Towards an Agent-Based Architecture using Deep Reinforcement Learning for Intelligent Internet of Things Applications

Dhouha Ben Noureddine[1,2], Moez Krichen[3,4], Seifeddine Mechti[5], Tarik Nahhal[6], and Wilfried Yves Hamilton Adoni[6]

[1] LISI, INSAT, University of Carthage, Tunisia,
[2] FST, University of El Manar, Tunisia
[3] FCSIT, Albaha University, Albaha, Saudi Arabia
[4] REDCAD, University of Sfax, Tunisia
[5] Miracl Laboratory, University of Sfax, Tunisia
[6] Hassan II University of Casablanca, Morocco
dhouha.bennoureddine@fst.utm.tn, moez.krichen@redcad.org, mechtiseif@gmail.com, t.nahhal@fsac.ac.ma, adoniwilfried@gmail.com

Abstract. Internet of Things (IoT) is composed of many IoT devices connected throughout the Internet, that collect and share information to represent the environment. IoT is currently restructuring the actual manufacturing to smart manufacturing. However, inherent characteristics of IoT lead to a number of titanic challenges such as decentralization, weak interoperability, security, etc. The artificial intelligence provides opportunities to address IoT's challenges, e.g the agent technology. This paper presents first an overview of ML and discusses some related work. Then, we briefly present the classic IoT architecture. Then we introduce our proposed Intelligent IoT (IIoT) architecture. We next concentrate on introducing the approach using multi-agent DRL in IIoT. Finally, in this promising field, we outline the open directions of future work.

Keywords: Internet of Things, intelligent IoT, software agent, deep reinforcement learning, intelligent IoT architecture

1 Introduction

The Internet of Things is a network of systems where many objects are interconnected to each other through the Internet. It provides new applications to enhance the life quality of humans; including the waste management, urban traffic control, environmental detection, social interaction gadgets, sustainable urban environment, emergency services, mobile shopping, smart metering, smart home automation, healthcare, transportation and logistics, etc. Nonetheless, many study issues in the fields of software architecture, application development, network security, and reliability need to be addressed to attain the complete potential of IoT in the aforementioned fields. As more of the infrastructure, both virtual and physical, "get interconnected through the IoT, the

opportunities for more disruptive and widespread cyberattacks grow concomitantly" [1]. As a further matter, the real circumstance is that these objects remain far off from reaching natural intelligence boundaries and there are still many other problems resulting from the interoperability of things and edge devices in a dynamic environment.

In this context, many researchers ([2]; [3]) have integrated the Artificial Intelligence (AI) with IoT technologies. For example, when a large number of devices connected to the Internet generate big quantities of sensory data to represent the physical world's status. These data can be interpreted and analyzed using machine learning (ML) algorithm, which makes good decisions to monitor devices' reactions to the physical world. ML is considered to be one of the most suitable computational paradigms to provide embedded intelligence in the IoT devices [7]. The integration process of IoT and AI, therefore, plays a crucial role in the technology and makes the IoT became intelligent and autonomous by seeing system capabilities grow, including increasing operational efficiency.

The Intelligent IoT has a more complicated system involving identification, sensing, perception, communication, computation, and services. We use in our approach the software agents that have been already proposed for distributed IoT application execution in such distributed context [10]. Agents are designed and programmed to execute tasks as components of distributed applications. Similarly, agents related to things and edge devices in IoT improve interoperability, autonomy, behavior, reasoning, learning, etc. Accordingly, the development of the IoT system architectures has been facilitated by the communication and cooperation between these agents representing things. Thus reducing at the least human interventions through the use of autonomous behavior of agents in the IoT applications control or execution. Agents have so far largely unexplored, specifically in the IoT interoperability context.

This paper aims at providing new approach integrating the learning agent that is able to act and adapt based on new information into the classic architecture IoT. The presented research provides an IIoT architecture based on software agent framework that applied machine learning to tackle many IoT applications. The rest of this paper is structured as following. Section 2 includes some current research related to this area. Section 3 presents the machine learning algorithms. After that, in section 4 an architecture of IIoT and a proposed approach using multi-agent DRL in IIoT will be depicted in detail. Finally, we discuss and conclude our work in Section 5.

