ISSN: 1692-7257 - Volume 1 – Number 45 - 2025

RCTA
Revista Colombiana de Tecnologías de Avanzada
UNIPAMPLONA

Digital Object Identifier: 10.24054/rcta.v1i45.3131

Software agents based on machine learning technique. An outlook from 2010 to 2023

Agentes de software basados en técnicas de aprendizaje automático. Perspectivas desde 2010 hasta 2023

MSc. Hipatia Cazares Alegría[1], Dr. Pablo Pico Valencia[2]

[1] ***Pontificia Universidad Católica del Ecuador,*** *Maestría en Tecnologías de la Información, Esmeraldas, Ecuador.*
[2] ***Universidad de Granada,*** *Departamentos de Lenguajes y Sistemas Informáticos, Granada, Spain.*

Correspondence: *pablo.pico@ugr.es*

Received: *september 04, 2024.* ***Accepted:*** *december 15, 2024.* ***Published:*** *january 01, 2025.*

How to cite: *H. Cazares Alegría and P. Pico Valencia, "Software agents based on machine learning technique. An outlook from 2010 to 2023", RCTA, vol. 1, no. 45, pp. 39–56, jan. 2025.*
Recovered from https://ojs.unipamplona.edu.co/index.php/rcta/article/view/3131

Abstract: This study aims to analyze the main theoretical and practical proposals in which software agents have been integrated with machine learning models to determine their scope in terms of intelligence, proactivity, collaboration and learning. For the development of this research, the methodology proposed by Kofod-Peterson was carried out. Applying the methodology, 55 studies were analyzed. The studies showed that in the interaction between software agents and machine learning, cooperative and collaborative processes have been widely used in the resolution of control problems and in the optimization of data in distributed scenarios such as home, games and telecommunication. It was also found that mostly reinforcement learning models were used compared to machine learning models because they contribute more significantly to cooperative task modeling, which is widely used in intelligent systems.

Keywords: machine learning, software agents, multi-agent system, artificial intelligence.

Resumen: Este estudio tiene como objetivo analizar las principales propuestas teóricas y prácticas en las que se han integrado agentes de software con modelos de aprendizaje automático para determinar su alcance en términos de inteligencia, proactividad, colaboración y aprendizaje. Para el desarrollo de esta investigación se usó la metodología propuesta por Kofod-Peterson. Se analizaron 55 estudios los cuales mostraron que, en la interacción entre agentes de software y aprendizaje automático, los procesos cooperativos y colaborativos se han utilizado ampliamente en la resolución de problemas de control y en la optimización de datos en escenarios distribuidos como el hogar, juegos y las telecomunicaciones. También se evidenció que se utilizaron principalmente modelos de aprendizaje por refuerzo en comparación con los modelos de aprendizaje automático porque contribuyen de manera más significativa al modelado cooperativo de tareas en sistemas inteligentes.

Palabras clave: aprendizaje automático, agente software, sistema multiagente, inteligencia artificial.

1. INTRODUCTION

Software agents and multi-agent systems (MASs) have been widely applied in recent years. Some of these applications have been theoretical proposals, while others have been implemented to solve real-world production problems in various scenarios, including health monitoring [1], online education management [2], intelligent environment control [3], evolutionary microeconomy [4], information management in social networks [5], and technology monitoring [6], among others.

In these applications, software agents have primarily implemented intelligence mechanisms based on rules, logic and predicates, deliberative components (i.e., beliefs, desires, and intentions), and ontologies. However, software agents have also enhanced these intelligence levels by incorporating Artificial Intelligence (AI) techniques such as Machine Learning [7]. This integration has enabled MASs to utilize mechanisms that allow machines to learn from large datasets, thereby increasing their effectiveness and adaptability.

Currently, MASs are one of the AI technologies used in developing reactive web-oriented systems, mobile apps, and embedded systems [8]. The inherent characteristics of software agents, such as proactivity, autonomy, collaboration, and intelligence, enable the development of cognitive systems compatible with applications in diverse scenarios like Industry 4.0 [9], smart cities [9], smart homes [10], smart hospitals [11], and smart universities [12], among others. These scenarios vary in complexity and require intelligence mechanisms at different scales. In this context, software agents play a crucial role and are increasingly being integrated into cloud, fog, and edge computing applications to proactively manage resources within Internet of Things (IoT) ecosystems.

Artificial intelligence (AI) is becoming an integral part of daily life through applications that leverage data-driven learning strategies. Platforms like Netflix, YouTube, Amazon, Spotify, Facebook, and Tripadvisor, among others, exemplify this trend. Generally, these systems utilize machine learning models based on regression, classification, or clustering algorithms to train automatic predictors. This enables them to perform actions with a high degree of consistency, such as product recommendations, customer segmentation, and price prediction, among other tasks. Therefore, software agents have embedded machine learning models to improve their level of intelligence and thus become pro-active entities with the ability to learn and make more accurate and coherent decisions, closer to those that a human being could reach.

The recent surge in integrating software agents with supervised, unsupervised, and reinforcement learning models—widely documented in the literature—has motivated a systematic literature review. This review aims to provide comprehensive insights into the scope of software agents and multi-agent systems (MASs) as addressed by data-driven learning models. By doing so, it reveals the knowledge domains in which these models have been applied, the tasks they have optimized, the learning algorithms that have proven most effective, and their role in supporting automatic decision-making in systems. Understanding these research experiences is crucial for the next generation of IoT, known as the Internet of Agents, to incorporate models that enhance IoT's intelligence and autonomy.

The objective of this study is to analyze the scope of agent-oriented technologies combined with supervised, unsupervised, and reinforcement learning models to identify best practices for developing intelligent systems. The methodology used for the systematic review process follows the guidelines proposed by Kofod-Petersen [13]. This approach provides comprehensive instructions for conducting literature reviews in the field of Computer Science, including the design of the study, the definition of a scientific search chain, recommendations for specialized documentary databases, and the establishment of inclusion and exclusion criteria for the retrieved studies.

This paper is organized into four sections. Section 2 describes the methodology used to conduct the systematic literature review. In this section, we formulate the research questions, explain the process for retrieving studies, and outline the criteria for selecting studies for analysis. Section 3 presents the results of the systematic review, providing answers to each of the research questions formulated in the methodology section. It also discusses the results in terms of the scope and effectiveness of machine learning-based agents in solving problems within intelligent environments. Finally, Section 4 presents the main conclusions and suggestions for future works.

2. THEORETICAL BASIS

Artificial Intelligence (AI) is defined as the field of computer science that studies intelligence in artificial entities. From an engineering perspective, it involves creating entities that exhibit intelligent behavior [14]. In essence, AI aims to develop systems that model human-like intelligent behaviors, such as communication, learning, collaboration, proactivity, and decision-making [15]. In this context, software agents and machine learning techniques play a crucial role in advancing AI and automating processes.

Robotics has benefited from software agents since its inception, as they enable systems to support inherent human characteristics such as autonomy, proactivity, and collaboration. While agents are not a new technique, they continue to be pivotal in creating intelligent environments, such as the Internet of Things (IoT) [16]. Typically, agent intelligence is implemented using methods based on logic and predicates, allowing for logical inference processes. However, in complex systems and emerging ecosystems with large volumes of data (big data), it is crucial for agents to model learning processes akin to the human brain. Consequently, artificial neural networks (not covered in this study) and machine learning models have gained prominence. This section describes in deep these two techniques: software agents and machine learning.

2.1. Machine learning

Machine learning (ML) is a strategy for information analysis that automates the construction of learning models. It is also a strategy focused on developing flexible software to adapt whenever new data are considered in a model. In general terms, there are three main types of automatic learning: supervised, unsupervised and reinforcement. They are outlined as follows.

2.1.1. Supervised learning

It is a subset of ML techniques whose goal is to build models that perform evidence-based prediction [17]. Supervised learning algorithms work with labelled data, attempting to find a hypothesis function that, given the input variables, assigns the appropriate output label [18]. These algorithms have a manually sorted corpus on which the algorithm performs two processes: finding the best parameters for the model and evaluating the level of reliability with those parameters. This phase is called learning or training phase. Subsequently, the trained model can make predictions from new data, thus evidencing the learning.

Supervised learning algorithms can be regression, classification and clustering algorithms [19]. A brief description of such algorithms is presented below in a way that helps to better understand the results of the study.

