, vol. 47, no. 3, pp. 441–461, 2017.

[18] A. Diaconescu, K. L. Bellman, L. Esterle, H. Giese *et al.*, "Architectures for collective self-aware computing systems," in *Self-Aware Computing Systems*. Springer, 2017, pp. 191–235.

[19] M. Mitchell, "Self-awareness and control in decentralized systems." in *AAAI'05*, 2005, pp. 80–85.

[20] J. O. Kephart, A. Diaconescu, H. Giese, A. Robertsson *et al.*, "Self-adaptation in collective self-aware computing systems," in *Self-Aware Computing Systems*. Springer, 2017, pp. 401–435.

[21] A. Rodrigues, R. D. Caldas, G. N. Rodrigues, T. Vogel, and P. Pelliccione, "A learning approach to enhance assurances for real-time self-adaptive systems," in *SEAMS'18*. ACM, 2018, pp. 206–216.

[22] P. J. t. Hoen, K. Tuyls, L. Panait, S. Luke, and J. A. La Poutré, "An overview of cooperative and competitive multiagent learning," in *LAMAS'05*. Springer, 2006, pp. 1–46.

[23] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, Nov. 2005.

[24] B. A. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele University, Technical Report EBSE-2007-01, 2007.

[25] H. Muccini, M. Sharaf, and D. Weyns, "Self-adaptation for cyber-physical systems: A systematic literature review," in *SEAMS'16*, 2016, pp. 75–81.

[26] L. Kraemer and B. Banerjee, "Reinforcement learning of informed initial policies for decentralized planning," *ACM Trans. Auton. Adapt. Syst.*, vol. 9, no. 4, pp. 18:1–18:32, Dec. 2014.

[27] W. Richert and B. Kleinjohann, "Adaptivity at every layer: A modular approach for evolving societies of learning autonomous systems," in *SEAMS'08*. ACM, 2008, pp. 113–120.

[28] A. Valtazanos and S. Ramamoorthy, "Bayesian interaction shaping: Learning to influence strategic interactions in mixed robotic domains," in *AAMAS'13*, 2013, pp. 63–70.

[29] S. Kalyanakrishnan and P. Stone, "Batch reinforcement learning in a complex domain," in *AAMAS'07*. ACM, 2007, pp. 94:1–94:8.

[30] K. Tumer, A. K. Agogino, and D. H. Wolpert, "Learning sequences of actions in collectives of autonomous agents," in *AAMAS'02*. ACM, 2002, pp. 378–385.

[31] M. Hölzl and T. Gabor, "Continuous collaboration: A case study on the development of an adaptive cyber-physical system," in *SEsCPS'15 Workshop*. IEEE Press, 2015, pp. 19–25.

[32] M. Beetz, A. Kirsch, and A. Muller, "Rpllearn: Extending an autonomous robot control language to perform," in *AAMAS'04*. IEEE, 2004, pp. 1022–1029.

[33] D. Szer and F. Charpillet, "Coordination through mutual notification in cooperative multiagent reinforcement learning," in *AAMAS'04*. IEEE, 2004, pp. 1254–1255.

[34] N. Bicocchi, M. Mamei, A. Prati, R. Cucchiara, and F. Zambonelli, "Pervasive self-learning with multi-modal distributed sensors," in *SASO'08*, 2008, pp. 61–66.

[35] P. R. Lewis, L. Esterle, A. Chandra, B. Rinner, and X. Yao, "Learning to be different: Heterogeneity and efficiency in distributed smart camera networks," in *SASO'13*, 2013, pp. 209–218.

[36] S. Rudolph, S. Tomforde, B. Sick, and J. Hähner, "A mutual influence detection algorithm for systems with local performance measurement," in *SEAMS'15*, Sept 2015, pp. 144–149.

[37] W. T. L. Teacy *et al.*, "Decentralized bayesian reinforcement learning for online agent collaboration," in *AAMAS '12*, 2012, pp. 417–424.

[38] L. Esterle, B. Rinner, and P. R. Lewis, "Self-organising zooms for decentralised redundancy management in visual sensor networks," in *SASO'15*, 2015, pp. 41–50.

[39] L. Fang and S. Dobson, "Towards data-centric control of sensor networks through bayesian dynamic linear modelling," in *SASO'15*, 2015, pp. 61–70.