2 Related work

Most applications of IoT in the literature focus on the consolidation of the other approaches such as Internet of agents, Internet of robots, Internet of Services, Internet of Drones, Internet of People, etc. Software agents can interact with the other agents in the environment to accomplish their goals [6]. Some research using the agents, create an embodied agents, e.g virtual robots, wireless devices, ubiquitous computing, etc, to design the interactions with the sensors and actu-

ators of the IoT applications [11]. To create an embodied agent, the IT experts provides this agent with a behavior description that is consistent with its body and the task to be performed. It is very hard, however, to completely define a physical system's behaviors at the time of design and to recognize and promote characteristics that contribute to attractive collective behavior. In order to solve these issues, some approaches ([12, 13]) have suggested using evolving machine learning such as neural networks to enable an embodied agent to learn how to adapt their behavior in a dynamic environment. In fact, through learning skills, agents will be able to reason conveniently, create a suitable policy and make good decisions [5]. During the research, many contributions have been proposed in the field of machine learning. In particular, approaches from deep reinforcement learning (DRL) for intelligent IoT.

Others have proposed an architecture based on multi-agent system (MAS) [15] that aims to coordinate IoT devices. This architecture is distinguished by its capabilities to allow dialogues between IoT devices using rational agents. This work focuses on the infrastructure of IoT services, while our work is not dependent on these kind of services. Moreover, the focus of [15] was on traditional reasoning and dialogue capabilities, whereas this paper focuses on advanced and DRL techniques.

Some researchers [14] have suggested an architecture based on autonomous technical assistance of agent-based objects (detection of failures) associated with safety and medical assistance in an ambient environment. However, that work did not consider machine learning, which is the focus of our approach.

3 Introduction to Machine Learning

In this section, we first present the best-known machine learning examples such as reinforcement learning (RL), deep learning (DL), and their combination deep reinforcement learning (DRL). Then, we introduce the MASs using DRL, where by communicating with the environment, each DRL agent learns to develop his own strategy and the other agents to attain rewards.

3.1 Reinforcement Learning

RL is defined as a type of ML algorithm that can optimally control a Markov Decision Process (MDP). RL presents ambient intelligence in IIoT systems by offering a class of sequential experience for processing sensory data in order to produce reaction control decisions. More specifically, "when an RL agent observes the current state s_t at each discrete time-step t, selects action a_t according to a policy π, receives the reward r_{t+1} thereafter, and transitions according to some probability distribution to a new state s_{t+1}" [5]. The agent's objective is to maximize their expected discounted rewards by interacting with an environment and learn from sequential experience. The expected discounted rewards is defined as follows: $R_t = r_t + \gamma * r_{t+1} + \gamma^2 * r_{t+2} + ...$, where $\gamma \in [0, 1]$ is a discount factor. So the action-value function is defined as following:

$$Q(s,a) = E^{\pi}[R_t|s_t = s, a_t = a] \tag{1}$$

3.2 Deep Learning

DL provides a subset of ML approaches that attempt to model with a high level of data abstraction by articulating architectures of various non-linear transformations. DL is a subset of techniques that exploit Artificial Neural Networks (ANNs) to learn autonomously from a large amounts of data. Significant and rapid progress has been made in the fields of audio and visual signal analysis, including facial recognition, speech recognition, computer vision, automatic language processing. DL is able to perform tasks such as regression and classification. There are many different DL architectures available in the literature such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Boltzmann Machine (BM), Long-short Term Memory (LSTM) Networks, Feedforward Deep Networks (FDN) and Deep Belief Networks (DBN).

3.3 Deep Reinforcement Learning

DRL is a subset of RL. Rather, it's the combination of DL and RL. As previously stated in section 3.1, for simple cases in the RL, it is possible to memorize and refine the reward estimator for all combinations of states or actions. But the problem arises when this number of possible states/actions takes huge proportions that a computer is no longer able to process. In this case it is necessary to find a more compact way than the exhaustive storage to store this function. This compression of information amounts to saying that certain sets of states/actions are actually similar and will produce the same reward.