- **Support vector machine (SVM).** Algorithm used to perform the classification of problems in which there are different classes. The learning process consists of finding the optimal hyperplane of the separation in dimensional scales. This hyperplane is a line that divides a plane in two parts where each class is on one side of the plane (see Figure 1.a).

- **Naïve Bayes (NB).** This algorithm is based on probabilistic techniques that is used when there are multidimensional inputs that provide the probability distribution between two events. Although it is a very simple algorithm, it has competitive advantages providing better results than many of the other existing algorithms. Therefore, it is used in cases where more advanced classification methods are required. (see Figure 1.b).

- **K-nearest neighbors (KNN).** It is an algorithm that implements a supervised technique which assumes that there are similar entities in the neighborhood. This model allows to rank values by seeking the most similar data points (by proximity) and specifying a value of *k* that corresponds to the number of neighbors. (see Figure 1.c).

- **Decision tree (DT).** This algorithm identifies the most significant variable and the value that provides the best sets of the population. Decision trees start from the principle of creating a set of decisions in the form of a tree, so that the intermediate nodes represent solutions, and the final nodes determine the prediction of the model. (see Figure 1.d).

- **Random forest (RF).** It is an algorithm that implements a versatile ensemble learning method, that is, a group of weaker models (decision trees) are combined to form a powerful model capable of performing dimensional reduction methods. Each tree gives a classification (votes for a class) and the result

is the class with the highest number of votes in the whole forest formed. (see Figure 1.e).

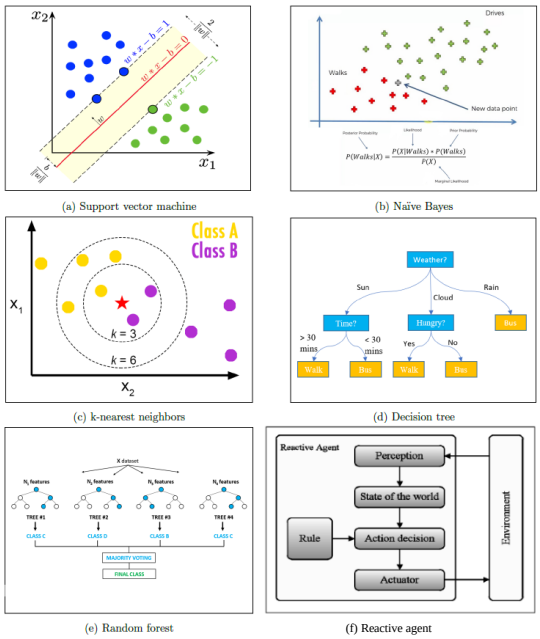

***Fig. 1**. Graphical representation of the main Machine Learning and architecture of a reactive agent.*
***Source:** own elaboration.*

2.1.2. Unsupervised learning

In this type of learning, unlike supervised learning, the algorithm is not specified what the output should be; that is, there is no defined relationship between the inputs and the desired output. Furthermore, in this learning model there is no external influence, since it is not informed whether an output was correct or not. It is only supplied with large amounts of data and each model builds its own associations.

The learning methods are different from those used in the supervised training model. The main algorithm of this type is k-means, which allows grouping or segmenting group observations based on characteristics and distances between each of the observations. This grouping consists of performing a minimization of the sum of the distances between each of the objects and the centroid of its group (cluster) [20].

2.2. Multi-agent systems

An intelligent agent is an autonomous entity capable of performing actions in its environment and perceiving the state of the environment to solve a problem [21]. An agent can be a physical entity, such as a robot equipped with sensors and actuators, or a virtual entity, such as software agents. The latter have fundamental properties of perception, reasoning, learning, decision-making, problem-solving, interaction, and communication [22].

Software agents are classified according to many different criteria. Their categorization depends on the researcher, the tasks they perform, the way they learn, their architecture, among others. According to the architecture, a software agent can be of the following types: reactive, deliberative, and hybrid agents.

The definitions presented above correspond to a simple agent. However, these entities can be grouped together to solve complex problems. The association of several distributed agents, called MAS (Multi-Agent Systems), can achieve common goals. These types of systems have been applied in a wide variety of computer applications, ranging from small autonomous control systems to sophisticated systems capable of negotiation [23].

2.2.1. Reactive agents

This architecture is characterized by the absence of a central reasoning element or a symbolic model of its environment. Instead, agents based on this architecture act and respond to the stimuli presented by the current state of the environment in which they are embedded (Figure 1.f). This implies that the agents handle an event management mechanism based on rules, which are triggered whenever a change occurs in the environment [24]. Such a mechanism is straightforward to implement. However, there are some problems with this model, including the following: it does not support continuous learning, it does not reason, it does not plan for the long term, and each situation is recorded only in a rule system. The operation of a reactive agent is driven by data it retrieves from sensors that collect information from the environment.

2.2.2. Deliberative agents

This architecture uses symbolic representation models of explicitly represented knowledge. It integrates BDI components such as beliefs, desires, and intentions [25]. A belief represents the state of the agent's environment, a desire represents its motivations, and an intention models its goals or objectives. In this architecture, decisions are made using logical reasoning mechanisms based on pattern matching and symbolic manipulation. Due to the complexity of symbolic manipulation algorithms, these agents are challenging to implement.

2.2.3. Hybrid agents

These agents combine two or more philosophies within a single agent. They arise from the need to maximize the abilities and minimize the deficiencies of each agent architecture described above for a specific purpose [26].

3. METHODOLOGY

A systematic review is a synthesis of research in which the content of various studies on a particular topic is systematically identified, critically evaluated, and synthesized according to strict methodological criteria. To conduct the literature review proposed in this study, we followed the methodology proposed by Kofod-Petersen [13]. Based on this methodology, which specializes in literature reviews in Computer Science, we performed the following tasks: (i) formulation of the research questions, (ii) definition of the sources of information, (iii) definition of the scientific search string, and (iv) definition of the inclusion and exclusion criteria for selecting primary studies. These tasks are described in detail in this section.

3.1. Request questions (RQs)

The study presents the following general research question: How have software agents been integrated with machine learning to improve their learning capabilities? The specific questions addressed in this study are:

- RQ1: What is the scope of software agents after integrating machine learning techniques into their structure or behavior?
- RQ2: How well have the most popular machine learning algorithms been used for the development of intelligent systems?
- RQ3: What kind of problems have been solved with the integration of software agents and machine learning techniques?
- RQ4: What kind of learning did intelligent agents' model through machine learning?
- RQ5: What are the strengths, opportunities, weaknesses and threats of agent models based on machine learning techniques?

3.2. Information sources

Two documentary databases (Scopus and Web of Science) and five digital libraries (IEEE Xplore, Elsevier, ACM, Wiley and Springer) were used as sources of information to carry out the scientific search process and thus recover the primary studies to be analyzed.

3.2. Search strategy

The previously defined RQs provided the guidelines to determine a set of terms that allowed us to carry out the search process. The primary terms are constituted by the keywords (software agent, multi-agent system), (machine learning, ML) and (intelligent).

3.3. Inclusion and exclusion criteria

The inclusion criteria defined for this study focused on considering how software agents used machine learning as a basis for creating intelligent systems. Additionally, four exclusion criteria were established to exclude studies that did not provide relevant information. These exclusion criteria were as follows: repeated articles, articles written in a language other than English, inaccessible articles, and articles published before 2010. Articles retrieved with the search string that did not meet these criteria were not analyzed. After applying the criteria described above, 55 studies were analyzed.

4. RESULTS AND DISCUSSION

This section describes the results of conducting the literature review. In summary, this section provides answers to each of the research questions (RQ1-RQ5) that were proposed in the section methodology. To present the results, tables summarizing the results are used.

4.1. Analyzed studies

The scientific search applied in the methodology made it possible to retrieve 55 studies in which proposals for the integration of agents and machine learning were made. A summary of the selected studies is shown in Table 1. Both tables describe the study label, source of information, year of publication, country where the study was conducted, study title and reference.