[40] P. R. Lewis *et al.*, "Static, dynamic, and adaptive heterogeneity in distributed smart camera networks," *ACM Trans. Auton. Adapt. Syst.*, vol. 10, no. 2, pp. 8:1–8:30, Jun. 2015.

[41] M. Mihaylov, Y. Le Borgne, K. Tuyls, and A. Nowé, "Distributed cooperation in wireless sensor networks," in *AAMAS'11*, 2011, pp. 249–256.

[42] G. Anders, A. Schiendorfer, F. Siefert, J.-P. Steghöfer, and W. Reif, "Cooperative resource allocation in open systems of systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 10, no. 2, pp. 11:1–11:44, Jun. 2015.

[43] A. Marinescu, I. Dusparic, and S. Clarke, "Prediction-based multi-agent reinforcement learning in inherently non-stationary environments," *ACM Trans. Auton. Adapt. Syst.*, vol. 12, no. 2, pp. 9:1–9:23, 2017.

[44] P. E. Petruzzi, J. Pitt, and D. Busquets, "Inter-institutional social capital for self-organising 'nested enterprises'," in *SASO'16*, 2016, pp. 90–99.

[45] U. Montanari and A. T. Siwe, "Prosumers as aggregators in the dezent context of regenerative power production," in *SASO'14*, 2014, pp. 167–174.

[46] P. Vrancx, P. Gurzi, A. Rodriguez, K. Steenhaut, and A. Nowé, "A reinforcement learning approach for interdomain routing with link prices," *ACM Trans. Auton. Adapt. Syst.*, vol. 10, no. 1, pp. 5:1–5:26, Mar. 2015.

[47] E. Gelenbe, M. Gellman, R. Lent, P. Liu, and P. Su, "Autonomous smart routing for network qos," in *ICAC'04*, 2004, pp. 232–239.

[48] W. Galuba, K. Aberer, Z. Despotovic, and W. Kellerer, "Self-organized fault-tolerant routing in peer-to-peer overlays," in *SASO'09*, 2009, pp. 30–39.

[49] S. Wildermann, T. Ziermann, and J. Teich, "Self-organizing bandwidth sharing in priority-based medium access," in*SASO'09*,2009, pp. 144–153.

[50] B. Xu, O. Wolfson, and C. Naiman, "Machine learning in disruption-tolerant manets," *ACM Trans. Auton. Adapt. Syst.*, vol. 4, no. 4, pp. 23:1–23:36, Nov. 2009.

[51] Y.-H. Chang, T. Ho, and L. P. Kaelbling, "Mobilized ad-hoc networks: a reinforcement learning approach," in *ICAC'04*, 2004, pp. 240–247.

[52] M. Couceiro, P. Romano, and L. Rodrigues, "A machine learning approach to performance prediction of total order broadcast protocols," in *SASO'10*, 2010, pp. 184–193.

[53] D. Dorsey, B. J. Carandang, M. Kam, and C. Gaughan, "Self-adaptive dissemination of data in dynamic sensor networks," in *SASO'08*, 2008, pp. 380–389.

[54] K. Malialis, S. Devlin, and D. Kudenko, "Resource abstraction for reinforcement learning in multiagent congestion problems," in *AAMAS'16*, 2016, pp. 503–511.

[55] I. Dusparic and V. Cahill, "Distributed w-learning: Multi-policy optimization in self-organizing systems," in *SASO'09*, 2009, pp. 20–29.

[56] A. Stein, S. Tomforde, D. Rauh, and J. Hähner, "Dealing with unforeseen situations in the context of self-adaptive urban traffic control: How to bridge the gap," in *ICAC'16*, July 2016, pp. 167–172.

[57] S. Tomforde *et al.*, "Decentralised progressive signal systems for organic traffic control," in *SASO'08*, 2008, pp. 413–422.

[58] I. Dusparic and V. Cahill, "Autonomic multi-policy optimization in pervasive systems: Overview and evaluation," *ACM Trans. Auton. Adapt. Syst.*, vol. 7, no. 1, pp. 11:1–11:25, May 2012.