This is where DRL comes in as a way of compressing information. The idea is to use neural networks as can be used in DL. These networks are extremely efficient for generalizing and approximating any function. The purpose of the game is, therefore, to use such a network which takes as input the current state and an action, and output an estimate of the reward that can be obtained. Once the network is trained, selecting the right action is like comparing the potential rewards of each one and choosing the best one.

The RL and DRL models for real-world intelligent IoT systems is not as simple as it might seem. The RL and DRL models are only interested in the environment and the agent. On the one hand, the RL/DRL environment can only reflect the physical system or be expanded to include wireless networks, edge/fog servers, and cloud servers. And it would be so because of the communication deadline, computing delay, power consumption, and network reliability, etc that will have significant consequences on the physical system's control efficiency. The control actions in RL/DRL can, therefore, be divided into two levels: control of actuators and control of resources. It is possible to separate or jointly learn and optimize the two levels of control. On the other hand, the agent in RL is an entity that makes decisions about action selection. In IIoT systems, the

intelligence agencies can reside in IoT devices, edge servers, fog servers, and/or cloud servers.

3.4 Multi-agent Deep Reinforcement Learning

In section 3.3, we aim attention at the DRL methods for single-agent cases. In reality, situations exist where many agents require to interact together to achieve the general aim of the system, e.g. in the multi-robot systems, in the network vehicles, in the cloud robotics, etc. DRL methods are conceived for multi-agent systems in these cases. A MAS is composed of multiple interacting agents within a common environment.

In multi-agent RL, by interacting with the environment to attain rewards, each agent learns to improve their own policy. The environment is generally dynamic, and the system can detect the problem of explosion space action. Because of many agents learn simultaneously, the optimal policy of itself may also change for a certain agent when the policies of the others change. This can affect the learning algorithm's convergence and cause instability. Using, therefore, independent learning agents is the simplest method to learning in multi-agent settings. For instance, "independent Q-learning is an algorithm where each agent learns its own policy independently, while the other agents are viewed as part of the environment" Independent Q-learning, nevertheless, can not handle the non-stationary environment problem. Throughout recent years, single-agent DRL techniques have been expanded to cases involving multiple agents.

4 Proposed approach using multi-agent DRL in IIoT

The IIoT presents different concepts of AI using the data needed to apply both timely analytics and informed action predicated based on this continuous data transmission. IoT software agents extend the capabilities of devices and software components across all levels of IoT device architectures for autonomous intelligent behavior [10]. We present, in this section, the description of a classic IoT architecture existing in the literature followed by our proposed architecture based on agents using the deep reinforcement learning.

4.1 Classic IoT architecture

A classic IoT system consists of four main blocks The principal objective in the process of layers design and implementation is to provide independence and flexibility [4]. The IoT architecture is layered, simple, able to integrate networked objects and transfer valuable data to the computer system, which can then be processed to give a concrete operational vision.

1. *Block 1*: It includes sensors and actuators. On the one hand, it consists of sensors capable of detecting physical parameters, gathering data from the environment, and identifying other intelligent objects. On the other hand, it consists of actuators that can influence environmental change.

2. *Block 2*: It contains the data from block 1 (from sensors). It is necessary to aggregate and convert these received data in analogue form to digital form.
3. *Block 3*: It contains the incoming pre-treated data for advanced processing analytic in the computational computing systems.
4. *Block 4*: In this block, the data on high-performance computer systems are processed, controlled, managed and stored.

We can also describe by analogy with the IoT blocks architecture, a four-layered architecture such as:

1. **The perception layer** presents essentially the physical autonomous layer of sensors and actuators interacting with the environment for data acquisition and control actions.
2. **The network layer** connects IoT communication network devices including wireless access networks, the Internet, and servers. It essentially transfers perception sensor data to the processing layer through networks. In other words, this layer allows the IoT devices to be discovered and connected for data and control command transmission to the edge/fog servers and cloud servers.
3. **The processing layer** stores, interprets and processes enormous data using different technologies, e.g big data... This refers to the commitment of data processing/storage and analysis actions by IoT edge/fog computing systems.
4. **The application layer** provides the client with application-specific services. This refers to the commitment of data processing/storage and control actions determination by IoT cloud computing systems.