Table 1: *Analyzed studies in the literature review.*

ID	SOURCE	YEAR	COUNTRY	REF
E1	ACM	2010	USA	[27]
E2	ACM	2010	Canada	[28]
E3	ACM	2011	China	[29]
E4	IEEE	2011	France	[30]
E5	ACM	2010	Canada	[31]
E6	ACM	2012	Spain	[7]
E7	ACM	2012	USA	[32]
E8	ACM	2012	Spain	[33]

E9	ACM	2013	USA	[34]
E10	IEEE	2013	USA	[35]
E11	ACM	2013	USA	[36]
E12	IEEE	2014	Poland	[17]
E13	ACM	2014	USA	[37]
E14	ACM	2016	Singapore	[38]
E15	ACM	2014	USA	[39]
E16	ACM	2017	USA	[40]
E17	ACM	2017	USA	[21]
E18	IEEE	2018	USA	[41]
E19	ACM	2018	Sweeden	[42]
E20	ACM	2018	USA	[43]
E21	ACM	2018	Sweeden	[44]
E22	ACM	2018	Sweeden	[45]
E23	ACM	2018	Canada	[46]
E24	SCOPUS	2018	USA	[47]
E25	ACM	2019	USA	[48]
E26	ACM	2019	Canada	[49]
E27	ACM	2019	Canada	[50]
E28	ACM	2019	Canada	[51]
E29	ACM	2019	USA	[52]
E30	ACM	2019	Canada	[53]
E31	ACM	2019	Cyprus	[54]
E32	ACM	2019	Canada	[55]
E33	ACM	2019	Canada	[56]
E34	ACM	2019	Canada	[57]
E35	ACM	2019	USA	[58]
E36	ACM	2019	Canada	[59]
E37	ACM	2019	Cyprus	[60]
E38	ACM	2019	Canada	[61]
E39	ACM	2019	USA	[62]
E40	ACM	2019	Canada	[63]
E41	ACM	2019	USA	[64]
E42	IEEE	2019	USA	[65]
E43	ACM	2019	USA	[66]
E44	SCOPUS	2020	USA	[64]
E45	ACM	2020	New Zeeland	[67]
E46	ACM	2020	USA	[68]
E47	SCOPUS	2020	USA	[69]
E48	SCOPUS	2020	China	[70]
E49	ACM	2020	Sweeden	[71]
E50	ACM	2020	France	[72]
E51	ACM	2020	USA	[73]
E52	ACM	2020	New Zeeland	[74]
E53	ACM	2020	New Zeeland	[75]
E54	ACM	2020	USA	[76]
E55	IEEE	2020	Australia	[77]

Source: *own elaboration.*

On the other hand, it has been evidenced that the number of publications in which agents and machine learning techniques have been combined has been variant in the last decade. Although the first publications were made between 2010 and 2016, it has not been until 2017 that more studies were published (23 studies as shows Figure 2). One of the explanations for this phenomenon is because machine learning has become popular in recent years thanks to emerging paradigms such as IoT, cloud computing and the big data. In addition, thanks to the development of computing, several machine learning libraries have been incorporated to create predictive models. These models have reached the field of agents to enhance their data-driven learning.

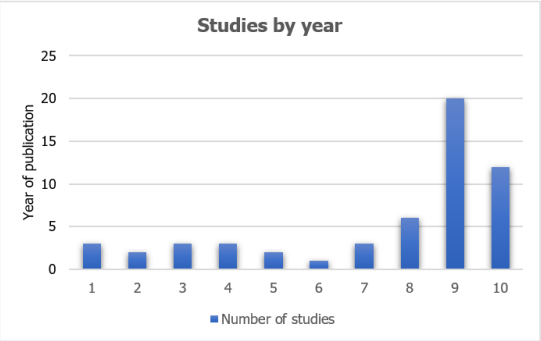

Fig. 2*. Analyzed studies published by year.*
Source: *own elaboration.*

To summarize the data from Table 1, Figure 3 shows a chart illustrating the countries reporting more proposals aimed at integrating software agents and machine learning techniques. It highlights countries such as: USA, Sweden, and Canada with 22, 10 and 11 studies, respectively.

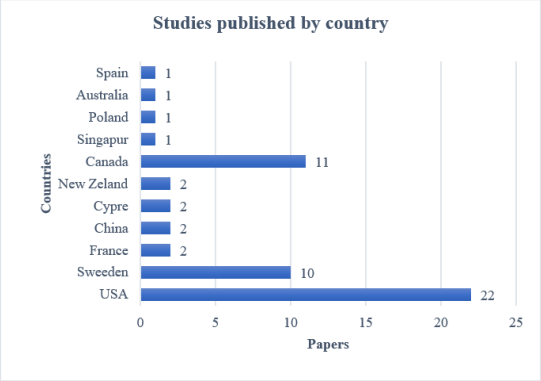

Fig. 3*. Analyzed studies published by country.*
Source: *own elaboration.*

4.2. Findings

Tables have been used to summarize the results, and each question has been logically analyzed and discussed based on the information obtained after reading all 55 papers selected in this study labelled as follows E1-E55.

4.2.1. RQ1: What is the scope of software agents after integrating machine learning techniques into their structure or behavior?

Table 2 shows the scope that software agents have had after being integrated with machine learning techniques. In general terms, such integration made possible changes in the structure of the agents, achieving improvements in the following behaviors: collaboration (58% of the analyzed studies), learning capacity (65%), intelligence (51%) and

finally, the capacity to act autonomously (2% of all studies).

Table 2: *Main scope of the machine learning-based agents used in systems proposed. INT=intelligence, COL=collaboration, LEA=learning, AUT=autonomy.*

ID	INT	COL	LEA	AUT	REF
E1	x	x	x		[27]
E2					[28]
E3			x		[29]
E4	x	x	x		[30]
E5		x			[31]
E6	x		x		[7]
E7	x		x		[32]
E8			x		[33]
E9	x	x	x		[34]
E10		x	x		[35]
E11	x	x		x	[36]
E12			x		[17]
E13	x		x		[37]
E14		x	x		[38]
E15		x			[39]
E16			x		[40]
E17		x			[21]
E18		x			[41]
E19	x		x		[42]
E20	x	x			[43]
E21			x		[44]
E22		x	x		[45]
E23	x	x	x		[46]
E24			x		[47]
E25	x				[48]
E26			x		[49]
E27	x	x	x		[50]
E28		x			[51]
E29			x		[52]
E30	x	x	x		[53]
E31					[54]
E32	x		x		[55]
E33			x		[56]
E34	x	x	x		[57]
E35	x	x	x		[58]
E36		x	x		[59]
E37	x		x		[60]
E38		x	x		[61]
E39	x	x			[62]
E40		x			[63]
E41	x	x			[65]
E42		x	x		[65]
E43	x	x			[66]
E44		x			[64]
E45	x		x		[67]
E46	x	x			[68]
E47			x		[69]
E48	x				[70]
E49		x			[71]
E50	x	x	x		[72]
E51		x	x		[73]
E52	x		x		[74]
E53	x	x			[75]
E54	x	x	x		[76]
E55	x		x		[77]

Source: *own elaboration.*

It is important to point out that some studies took advantage of the integration of both technologies to enhance two or more of the behaviors mentioned above according to the needs of the developed application. Some examples shown in Table 2 describe cases where two (i.e., E6, E10) or more (i.e., E1, E4) aspects were empowered in the same entity.

Contribution in terms of intelligence. In terms of intelligence, software agents have focused mainly on improving their logical reasoning mechanism from which they act in different ways. Intelligence is a core characteristic achieved in many studies, including E1, E5, E16, E19, E20, E30, E36, E50, E54, and E55. These studies employed different approaches to endow agents with the ability to make optimal decisions and solve complex problems. For example, in E55, a combination of advanced classification techniques, such as SVM and Random Forest, was used to enable agents to analyze large volumes of data and make informed decisions in communication networks. Similarly, in E54, intelligence was instrumental in optimizing traffic light networks, where agents learned to coordinate and adjust their decisions in real time to improve traffic flow. Compared to E1 and E16, where intelligence focused on continuous adaptation through Q-Learning, studies E19 and E20 integrated techniques such as PCA and Q-Learning to address monitoring and urban planning problems, demonstrating that intelligence can manifest itself in a variety of ways depending on the context and complexity of the environment.