[59] J. Hao, H.-F. Leung, and Z. Ming, "Multiagent reinforcement social learning toward coordination in cooperative multiagent systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 9, no. 4, pp. 20:1–20:20, Dec. 2014.

[60] J. Hao, J. Sun, G. Chen, Z. Wang, C. Yu, and Z. Ming, "Efficient and robust emergence of norms through heuristic collective learning," *ACM Trans. Auton. Adapt. Syst.*, vol. 12, no. 4, pp. 23:1–23:20, Oct. 2017.

[61] R. Chiong and M. Kirley, "Co-evolution of agent strategies in n-player dilemmas," in *AAMAS'10*, 2010, pp. 1591–1592.

[62] X.-F. Xie and J. Liu, "How autonomy oriented computing (aoc) tackles a computationally hard optimization problem," in *AAMAS'06*. ACM, 2006, pp. 646–653.

[63] W. Lorkiewicz, R. Kowalczyk, R. Katarzyniak, and Q. B. Vo, "On topic selection strategies in multi-agent naming game," in *AAMAS'11*, 2011, pp. 499–506.

[64] N. K. Altshuler and D. Sarne, "Modeling assistant's autonomy constraints as a means for improving autonomous assistant-agent design," in *AAMAS'18*, 2018, pp. 1468–1476.

[65] J. Jumadinova and P. Dasgupta, "Firefly-inspired synchronization for improved dynamic pricing in online markets," in *SASO'08*, 2008, pp. 403–412.

[66] S. Das, B. Grosz, and A. Pfeffer, "Learning and decision: Making for intention reconciliation," in *AAMAS'02*. ACM, 2002, pp. 1121–1128.

[67] S. Abdallah and V. Lesser, "Multiagent reinforcement learning and self-organization in a network of agents," in *AAMAS'07*. ACM, 2007, pp. 39:1–39:8.

[68] D. Ye, M. Zhang, and D. Sutanto, "Self-organisation in an agent network via learning," in *AAMAS'10*, 2010, pp. 1495–1496.

[69] C. Zhang, V. Lesser, and S. Abdallah, "Self-organization for coordinating decentralized reinforcement learning," in *AAMAS'10*, 2010, pp. 739–746.

[70] G. Yang and V. Danos, "Learning in open adaptive networks," in *SASO'16*, 2016, pp. 50–59.

[71] M. A. M. d. Oca, T. Stuetzle, M. Birattari, and M. Dorigo, "Incremental social learning applied to a decentralized decision-making mechanism: Collective learning made faster," in *SASO'10*, 2010, pp. 243–252.

[72] P. Scerri, D. Pynadath, and M. Tambe, "Adjustable autonomy in real-world multi-agent environments," in *AGENTS'01*, 2001, pp. 300–307.

[73] Gurcan, C. Bernon, K. S. Turker, J. Mano, P. Glize, and O. Dikenelli, "Simulating human single motor units using self-organizing agents," in *SASO'12*, 2012, pp. 11–20.

[74] R. Kota, N. Gibbins, and N. R. Jennings, "Decentralized approaches for self-adaptation in agent organizations," *ACM Trans. Auton. Adapt. Syst.*, vol. 7, no. 1, pp. 1:1–1:28, May 2012.

[75] S. Ontañón and E. Plaza, "A bartering approach to improve multiagent learning," in *AAMAS'02*. ACM, 2002, pp. 386–393.

[76] S. Wang and K. Hu, "Towards dynamic epistemic learning of actions in autonomic multi-agent systems," in *ICAC'16*, 2016, pp. 237–238.

[77] Y. Miyashita, M. Hayano, and T. Sugawara, "Self-organizational reciprocal agents for conflict avoidance in allocation problems," in *SASO'15*, 2015, pp. 150–155.

[78] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *AAAI'98*, 1998, pp. 746–752.

[79] M. Humphrys, "Action selection methods using reinforcement learning," pp. 135–144, 1996.

[80] C. Muldoon, M. J. O'Grady, and G. M. O'Hare, "A survey of incentive engineering for crowdsourcing," *The Knowledge Engineering Review*, vol. 33, 2018.

[81] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[82] S. Whiteson, B. Tanner, M. E. Taylor, and P. Stone, "Protecting against evaluation overfitting in empirical reinforcement learning," in *ADPRL'11*, 2011, pp. 120–127.