In contrast, a fog-based architecture can also be implemented by cloud-based systems architecture. In this case, data processing is performed in a centralized manner by cloud computers, where the cloud is at the core of the system. This type of system, including infrastructure, network, platform, and storage provides excellent flexibility and scalability. The architecture of fog computing consists of control, pre-processing, storage, and security between the layers of physical and transport.

4.2 Proposed IIoT architecture

Our goal in this paper is to make an IoT (that is composed of many devices connected with the network) intelligent. But these dropping costs, combined with AI, mean that we are at the forefront of the next wave of the IoT, which is to make these devices detect, understand, adapt and react to the world. IoT devices became smarter, they can further manage real-time data processing, analytic, sensing, acting in the environment. The decision quality is improving, but most importantly, the timing of these decisions is also improving as well.

We propose an IIoT architecture as shown Figure 1 by integrating smart agents using ML to learn from their past experiences to improve system performance. Each layer is coupled with an intelligent agent(s) allowing devices and

machines to be able to make decisions autonomously and intelligently through ML. When this happens, humans can be excluded from the equation, generating value and profitability. In the IIoT architecture, therefore, one or more layers may be included in the environment. IoT devices, IoT edge/fog servers, IoT cloud servers, and IoT Internet gateway can be located by the agent(s).

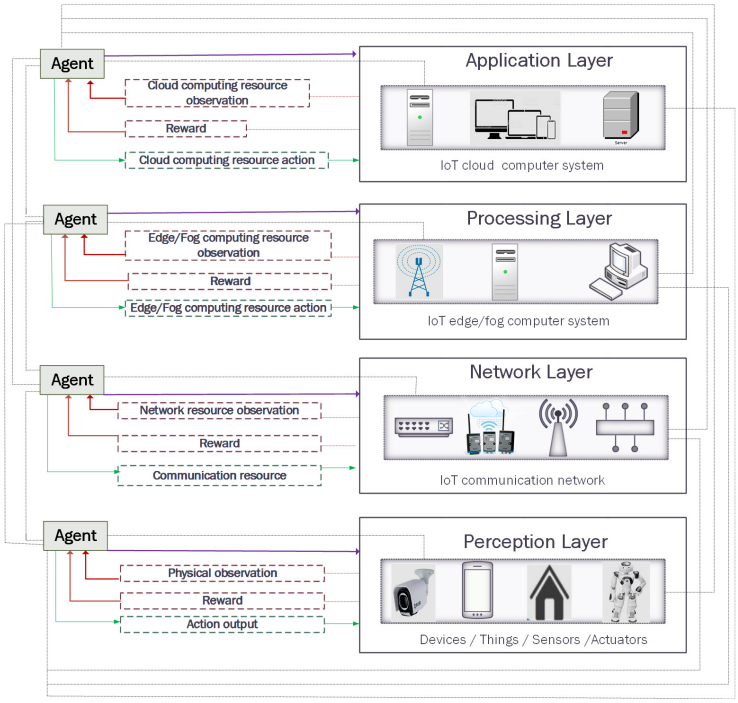

Fig. 1. Intelligent Internet of Things architecture based on agents using deep reinforcement learning

Perception layer Assuming the environment only contains the perception layer, the physical system is designed as a stochastic process where the state, action, and reward defined as follows:

– Physical observation (External device state) s_{layer_1}, for example, the observation in the time-step t o_t consists of the readings of the system RGB images o_t^{RGB}, the relative goal position o_t^g, and the current velocity of the robot o_t^v, the locations of the robot o_t^l, etc;

- Action output is a set of permissible actions in continuous space a_{layer_1}, for example, control of robot motion, switching on/off the device, moving a robotic arm, processing of workpieces, etc;
- Reward (physical system performance) r_{layer_1}: a reward feature is designed to direct a team of robots to achieve the goal of the system, for example, How easily a robot or a vehicle can move, or is it away from obstacles?

There is much research in the literature on DRL applications in the perception layer. We do not describe in this section in detail the existing work, but we will choose one for more explanations, such as the work of [8]. Where, the authors apply Deep Q-network DQN to the robot behavior learning simulation environment to allow mobile robots to learn how to achieve good behaviors, for example, avoid the wall and move along the centerline by using high-dimensional visual information as input data.