Contribution in terms of learning. In terms of learning, an agent can observe its environment and based on the changes that occur, internalize them to execute processes that give the agent itself the ability to modify its reasoning skills. Learning is a feature observed in studies E1, E5, E19, E20, E28, E30, E36, E50, E54, and E55, where agents improved their performance through experience and continuous adaptation. In E28, evolutionary learning was applied, allowing agents to explore and optimize solutions over time, which is similar to the adaptive approach observed in E36, where the Actor-Critic approach was used to improve long-term coordination. In E54 and E30, agents learnt through experience in traffic optimization, improving their policies as more data accumulated and more knowledge was gained. In contrast, E5 and E55 used more traditional supervised learning techniques to improve network classification and management, while E19 and E20 integrated learning and optimization into monitoring and planning tasks, demonstrating how learning can be applied in a variety of contexts, from technical optimization to strategic urban planning.

Contribution in terms of collaboration. An agent is collaborative if it can dynamically determine coordination actions in different situations to meet goals without affecting its performance. Collaboration was a key element in studies such as E1, E5, E16, E20, E30, E50, and E54, where agents worked together to achieve common goals. In E54, collaboration was essential for traffic signal network optimization, where agents coordinated their actions to improve traffic efficiency. In contrast, in E50, collaboration was used in spectrum management in IoT networks, allowing agents to share resources and avoid interference. In E30, collaboration was instrumental in traffic simulation at the microscopic level, where agents cooperated to manage vehicular flow in complex urban environments. Unlike studies such as E1 and E5, where collaboration focused more on basic coordination between agents, E20 and E16 explored how agents can collaborate in more complex ways, sharing information and adapting to changes in the environment to optimize urban planning and prediction in dynamic environments.

Contribution in terms of autonomy. An autonomous agent not merely acts as responses to stimuli in the environment but exhibits dynamic and scalable goal-directed behavior. Autonomy was manifested in studies E1, E20, E27, E30, and E36, where agents were able to operate independently, without the need for constant external intervention. In E27, autonomy was achieved allowing agents to make decisions based on their own evaluation of actions, which is similar to what was observed in E36, where agents developed long-term autonomous coordination capabilities. In E30, agents demonstrated autonomy in traffic management, making independent decisions that optimized vehicular flow in real time. On the other hand, in E1 and E20, autonomy was key in adaptation and optimization in complex environments, where agents acted independently to solve control and planning problems without the need for external coordination. These studies highlight how autonomy enables agents to act more effectively in environments where constant supervision is not possible, although this capability requires advanced algorithms and techniques to ensure that agents can independently maintain optimal performance.

4.2.2. RQ2: How well have the most popular machine learning algorithms been used for the development of intelligent systems?

One of the motivations for addressing this study was to learn about the machine learning algorithms used to create predictive models that learn based on data, from the perspective of agents that have proactive decision-making capabilities. Next, how agents have used machine learning algorithms, i.e., supervised, unsupervised and reinforcement learning, is described.

Supervised learning. Supervised algorithms allow the recognition of patterns that are extracted from large data sets. A list of the main studies that have integrated supervised learning to achieve their objectives is presented in Table 3.

Table 3: *Machine learning algorithms used for the analyzed studies. EMPC= Explicit Model Predictive Control, EMPC-based control, XGBoost=Gradient Boosting, RNN= Neural network, KNN=K-nearest neighbors, SVN= Support Vector Machines, RF=Random Forest.*

ID	EMPC	XGBOOST	RNN	KNN	SVM	RF
E18	x					
E25		x				
E36			x			
E37			x			
E55				x	x	x

Source: *own elaboration.*

In study E25, XGBoost was used for software development effort estimation. This algorithm was selected because of its ability to handle large volumes of data and generate accurate predictions, significantly improving the accuracy in estimating the effort required to complete software projects.

On the other hand, Recurrent neural networks (RNNs) were applied in studies E36 and E37 to capture temporal dependencies in multi-agent reinforcement learning environments and in problems where information on the complete state of the system is not always available. These networks enabled agents to learn patterns in temporal sequences and predict future states, thus improving decision making in dynamic and partially observable environments.

Also, the K-Nearest Neighbors (KNN) algorithm was used in the E55 study to classify the transmission quality in a multi-agent managed network based on parameters such as latency and jitter. KNN proved to be effective in real-time classification of these parameters, which is essential for maintaining QoS in communication networks.

Finally, in study E55, SVM, Random Forest and Nu-Support Vector Classifier (Nu-SVC) algorithms were used in a complementary manner to improve the accuracy of transmission parameter classification in networks managed by multiple

agents. SVM was selected for its ability to handle high-dimensional data and perform accurate classifications in complex scenarios. Random Forest contributed its robustness and ability to reduce overfitting, which improved system reliability when classifying complex and noisy data sets. Nu-SVC stood out for its flexibility in adjusting separation margins, which allowed for more accurate and adaptive classification in a network environment with data variations.

No supervised learning. Within the analysis of the studies, the application of unsupervised learning was also evidenced. However, because there are few variants of algorithms of this nature and because these algorithms start from a data set of which there is no a priori knowledge, their use is not widely evidenced. Since the objective of these algorithms is to analyze the understanding of the data or their automatic transformation, the studies analyzed have generally applied more supervised and reinforcement models (Figure 4).

Table 4: *Machine learning unsupervised algorithms used for the analyzed studies. PCA=Principal Component Analysis, HC= Hierarchical Clustering*

ID	K-MEANS	PCA	HC
E5	x		
E8		x	
E13			x
E19		x	
E42		x	
E43	x		

Source: *own elaboration.*

One of the studies that did employ this type of algorithm is the proposals described in E5 and E43. They employed the K-means algorithm to group data into clusters with similar characteristics. In study E5, K-means was applied to classify different types of traffic in cognitive networks, facilitating efficient spectrum management. The agents in this context acted as controllers that manage and adapt spectrum allocation in real time, based on the clusters identified by the algorithm. In study E43, K-means was employed in an adaptive learning system based on case-based reasoning, where agents used clustering to improve decision making and problem solving by organizing knowledge into coherent groups.

On the other hand, the Principal Component Analysis (PCA) algorithm was also employed. In E8, PCA was employed to simplify the feature space in a multi-agent reinforcement learning environment, improving learning efficiency by reducing model complexity. The agents in this study took advantage of the reduced representations to learn faster and more accurately. In E19, PCA facilitated the process of tracking and monitoring multiple objects, allowing agents to process high-dimensional data more efficiently. And in E42, PCA was used in the context of microgrids, where agents used dimensionality reduction to manage and optimize power distribution more effectively, reducing the number of variables to consider without losing critical information.

Finally, the Hierarchical Clustering algorithm was also employed by the E13 study. The algorithm helped to organize data into a hierarchical structure, which allowed for a more detailed and progressive classification of the data. In this study, agents were employed in an agent-based simulation environment, where the hierarchy of clusters helped the agents identify patterns and make decisions based on the level of detail needed. This was particularly useful in simulating human behaviors in urban environments, where agents needed to interpret and act on data clustered at different levels of granularity.

These unsupervised algorithms enabled agents to organize and simplify data, which in turn improved the agents' ability to learn, adapt and act in their respective environments. These algorithms served as fundamental tools for improving the effectiveness and efficiency of agents in solving complex problems in real time.

Reinforcement learning. Reinforcement learning algorithms have also been applied in an integrated manner with agents to carry out decision making processes. The use of the main algorithms is detailed in the summary in Table 5.

Table 5: *Main reinforcement learning algorithms used by the analyzed studies.*

ID	Q-LEARNING	DEEP Q-LEARNING (DQN)	SARSA	ACTOR-CRITIC	POLICY GRADIENT	MONTE CARLO METHODS
E1	x					
E16	x					
E27				x		
E30	x	x		x	x	
E36				x	x	

E44	x	x	x	x	x	x
E50	x		x			x
E23		x				
E13	x					x
E20	x				x	x

Source: *own elaboration.*

The Q-Learning algorithm was used in studies such as E1, E16, E30, E44, E50, E13, and E20. Q-Learning is a value-based method that allows agents to learn an optimal policy by directly iterating over the action value function. In these studies, Q-Learning proved to be a popular choice due to its simplicity and effectiveness, particularly in multi-agent environments where agents need to explore and exploit the environment to find the best action policy. Compared to other methods, Q-Learning is easier to implement and is robust in environments with discrete states. However, one of its challenges is slow convergence in environments with large state spaces. For example, in E16, Q-Learning was used to address prediction problems in dynamic environments, where agents benefited from the algorithm's ability to adapt to real-time changes.