[83] S. Rudolph, S. Tomforde, B. Sick, and J. Hähner, "A mutual influence detection algorithm for systems with local performance measurement," in *SASO'15*, 2015, pp. 144–149.

[84] J. Kantert, C. Reinbold, S. Tomforde, and C. Müller-Schloer, "An evaluation of two trust-based autonomic grid computing systems for volunteer-based distributed rendering," in *ICAC'16*, 2016, pp. 137–146.

[85] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

[86] M. Luck, S. Munroe, R. Ashri, and F. L. y Lopez, "Trust and norms for interaction," in *2004 IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, 2004, pp. 1944–1949 vol.2.

[87] E. Mezuman and Y. Weiss, "Learning about canonical views from internet image collections," in *NIPS'12*, 2012, pp. 719–727.

[88] S. Ghahremani, H. Giese, and T. Vogel, "Efficient utility-driven self-healing employing adaptation rules for large dynamic architectures," in *ICAC'17*, pp. 59–68.

[89] S. Ghahremani, C. M. Adriano, and H. Giese, "Training prediction models for rule-based self-adaptive systems," in *ICAC'18*, pp. 187–192.

[90] M. Sommer, S. Tomforde, J. Hähner, and D. Auer, "Learning a dynamic re-combination strategy of forecast techniques at runtime," in *ICAC'15*, 2015, pp. 261–266.

[91] M. Jänicke, B. Sick, and S. Tomforde, "Self-adaptive multi-sensor activity recognition systems based on gaussian mixture models," *Informatics*, vol. 5, no. 3, p. 38, 2018.

[92] E. Pournaras, S. Yadhunathan, and A. Diaconescu, "Holarchic structures for decentralized deep learning-a performance analysis," *Cluster Computing*, 2019.

[93] S. Dey, "A proof of work: Securing majority-attack in blockchain using machine learning and algorithmic game theory," *International Journal of Wireless and Microwave Technologies*, vol. 8, no. 5, pp. 1–9, 2018.

[94] C. Hinrichs and M. Sonnenschein, "A distributed combinatorial optimisation heuristic for the scheduling of energy resources represented by self-interested agents," *International Journal of Bio-Inspired Computation*, vol. 10, no. 2, pp. 69–78, 2017.

[95] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip learning with linear models on fully distributed data," *Concurrency and Computation: Practice and Experience*, vol. 25, no. 4, pp. 556–571, 2013.

[96] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *NIPS'16*, 2016, pp. 2137–2145.

[97] J. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, and I. Mordatch, "Learning with opponent-learning awareness," in *AAMAS'18*, 2018, pp. 122–130.

[98] D. Helbing and E. Pournaras, "Society: Build digital democracy," *Nature News*, vol. 527, no. 7576, p. 33, 2015.

[99] N. Helberger, K. Karppinen, and L. D'Acunto, "Exposure diversity as a design principle for recommender systems," *Information, Communication & Society*, vol. 21, no. 2, pp. 191–207, 2018.

[100] T. Asikis and E. Pournaras, "Optimization of privacy-utility trade-offs under informational self-determination," *Future Generation Computer Systems*, 2018.

[101] H. Takabi, E. Hesamifard, and M. Ghasemi, "Privacy preserving multi-party machine learning with homomorphic encryption," in *NIPS'16*, 2016.

[102] V. Torra and G. Navarro-Arribas, "Probabilistic metric spaces for privacy by design machine learning algorithms: Modeling database changes," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer, 2018, pp. 422–430.

[103] T. John and M. Pam, "Complex adaptive blockchain governance," in *MATEC Web of Conferences*, vol. 223. EDP Sciences, 2018, p. 01010.

[104] R. Calinescu, D. Weyns, S. Gerasimou, M. U. Iftikhar, I. Habli, and T. Kelly, "Engineering trustworthy self-adaptive software with dynamic assurance cases," *IEEE TSE*, vol. 44, no. 11, pp. 1039–1069, 2018.

[105] D. Weyns, N. Bencomo, R. Calinescu, J. Camara, C. Ghezzi *et al.*, "Perpetual assurances for self-adaptive systems," in *Software Engineering for Self-Adaptive Systems III. Assurances*. Springer, 2017, pp. 31–63.