Network layer Assuming the environment only contains the network layer, so the network dynamic system is designed as a stochastic process where the state, action, and reward defined as follows:

- Network resource observation s_{layer_2}, for example, the bandwidth allocation, the signal to interference and noise ratio, etc;
- Communication resource action a_{layer_2}, for example, the planning of multiple users, etc;
- Reward (Network performance) r_{layer_2}, for example, the probability of transmission failure, the power consumption of transmission, etc.

Processing layer Assuming the environment only contains the processing layer, so the edge/fog computing dynamic system is designed as a stochastic process where the state, action, and reward defined as follows:

- Edge/fog computing resource observation s_{layer_3}, for example, the number of virtual machines at runtime, the fault detection and the state analysis of devices in assembly lines, etc;
- Edge/fog computing resource action a_{layer_3}, for example, the computational and storage services, etc;
- Reward (Edge/fog computing performance) r_{layer_3}, for example, the utilization rate of the Edge/fog computing resources, etc.

Application layer Assuming the environment only contains the application layer, so the cloud computing system is designed as a stochastic process where the state, action, and reward defined as follows:

- Cloud computing resource observation s_{layer_4}, for example, the number of task scheduling, the number of tasks for processing buffered in the queue, etc;

- Cloud computing resource action a_{layer_4}, for example, the selection of caches, the task offloading decisions, the virtual machine allocation, the task allocation, the resource allocation, etc;
- Reward (Cloud computing performance) r_{layer_4}, for example, the utilization rate of the cloud computing resources, etc.

Combination of all the layers Assuming the environment contains the four layers, so the DRL models typically contain elements (state, action, reward) described as following:

- IIoT state s_{IIoT} includes the aggregation of physical observation (external device state), network resource observation, edge/fog computing resource observation and cloud computing resource observation, i.e., $s_{IIoT} = \{s_{layer_1}, s_{layer_2}, s_{layer_3}, s_{layer_4}\}$;
- IIoT action a_{IIoT} includes the aggregation of action output, communication resource action, edge/fog computing resource action and cloud computing resource action, i.e., $a_{IIoT} = \{a_{layer_1}, a_{layer_2}, a_{layer_3}, a_{layer_4}\}$;
- IIoT reward r_{IIoT} is usually set to maximize the system performance that can be defined as a function of the network performance, edge/fog computing performance and cloud computing performance, i.e., $r_{IIoT} = r_{layer_1}(r_{layer_2}, r_{layer_3}, r_{layer_4})$.

In our model, since the agent in DRL is a software concept, the agent related to each layer can solve the DRL problem by observing the states and rewards of its environment and by applying learning strategies. These strategies aim at automatic learning of tasks by the gradual improvement of a function making it possible to estimate a future reward according to an action chosen at a given instant to determine the appropriate actions.

5 Conclusion

AI, IoT, ML, and agents are now considered to be highly advanced science that cover a variety of concepts and technologies. The basic computing machines have been evolved to smartphones, the portable devices to wearables, the cloud to IoT, that causes data management, data processing, and data analytic to move to the next tier. Nowadays, multi-agent systems play a crucial role in many fields, including robotics, healthcare, video game, cellular biology models and genomics, etc. The complex problems will be executed in virtually an autonomous and intelligent way without any kind of human intervention, and by combining MAS, IoT, and ML. For IoT, several agents can be combined in a highly collaborative and cooperative way to perform tasks. In this paper, we have proposed a new intelligent IoT architecture. The paper comprehensively reviews machine learning as well as the classic architecture of IoT systems. In the near future, we would like to demonstrate our approach by experiences the application of deep reinforcement learning methods in autonomous robots, especially for autonomous

car driving scenarios and our focus will also be to propose some novel and Intelligent IoT applications integrating the multi-agent systems with intelligent edge computing using machine learning. In addition, we will work on adapting existing formal approaches [9, 16–20] for testing and validating our framework.