Another algorithm used was DQN. This algorithm, an extension of Q-Learning that uses deep neural networks, was used in studies E23, E30, and E44. DQN is especially useful in situations where the state space is continuous or very large, as in study E30, where agents had to handle traffic simulation at the microscopic level. DQN allowed agents to approximate the value function in complex state spaces, significantly improving the agent's ability to make decisions in high-dimensional environments compared to traditional Q-Learning. The integration of neural networks enabled agents to handle more information-rich environments, albeit at the cost of increased computational complexity and data requirements for training.

Also in the same vein, the SARSA algorithm, employed in studies E44 and E50. SARSA is a value-based method that, unlike Q-Learning, learns the policy directly from the agent's experience, making it more conservative in exploration. In these studies, SARSA was advantageous in scenarios where agents needed a safer approach, minimizing the risk of making decisions that could lead to negative outcomes in uncertain environments. For example, in E50, SARSA was used in IoT network spectrum management, where it was crucial to avoid decisions that could compromise quality of service. Compared to Q-Learning, SARSA offers greater stability in environments where rewards can be volatile.

The Actor-Critic approach was also applied in studies E27, E30, E36, and E44. This approach combines the best of the policy-based (actor) and value-based (critic) methods. This algorithm allows agents to learn what is the best action to take, and it enables also to evaluate the quality of that action, which reduces the variance in policy estimation and improves the stability of learning. In study E36, Actor-Critic was used for long-term coordination in multi-agent environments, where the need to reduce variance and ensure consistent decisions was critical. Compared to Q-Learning and SARSA, Actor-Critic has the advantage of being more efficient in continuous and high-dimensional environments but requires a careful balance between actor and critic updates to avoid instabilities.

Policy Gradient methods, used in E30, E36, E44, and E20, directly optimize the agent's policy as a function of the gradient of expected reward. These methods are particularly useful in scenarios where policies must be adjusted smoothly and continuously, as in E36, where coordination of multiple agents required constant adjustments in a dynamic environment. Agents benefited from the ability of Policy Gradient methods to handle stochastic policies, allowing for more effective exploration and better adaptation to changes in the environment. Compared to Q-Learning and SARSA, Policy Gradient is better suited for continuous and complex environments, although it is more susceptible to problems of slow convergence and high variance in policy updates.

Finally, Monte Carlo methods were also employed in studies E44, E50, E13, and E20. Basically, these methods were used to estimate value functions based on the average return of complete episodes. These methods were especially useful in studies such as E13, where the tasks were episodic and accurate state-action value estimation was required. Agents in these studies benefited from the simplicity of policy updating based on full samples of episodes, which is advantageous in settings where rewards can be highly variable. However, compared to methods such as Q-Learning or Policy Gradient, Monte Carlo methods may be less efficient in settings where episodes are long or where reward feedback is sporadic.

4.2.3. RQ3: What kind of problems have been solved with the integration of software agents and machine learning techniques?

Many of the models in which agents have been integrated with machine learning are conceptual proposals. The agent models that have been taken to the practical field have allowed solving some interesting, particular and useful problems in the modern world. Table 6 describes some of the problems in which the agents under study in this work have been used.

Table 6: *Main problems in which agent-based machine learning have been used.*

ID	DOMAIN	SOLVED PROBLEM
E1	AI and Multi-Agent	Development of machine learning algorithms for coordination in multi-agent systems.
E4	AI and Communication	Improving communication among agents in a multi-agent environment using reinforcement learning.
E5	Telecommunications	Spectrum management in cognitive radios using learning techniques.
E7	Reinforcement Learning	Improving generalization in multi-agent reinforcement learning through tensor factorization.
E10	Control and Automation	Implementation of cooperative learning in continuous control systems.
E11	Urban Traffic Management	Optimization of urban traffic control using multi-agent reinforcement learning.
E13	Agent-Based Simulation	Simulation and prediction of collective behaviors in urban environments.
E14	Logistics and Optimization	Optimization of dynamic dispatch in logistics operations using multi-agent reinforcement learning.
E16	Dynamic Environments	Development of prediction techniques to improve reinforcement learning performance in dynamic environments.
E18	Urban Traffic Management	Implementation of a shared machine learning approach for urban traffic management.
E19	Tracking and Monitoring	Development of a multi-agent system for tracking and monitoring multiple objects using reinforcement learning.
E20	Urban Planning	Application of multi-agent reinforcement learning for planning and development of urban projects.
E22	Artificial Intelligence and Multi-Agent	Proposal of value-decomposition networks to improve cooperation in multi-agent environments.
E24	Software Development	Application of machine learning techniques for effort estimation in software development.
E25	Risk Management	Development of risk-averse reinforcement learning approaches in mixed multi-agent environments.
E27	Control and Optimization	Development of actor-critic algorithms for multi-agent reinforcement learning in constrained environments.
E28	Artificial Intelligence and Multi-Agent	Training cooperative agents in multi-agent reinforcement learning systems.
E29	Autonomous Driving	Reinforcement learning architecture with parameter sharing for autonomous driving in multi-agent environments.
E30	Traffic Management	Simulation and optimization of microscopic traffic using deep multi-agent reinforcement learning.
E34	Deep Reinforcement Learning	Development of competitive deep reinforcement learning techniques in multi-agent environments.
E35	Artificial Intelligence and Learning	Improvement of cooperative reinforcement learning algorithms by incorporating mixed demonstrations.
E36	Deep Reinforcement Learning	Implementation of hierarchical deep reinforcement learning for long-term coordination in multi-agent environments.
E37	Adaptive Control	Adaptive control of multi-agent systems using deep reinforcement learning.
E38	Simulation and Healthcare	Evaluation of care services through agent-based simulation and machine learning.
E39	Urban Traffic Management	Creation of a multi-agent reinforcement learning environment for large-scale city traffic management.
E41	Logistics and Transportation	Optimization of order dispatching through vehicle allocation using multi-agent reinforcement learning.
E42	Energy and Power Grids	Development of a protection scheme for AC microgrids based on multi-agent systems and machine learning.
E43	Artificial Intelligence and Learning	Implementation of an adaptive multi-agent learning system based on incremental hybrid case-based reasoning.
E45	Urban Traffic Management	Application of feudal deep reinforcement learning for traffic management in urban environments.
E46	Logistics and Optimization	Development of a cooperative multi-agent reinforcement learning system for optimizing express systems.
E47	Artificial Intelligence and Communication	Development of techniques for learning multi-agent communication through grounded sender-receiver games.
E50	Telecommunications	Cooperative spectrum management in IoT cognitive networks using multi-agent reinforcement learning.

E52	Urban Traffic Management	Optimization of traffic management using multi-agent reinforcement learning with emergent communication.
E54	Urban Traffic Management	Optimization of traffic signal networks through the integration of independent and centralized multi-agent reinforcement learning.
E55	Networks and Communications	Application of machine learning techniques for transmission parameters classification in multi-agent managed networks

Source: *own elaboration.*

In the field of urban traffic optimization, several studies used multi-agent reinforcement learning techniques. Among them, studies E11, E39, E45, E54 and E52 share the objective of improving traffic management in urban environments by optimizing traffic signals and vehicular flows. However, each approaches the problem from a different perspective. For example, E39 presents a specific environment for large-scale traffic simulation and optimization, while E54 proposes a combination of independent and centralized learning to optimize traffic signal networks. On the other hand, E45 employs a feudal deep reinforcement learning approach to manage traffic, focusing on the hierarchical structure of decision making.

Within the field of artificial intelligence applied to multi-agent systems, studies E1, E4, E22, and E28 present significant advancements. These studies all focus on developing algorithms and architectures that enable efficient cooperation and communication between agents within a shared environment. Notably, E4 and E28 concentrate specifically on improving communication between agents by leveraging reinforcement learning techniques. In contrast, E22 proposes a value decomposition network that facilitates cooperation between agents. E1, however, takes a broader approach, exploring the development of learning algorithms for coordination in multi-agent systems without restricting itself to a specific domain.

Studies E14, E46, and E41 further demonstrate the applicability of reinforcement learning in logistics optimization. These investigations explore how these algorithms can optimize tasks like dynamic resource dispatching and vehicle allocation specifically for logistics and transportation. For instance, E14 tackles the broader challenge of dynamic dispatching in logistics operations, while E41 focuses on optimizing vehicle allocation for order delivery. E46, on the other hand, utilizes a cooperative reinforcement learning approach to optimize express courier systems.