References

1. P. Leong, and L. Lu, "Multiagent Web for the Internet of Things," In Proceedings of the International Conference on Information Science & Applications (ICISA), 2014, pp. 271–350.
2. A. M. Mzahm, M. S. Ahmad, and A. Y. C. Tang, " Towards a Design Model for Things in Agents of Things," In Proceedings of the International Conference on Internet of Things and Cloud Computing, 2016, pp. 3–7.
3. C. Savaglio et al., "Agent-Based Computing in the Internet of Things: A Survey," In Intelligent Distributed Computing XI, pp. 302–307, 2017.
4. D. B. Noureddine, A. Gharbi, and S. B. Ahmed, "An Approach for Multi-Robot System based on Agent Layered Architecture," International Journal of Management and Applied Science, ISSN: 2394-7926, Vol. 2, Issue. 12, 2016, pp. 135–143.
5. D. B. Noureddine, A. Gharbi, and S. B. Ahmed, "Multi-agent Deep Reinforcement Learning for Task Allocation in Dynamic Environment," In Proceedings of the 12th International Conference on Software Technologies (ICSOFT), 2017, pp. 17–26.
6. D. B. Noureddine, A. Gharbi, and S. B. Ahmed, "A Social Multi-agent Cooperation System based on Planning and Distributed Task Allocation: Real Case Study," In Proceedings of the 13th International Conference on Software Technologies (ICSOFT), 2018, pp. 483–493.
7. F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine Learning in IoT Security: Current Solutions and Future Challenges," In arxiv:1904.0573v1, 2019.
8. H. Sasaki, T. Horiuchi, and S. Kato, "A study on vision-based mobile robot learning by deep Q-network," In Proceedings of the 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), IEEE, 2017, pp. 799–804.
9. Afef Jmal Maâlej and Moez Krichen. A model based approach to combine load and functional tests for service oriented architectures. In *VECoS*, pages 123–140, 2016.
10. G. Fortino, "Agents meet the iot: Toward ecosystems of networked smart objects," IEEE Systems, Man, and Cybernetics Magazine, Vol. 2, Issue. 2, 2016, pp. 43–47.
11. S. Nolfi, J. Bongard, P. Husbands, and D. Floreano, "Evolutionary Robotics," ham: Springer International Publishing, 2016, pp. 2035–2068.
12. N. M do Nascimento, and C. J. P. de Lucena, "Engineering cooperative smart things based on embodied cognition," In Proceedings of the International Conference on Adaptive Hardware and Systems (AHS) NASA/ESA IEEE, 2017, pp. 109–116.
13. D. Marocco, and S. Nolfi, "Emergence of communication in embodied agents evolved for the ability to solve a collective navigation problem," Connection Science, Vol. 19, Issue. 1, 2007, pp. 53–74.
14. M. Zouai, O. Kazar, B. Haba, H. Saouli, and H. Benfenati "IoT Approach using Multi-agent System for Ambient Intelligence," International Journal of Software Engineering and its Applications, Vol. 11, Issue. 9, 2017, pp. 15–32.
15. J. C. Nieves, D. Andrade, and E. Guerrero, "MAIoT - An IoT Architecture with Reasoning and Dialogue Capabilityn," In Applications for Future Internet, Vol. 179, 2017, pp. 109–113.

16. Lahami, M., Fakhfakh, F., Krichen, M., Jmaiel, M.: Towards a ttcn-3 test system for runtime testing of adaptable and distributed systems. In: IFIP International Conference on Testing Software and Systems. pp. 71–86. Springer (2012)
17. Lahami, M., Krichen, M., Jmaiel, M.: Safe and efficient runtime testing framework applied in dynamic and distributed systems. Science of Computer Programming **122**, 1–28 (2016)
18. Krichen, M.: A formal framework for black-box conformance testing of distributed real-time systems. International Journal of Critical Computer-Based Systems **3**(1-2), 26–43 (2012)
19. Nathalie Bertrand, Amélie Stainer, Thierry Jéron, and Moez Krichen. A game approach to determinize timed automata. *Formal Methods in System Design*, 46(1):42–80, 2015.
20. Krichen, M.: Improving formal verification and testing techniques for internet of things and smart cities. Mobile Networks and Applications pp. 1–12 (2019)