In the domain of energy and power grids, study E42 proposes an approach to the protection of AC microgrids using multi-agent systems and machine learning. This study is unique in its application and shows how reinforcement learning approaches can be extended to more technical and specific domains such as critical infrastructure management and protection.

Finally, in agent-based simulation, study E13 focuses on the prediction of collective behavior in urban environments using agent-based models. This approach is different from other studies more oriented to process optimization, as it focuses on understanding and predicting human behavior through simulation.

4.2.4. RQ4: What other kind of learning the agents modelled to be more intelligent?

The analysis of the studies made it possible to determine that the agents, thanks to other techniques additional to machine learning, made it possible to model more complex and intelligent behaviors. A description of how these techniques or methos were applied is described as follow:

One of the most widespread techniques used in the studies is Rule-Based Learning. This technique is used in the E5 study, where agents make decisions following a predefined set of rules. This technique is especially useful in environments where the expected behavior can be explicitly codified.

Subsequently, Bayesian Learning is also applied in study E12. In this approach, agents continuously update their beliefs about the world as they acquire new evidence. This technique is fundamental in scenarios characterized by high uncertainty, where probability plays a crucial role in decision making.

Case-Based Reasoning is also studied in E13 and E43. This technique allows agents to solve new problems by adapting solutions from similar problems solved in the past. It is especially useful in situations where a rich history of previous experiences is available.

On the other hand, Decision Tree Learning is applied in study E24. This technique divides the decision space into more manageable subsets, allowing agents to make decisions based on a hierarchy of questions.

Continuing with the analysis, Transfer Learning, in E41, is presented. This technique allows agents to leverage knowledge acquired in one context to improve their performance in another related context, which is especially useful when data in the new domain is limited.

Study E43 combines case-based reasoning with incremental methods in Incremental Hybrid Case-Based Reasoning (IHCBR), allowing agents to continuously improve their knowledge base as they acquire new experience.

Finally, Imitation Learning is used by the authors of study E53. This technique allows agents to learn behaviors by observing other agents or humans, which is useful in environments where the desired behavior can be observed but not easily programmed.

4.2.6. RQ5: What are the strengths, opportunities, weaknesses and threats of agent models based on machine learning techniques?

The following strengths, opportunities, weaknesses and threats have been determined regarding the development of agents that use machine learning to improve the inherent features of agents (i.e., intelligence, collaboration, learning, adaptation and proactivity).

Strengths (S)

- S1. Existence of implementation-level machine learning models and algorithms compatible with several programming languages that facilitate their integration with agent development tools. For example, JADE can implement Weka or Deeplearning4j (DL4J) machine learning models, and SPADE can implement machine learning models in Python using scikit-learn.
- S2. Use of cloud services to distribute learning mechanisms without the need for agents to integrate them as part of their structure. This allows agents to be lightweight and new configurations to be applied without modifying the agents themselves. Additionally, the training processes of the models will be done in sophisticated infrastructures. This is of utmost importance for the development of cloud-compatible autonomous car systems and edge processing.
- S3. Availability of standards for agent development, which allows learning agents to share their knowledge with counterpart agents within the ecosystem in which the agent executes and collaborates. A specific case is the FIPA-ACL standard for establishing communication between agents and FIPA communication protocols for modeling complex interaction processes in heterogeneous systems, among others.

Weaknesses (W)

- W1. Machine learning models are usually too heavy for an agent to integrate into its structure. This complexity makes them difficult to use in embedded systems and dependent on cloud computing.
- W2. There are no formal methodologies or tools that allow the development of this type of agent without the need to integrate several available tools.

Opportunities (O)

- O1. There is a need for intelligent entities that learn from big data for modern systems. The availability of data generated by social networks and the IoT positions agents using machine learning techniques as useful entities to operate in advanced intelligent ecosystems.
- O2. There is availability of platforms oriented towards the development of agents in multiple programming languages. This enables the development of agents that enhance their learning capacity so they can be used in different emerging environments such as web applications, apps, embedded systems, the cloud, edge computing, fog computing, among others.
- O3. Agents integrating data-driven learning mechanisms can employ blockchain to determine the reliability of the data used to model learning actions. This would be of great importance to prevent agents from learning from unreliable information and consequently making inadequate decisions that affect the environment in which they operate.

Threats (T)

- T1. Agents are vulnerable to attacks by computer experts. Although machine learning techniques help to improve the learning capability of the agent's environment, they do not contribute significantly to improving security and data privacy mechanisms. However, using machine learning, agents could

learn to identify malicious behaviors, detect fake news, and more.

4. CONCLUSIONS

This paper analyzed how software agents enhance their level of intelligence, collaboration, autonomy, and adaptation by integrating machine learning models. The theoretical and practical foundations of agent-oriented technologies and machine learning for the creation of intelligent systems were examined. Proposals were made incorporating supervised and unsupervised learning algorithms. However, in most cases, agents and multi-agent systems used reinforcement learning algorithms. By combining these algorithms, the agents could optimize tasks related to communication and coordination in both known and unknown scenarios.

Agent-oriented technologies have wide development opportunities in light of the boom of paradigms such as Artificial Intelligence, Cloud Computing, Blockchain, and Big Data. It has been proven that agents, through machine learning algorithms, can learn from data and execute automatic prediction tasks to achieve their goals more effectively; that is, better coordinating actions in groups of agents and identifying patterns in the data from the environment in which they operate to make better decisions. Additionally, it has been shown that deep learning models, many of them based on supervised learning, have also been used by agents and multi-agent systems. Undoubtedly, this integration process will enable the enhancement of emerging scenarios such as the Internet of Things and the Internet of Agents, a paradigm in which intelligent agents are the predominant actors.

For future work, it is proposed that the dynamics of intelligent agents with machine learning technologies be incorporated into the interaction of big data in cloud computing to implement proactive decision support systems useful in different Internet ecosystems such as the Internet of Things and their respective applications in cities, universities, hospitals, and smart industries.

REFERENCES

[1] S. K. Polu, “Modeling of efficient multi-agent based mobile health care system,” *Int J Innov Res Sci Technol*, vol. 5, no. 8, pp. 10–14, 2019.

[2] S. Munawar, S. K. Toor, M. Aslam, and E. Aimeur, “PACA-ITS: A Multi-agent system for intelligent virtual laboratory courses,” *Applied Sciences (Switzerland)*, vol. 9, no. 23, 2019, doi: 10.3390/app9235084.

[3] S. Cho and F. Zhang, “An adaptive control law for controlled lagrangian particle tracking,” *WUWNet 2016 - 11th ACM International Conference on Underwater Networks and Systems*, 2016, doi: 10.1145/2999504.3001077.

[4] R. K. Jain *et al.*, “Stability analysis of piezoelectric actuator based micro gripper for robotic micro assembly,” *ACM International Conference Proceeding Series*, no. c, 2013, doi: 10.1145/2506095.2506105.

[5] M. Tanti, S. Fossey, L. Madrid-Briand, P. Carrieri, B. Spire, and P. Roux, “Une analyse de Twiter pour mieux comprendre les acteurs de la communication des nouvelles drogues et leurs discussions,” *ACM International Conference Proceeding Series*, pp. 36–38, 2018, doi: 10.1145/3240431.3240438.

[6] Y. Islen and S. Juan, “Componente para la extracción y transformación de datos en el proceso de vigilancia tecnológica Component for the data mining and transformation within the technological surveillance process,” no. June 2017, 2016.

[7] M. Kaisers, D. Bloembergen, and K. Tuyls, “A Common Gradient in Multi-agent Reinforcement Learning (Extended Abstract),” *Proc. of 11th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 1393–1394, 2012.

[8] S. H. Chen and T. K. Fu, “Eliminating artificial-natural dichotomy a formal study on a core cognitive process in artificial intelligence,” *ACM International Conference Proceeding Series*, vol. Part F1285, 2017, doi: 10.1145/3080845.3080866.

[9] S. R. Hamidi, E. N. M. Ibrahim, M. F. B. A. Rahman, and S. M. Shuhidan, “Industry 4.0 urban mobility: goNpark smart parking tracking module,” *ACM International Conference Proceeding Series*, pp. 503–507, 2017, doi: 10.1145/3162957.3163042.

[10] C. Cappelli, G. V. Pereira, M. B. Bernardes, F. Bernardini, and A. Gomyde, “Building a reference model & an evaluation method for cities of the Brazilian network of smart & human cities,” *ACM International Conference Proceeding Series*, vol. Part F1282, pp. 580–581, 2017, doi: 10.1145/3085228.3085257.

[11] A. A. F. Brandão, L. Vercouter, S. Casare, and J. Sichman, “Exchanging reputation

values among heterogeneous agent reputation models: An experience on ART testbed," *Proceedings of the International Conference on Autonomous Agents*, vol. 5, pp. 1047–1049, 2007, doi: 10.1145/1329125.1329405.

[12] I. Menchaca, M. Guenaga, and J. Solabarrieta, "Using learning analytics to assess project management skills on engineering degree courses," *ACM International Conference Proceeding Series*, vol. 02-04-Nove, pp. 369–376, 2016, doi: 10.1145/3012430.3012542.

[13] A. Kofod-petersen, "How to do a Structured Literature Review in computer science," 2014. [Online]. Available: https://research.idi.ntnu.no/aimasters/files/SLR_HowTo2018.pdf

[14] E. Saadatian, T. Salafi, H. Samani, Y. De Lim, and R. Nakatsu, "An affective telepresence system using smartphone high level sensing and intelligent behavior generation," *HAI 2014 - Proceedings of the 2nd International Conference on Human-Agent Interaction*, pp. 75–82, 2014, doi: 10.1145/2658861.2658878.

[15] J. Jumadinova, P. Dasgupta, and L. K. Soh, "Strategic capability-learning for improved multi-agent collaboration in ad-hoc environments," *Proceedings - 2012 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2012*, vol. 2, pp. 287–292, 2012, doi: 10.1109/WI-IAT.2012.57.

[16] J. A. Manrique, J. S. Rueda-Rueda, and J. M. T. Portocarrero, "Contrasting Internet of Things and Wireless Sensor Network from a Conceptual Overview," *Proceedings - 2016 IEEE International Conference on Internet of Things; IEEE Green Computing and Communications; IEEE Cyber, Physical, and Social Computing; IEEE Smart Data, iThings-GreenCom-CPSCom-Smart Data 2016*, pp. 252–257, 2017, doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2016.66.

[17] T. H. Teng, A. H. Tan, J. A. Starzyk, Y. S. Tan, and L. N. Teow, "Integrating motivated learning and k-winner-take-all to coordinate multi-agent reinforcement learning," *Proceedings - 2014 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IAT 2014*, vol. 3, pp. 190–197, 2014, doi: 10.1109/WI-IAT.2014.167.

[18] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," *Adv Neural Inf Process Syst*, vol. 4, no. January, pp. 3581–3589, 2014.

[19] E. Levy, O. E. David, and N. S. Netanyahu, "Genetic algorithms and deep learning for automatic painter classification," *GECCO 2014 - Proceedings of the 2014 Genetic and Evolutionary Computation Conference*, no. Dl, pp. 1143–1150, 2014, doi: 10.1145/2576768.2598287.

[20] H. Kim, Y. Kim, and J. Hong, "Cluster management framework for autonomic machine learning platform," *Proceedings of the 2019 Research in Adaptive and Convergent Systems, RACS 2019*, pp. 128–130, 2019, doi: 10.1145/3338840.3355691.

[21] R. Rădulescu, P. Vrancx, and A. Nowé, "Analysing congestion problems in multi-agent reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 3, pp. 1705–1707, 2017.

[22] S. Kim, Y. K. Row, and T. J. Nam, "Thermal interaction with a voice-based intelligent agent," *Conference on Human Factors in Computing Systems - Proceedings*, vol. 2018-April, pp. 1–6, 2018, doi: 10.1145/3170427.3188656.

[23] J. Yan, D. Hu, S. S. Liao, and H. Wang, "Mining agents' goals in agent-oriented business processes," *ACM Trans Manag Inf Syst*, vol. 5, no. 4, 2015, doi: 10.1145/2629448.

[24] K. Hassani and W. S. Lee, "On designing migrating agents: From autonomous virtual agents to intelligent robotic systems," *SIGGRAPH Asia 2014 Autonomous Virtual Humans and Social Robot for Telepresence, SA 2014*, 2014, doi: 10.1145/2668956.2668963.

[25] D. Singh, L. Padgham, and B. Logan, "Integrating BDI Agents with Agent-Based Simulation Platforms," *Auton Agent Multi Agent Syst*, vol. 30, no. 6, pp. 1050–1071, 2016, doi: 10.1007/s10458-016-9332-x.

[26] A. Leite, R. Girardi, and P. Novais, "Using ontologies in hybrid software agent architectures," *Proceedings - 2013 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Workshops, WI-IATW 2013*, vol. 3, pp. 155–158, 2013, doi: 10.1109/WI-IAT.2013.172.

[27] R. Amin and S. Khalid, “Machine Learning Algorithms for Depression”.

[28] A. Wilson and A. Fern, “Bayesian Role Discovery for (Extended Abstract),” *Learning*, pp. 1587–1588.

[29] M. E. Taylor, B. Kulis, and F. Sha, “Metric learning for reinforcement learning agents,” *10th International Conference on Autonomous Agents and Multiagent Systems 2011, AAMAS 2011*, vol. 2, pp. 729–736, 2011.

[30] S. Hoet and N. Sabouret, “Reinforcement learning of communication in a multi-agent context,” *Proceedings - 2011 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2011*, vol. 2, pp. 240–243, 2011, doi: 10.1109/WI-IAT.2011.125.

[31] C. Wu *et al.*, “Spectrum Management of Cognitive Radio Using Multi-agent Reinforcement Learning,” in *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, 2010, pp. 10–14. [Online]. Available: www.ifaamas.org

[32] S. Bromuri, “A tensor factorization approach to generalization in multi-agent reinforcement learning,” *Proceedings - 2012 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2012*, vol. 2, pp. 274–281, 2012, doi: 10.1109/WI-IAT.2012.21.

[33] W. T. L. Teacy *et al.*, “Decentralized Bayesian reinforcement learning for online agent collaboration,” *11th International Conference on Autonomous Agents and Multiagent Systems 2012, AAMAS 2012: Innovative Applications Track*, vol. 1, pp. 312–319, 2012.

[34] X. Zhu, C. Zhang, and V. Lesser, “Combining dynamic reward shaping and action shaping for coordinating multi-agent learning,” *Proceedings - 2013 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2013*, vol. 2, pp. 321–328, 2013, doi: 10.1109/WI-IAT.2013.127.

[35] L. Torrey and M. E. Taylor, “Teaching on a Budget: Agents advising agents in reinforcement learning,” *12th International Conference on Autonomous Agents and Multiagent Systems 2013, AAMAS 2013*, vol. 2, pp. 1053–1060, 2013.

[36] C. Zhang and V. Lesser, “Coordinating multi-agent reinforcement learning with limited communication,” *12th International Conference on Autonomous Agents and Multiagent Systems 2013, AAMAS 2013*, vol. 2, no. Aamas, pp. 1101–1108, 2013.

[37] W. Rand, “Machine Learning Meets Agent-Based Modelling: When Not To Go: When Not To Go To a Bar,” 2006. [Online]. Available: https://ccl.northwestern.edu/papers/agent2006rand.pdf

[38] P. Mannion, K. Mason, S. Devlin, J. Duggan, and E. Howley, “Multi-objective dynamic dispatch optimisation using Multi-Agent Reinforcement Learning,” *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pp. 1345–1346, 2016.

[39] H. Wang *et al.*, “Integrating reinforcement learning with multi-agent techniques for adaptive service composition,” *ACM Transactions on Autonomous and Adaptive Systems*, vol. 12, no. 2, 2017, doi: 10.1145/3058592.

[40] A. Marinescu, I. Dusparic, and S. Clarke, “Prediction-based multi-agent reinforcement learning in inherently non-stationary environments,” *ACM Transactions on Autonomous and Adaptive Systems*, vol. 12, no. 2, 2017, doi: 10.1145/3070861.

[41] G. Henri and N. Lu, “A Multi-Agent Shared Machine Learning Approach for Real-time Battery Operation Mode Prediction and Control,” *IEEE Power and Energy Society General Meeting*, vol. 2018-Augus, pp. 1–5, 2018, doi: 10.1109/PESGM.2018.8585907.

[42] P. Rosello and M. J. Kochenderfer, “Multi-agent reinforcement learning for multi-object tracking,” *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2, pp. 1397–1413, 2018.

[43] B. Khelifa and M. R. Laouar, “Multi-agent reinforcement learning for urban projects planning,” *ACM International Conference Proceeding Series*, 2018, doi: 10.1145/3330089.3330134.

[44] H. Kazmi, J. Suykens, and J. Driesen, “Valuing knowledge, information and agency in multi-agent reinforcement learning: A case study in smart buildings: Industrial applications track,” *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 1, pp. 585–587, 2018.

[45] P. Sunehag *et al.*, “Value-decomposition networks for cooperative multi-agent learning based on team reward,” *Proceedings of the International Joint*

Conference on Autonomous Agents and Multiagent Systems, AAMAS, vol. 3, pp. 2085–2087, 2018.

[46] G. Palmer and K. Tuyls, "Lenient Multi-Agent Deep Reinforcement Learning," no. Aamas, pp. 443–451, 2018.

[47] J. Wang and L. Sun, "Dynamic holding control to avoid bus bunching: A multi-agent deep reinforcement learning framework," *Transp Res Part C Emerg Technol*, vol. 116, no. April, p. 102661, 2020, doi: 10.1016/j.trc.2020.102661.

[48] W. Amaral, G. Braz, L. Rivero, and D. Viana, "Using machine learning technique for effort estimation in software development," *ACM International Conference Proceeding Series*, 2019, doi: 10.1145/3364641.3364670.

[49] G. Palmer, R. Savani, and K. Tuyls, "Negative update intervals in deep multi-agent reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 1, pp. 43–51, 2019.

[50] R. B. Diddigi, K. J. Prabuchandran, D. Sai Koti Reddy, and S. Bhatnagar, "Actor-critic algorithms for constrained multi-agent reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 4, pp. 1931–1933, 2019.

[51] S. Bhalla, S. G. Subramanian, and M. Crowley, "Training cooperative agents for multi-agent reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 3, pp. 1826–1828, 2019.

[52] M. Kaushik, N. Singhania, S. Phaniteja, and K. M. Krishna, "Parameter sharing reinforcement learning architecture for multi agent driving," *ACM International Conference Proceeding Series*, pp. 0–6, 2019, doi: 10.1145/3352593.3352625.

[53] G. Bacchiani, D. Molinar, and M. Patander, "Microscopic traffic simulation by cooperative multi-agent deep reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 3, pp. 1547–1555, 2019.

[54] T. Molderez, B. Oeyen, C. De Roover, and W. De Meuter, "Marlon - a domain-specific language for multi-agent reinforcement learning on networks," *Proceedings of the ACM Symposium on Applied Computing*, vol. Part F1477, pp. 1322–1329, 2019, doi: 10.1145/3297280.3297413.

[55] M. Zhou *et al.*, "Factorized Q-Learning for Large-Scale Multi-Agent Systems," 2019.

[56] D. S. K. Reddy, A. Saha, S. G. Tamilselvam, P. Agrawal, and P. Dayama, "Risk averse reinforcement learning for mixed multi-agent environments," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 4, no. 2, pp. 2171–2173, 2019.

[57] Y. Zhao and X. Ma, "Learning efficient communication in cooperative multi-agent environment," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 4, pp. 2321–2323, 2019.

[58] W. Zhou, Y. Chen, and J. Li, "Competitive Evolution Multi-Agent Deep Reinforcement Learning," in *CSAE2019*, China, 2019.

[59] H. R. Lee and T. Lee, "Improved cooperative multi-agent reinforcement learning algorithm augmented by mixing demonstrations from centralized policy," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2, no. Aamas, pp. 1089–1098, 2019.

[60] M. Ossenkopf, M. Jorgensen, and K. Geihs, "Hierarchical multi-agent deep reinforcement learning to develop long-term coordination," *Proceedings of the ACM Symposium on Applied Computing*, vol. Part F1477, pp. 922–929, 2019, doi: 10.1145/3297280.3297371.

[61] J. Castellini, R. Savani, F. A. Oliehoek, and S. Whiteson, "The representational capacity of action-value networks for multi-agent reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 4, no. 1, pp. 1862–1864, 2019.

[62] H. Zhang *et al.*, "CityFlow: A multi-agent reinforcement learning environment for large scale city traffic scenario," *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, pp. 3620–3624, 2019, doi: 10.1145/3308558.3314139.

[63] X. Li, J. Zhang, J. Bian, Y. Tong, and T. Y. Liu, "A cooperative multi-agent reinforcement learning framework for resource balancing in complex logistics network," *Proceedings of the International Joint Conference on Autonomous Agents and*

Multiagent Systems, AAMAS, vol. 2, pp. 980–988, 2019.

[64] J. Hook, V. De Silva, and A. Kondoz, "Deep Multi-Critic Network for accelerating Policy Learning in multi-agent environments," *Neural Networks*, vol. 128, pp. 97–106, 2020, doi: 10.1016/j.neunet.2020.04.023.

[65] M. Uzair, L. Li, J. G. Zhu, and M. Eskandari, "A protection scheme for AC microgrids based on multi-agent system combined with machine learning," *2019 29th Australasian Universities Power Engineering Conference, AUPEC 2019*, pp. 17–22, 2019, doi: 10.1109/AUPEC48547.2019.211845.

[66] N. El Ghouch, M. Kouissi, and E. M. En-Naimi, "Multi-agent adaptive learning system based on incremental hybrid case-based reasoning (IHCBR)," *ACM International Conference Proceeding Series*, 2019, doi: 10.1145/3368756.3369030.

[67] J. Ma and F. Wu, "Feudal multi-agent deep reinforcement learning for traffic signal control," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2020-May, pp. 816–824, 2020.

[68] Y. Li, Y. Zheng, and Q. Yang, "Cooperative Multi-Agent Reinforcement Learning in Express System," *International Conference on Information and Knowledge Management, Proceedings*, pp. 805–814, 2020, doi: 10.1145/3340531.3411871.

[69] H. Mao, Z. Zhang, Z. Xiao, Z. Gong, and Y. Ni, *Learning multi-agent communication with double attentional deep reinforcement learning*, vol. 34, no. 1. Springer US, 2020. doi: 10.1007/s10458-020-09455-w.

[70] C. Hu, "A confrontation decision-making method with deep reinforcement learning and knowledge transfer for multi-agent system," *Symmetry (Basel)*, vol. 12, no. 4, pp. 1–24, 2020, doi: 10.3390/SYM12040631.

[71] O. Batata, V. Augusto, and X. Xie, "Mixed Machine learning and Agent-based Simulation for Respite Care Evaluation," in *Proceedings of the 2018 Winter Simulation Conference*, 2016, pp. 1–23.

[72] D. Dašić, M. Vučetić, M. Perić, M. Beko, and M. Stanković, "Cooperative Multi-Agent Reinforcement Learning for Spectrum Management in IoT Cognitive Networks," *ACM International Conference Proceeding Series*, vol. Part F1625, no. Cm, pp. 238–247, 2020, doi: 10.1145/3405962.3405996.

[73] J. Yang, I. Borovikov, and H. Zha, "Hierarchical cooperative multi-agent reinforcement learning with skill discovery," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2020-May, pp. 1566–1574, 2020.

[74] S. Gupta, R. Hazra, and A. Dukkipati, "Networked multi-agent reinforcement learning with emergent communication," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2020-May, no. i, pp. 1858–1860, 2020.

[75] D. E. Hostallero, D. Kim, S. Moon, K. Son, W. J. Kang, and Y. Yi, "Inducing cooperation through reward reshaping based on peer evaluations in deep multi-agent reinforcement learning," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2020-May, pp. 520–528, 2020.

[76] Z. Zhang, J. Yang, and H. Zha, "Integrating independent and centralized multi-agent reinforcement learning for traffic signal network optimization," *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, vol. 2020-May, pp. 2083–2085, 2020.

[77] D. Zelasko, P. Plawiak, and J. Kolodziej, "Machine learning techniques for transmission parameters classification in multi-agent managed network," *Proceedings - 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing, CCGRID 2020*, pp. 699–707, 2020, doi: 10.1109/CCGrid49817.2020.00-20.