How to Support the Machine Learning Take-Off: Challenges and Hints for Achieving Intelligent UAVs

Patrizio Dazzi(✉) and Pietro Cassarà

Information Science and Technologies Institute,
National Research Council of Italy, Pisa, Italy
{patrizzio.dazzi,pietro.cassara}@isti.cnr.it

Abstract. Unmanned Aerial Vehicles (UAVs) are getting momentum. A growing number of industries and scientific institutions are focusing on these devices. UAVs can be used for a really wide spectrum of civilian and military applications. Usually these devices run on batteries, thus it is fundamental to efficiently exploit their hardware to reduce their energy footprint. A key aspect in facing the "energy issue" is the exploitation of properly designed solutions in order to target the energy- and hardware-constraints characterising these devices. However, there are not universal approaches easing the implementation of ad-hoc solutions for UAVs; it just depends on the class of problems to be faced. As matter of fact, targeting machine-learning solutions to UAVs could foster the development of a wide range of interesting application. This contribution is aimed at sketching the challenges deriving from the porting of machine-learning solutions, and the associated requirements, to highly distributed, constrained, inter-connected devices, highlighting the issues that could hinder their exploitation for UAVs.

Keywords: Machine learning · UAV · Decentralized intelligence
Machine-to-machine · IoT

1 Introduction

Unmanned Aerial Vehicles (UAVs) [12] are arousing a growing interest from both industrial and scientific communities. This is mainly due to their flexibility: UAVs can be used for a really wide spectrum of civilian and military applications, either for supporting or replacing humans in dangerous and insalubrious environments.

Flexibility that is not only rooted in the large set of potential ways of using such vehicles, but also deriving from the almost endless possibilities of customization, personalization and configuration (i.e. UAVs range from very cheap drones to military planes, from devices having a reduced set of sensors to complete video or meteo flying stations).

P. Pillai et al. (Eds.): WiSATS 2017, LNICST 231, pp. 106–114, 2018.
https://doi.org/10.1007/978-3-319-76571-6_11

Even more, UAVs promise to have a tremendous impact on many areas, also from an economic perspective, many UAVs (even some powerful ones) are built by exploiting commodity hardware. It is quite common to see UAV which installed hardware depends on the specific goals to be pursued by that device. Typically, the preferred key for the hardware selection process is performance/energy trade-off [17].

In fact, usually this devices run on batteries, thus it is fundamental to efficiently exploit the hardware to reduce the energy consumption, e.g. using GPU, APU or accelerators that demonstrated to be very effective solutions in pursuing such goal.

Beyond a careful hardware selection, a key aspect in facing the "energy issue" is the exploitation of proper algorithmic solutions. Properly conceived, designed and implemented to target energy- and hardware-constrained devices and, more in details UAVs. To achieve this goal, many different approaches have been proposed so far [29,36], however two of the most promising strategies rely on approximation and collaboration.

The former as a way to reduce the energy footprint by accepting results that are not exact but still having enough significance to be useful and/or valuable for the purpose of the UAV. The latter consisting in an active collaboration occurring between UAVs to pursue altogether a common goal.

As matter of fact, there is no standard or universal approach and/or guideline to adopt for implementing the aforementioned strategies; it just depends on the class of problems to be faced. Much research effort need to be spent to innovate solutions in order to target UAV, in fact, as we mentioned before, due to their flexibility, there are many different application domains in which it is worth to exploit UAVs. In spite of this, there are a few classes of solutions that, if properly targeted to UAVs, would be beneficial for many different kinds of applications. Actually, such applications could fruitfully exploit these solutions to achieve an efficient, smart behavior of UAVs. Among them, there are a few machine-learning solutions that could be good candidates to that aim.

The aim of this paper is to outline the challenges that need to be faced to achieving machine learning solutions on a decentralized context and more in particular UAVs. Along the challenges, the paper provides some hints and suggestions as derived from the existing literature, borrowed from other scientific fields.

2 Challenges

Standard machine learning approaches require centralizing the training data on one machine or in a datacenter. However, recently, many approaches [22,23,30] have been proposed aiming to provide machine-learning capabilities on a decentralized scenario. In spite of this plethora of proposal, achieving an highly decentralized machine-learning requires to overcome many algorithmic and technical challenges.

In fact, with a typical machine learning system, an optimization algorithm, such as the Stochastic Gradient Descent (SGD) [8], runs on a large dataset appropriately partitioned across servers in a large datacenter or in a cloud. These algorithms require low-latency, high-throughput connections to the training data.

However, as can be easily noticed in a distributed scenario, data is distributed across a large set of devices in a highly unpredictable way. In addition, these devices have higher-latency, lower network bandwidth and are not always online, making them not always available to be involved in the training process. These bandwidth and latency limitations clearly motivate the need for properly defined approaches, tools, methodologies and protocols.

Achieving the computation of these algorithms by means of a large set of distributed, heterogenous and dynamic devices requires an innovative and sophisticated technology stack. Classical solutions need to be re-thought from this actual perspective. Design of algorithms, programming models, runtime supports, network stack, will require a paradigm shift to embrace the peculiarities characterising UAVs.

On device training need to exploit ad-hoc programming tools, properly tuned to the features and capabilities of the devices. Specifically conceived scheduling and runtime resource management ensures that training will happen only when the device is idle, fully charged, or on a high-capacity wireless connection, to limit the impact on the device performance. It is also fundamental that the system communicates and aggregates updates to the originally computed model in an efficient and fault-tolerant way. Communication shall be compressed and provided at irregular time intervals, the whole infrastructure supporting the interactions among UAVs and the computational back-end need to be properly tuned and designed.

Inter-UAVs connections and interaction are also important to achieve a more effective exploitation of the decentralized resources, on the one hand resulting into a reduced amount of data to be sent via the uplink but, on the other hand requiring a careful design of the interaction protocol.

2.1 Communication and Collaboration

Following from their distributed deployment, it is of paramount importance to properly design the communication processes occurring among "intelligent UAVs" to drive the information process exchange in a way that will allow to build a distributed knowledge to exploit for the jointly distributed decision making process. To this end many different approaches could be considered, ranging from technologies borrowed from Peer-to-peer computing, to approaches derived from Agent-based computing.

Smart UAV-2-UAV Computing. As aforementioned, one of the key requirements for enabling the development of smart/intelligent solutions, based on ML technologies, targeting fleets of UAVs, stands in the ability of leveraging the distributed knowledge owned by a fleet of UAVs as it was a unique knowledge base.

To this end, good candidates are the technologies originally developed in the context of P2P computing [2,9,21,24,34]. In particular, can be considered those approaches aimed at enabling a totally decentralized support for indexing and accessing data [16,28,33].

These approaches work by defining logic overlay networks, either structured or unstructured, depending by the degree of dynamicity characterising data, each aimed at supporting the distributed and decentralized execution of a specific set of operations, of various kinds, performed by relying on a subset of the peers composing the network. Regardless the limited involvement of the peers composing the network, the system is able to produce results that consider the whole set of data belonging to the entire system.

As two examples consider:

- a DHT indexes the data belonging to the entire system, then it is able resolve queries without involving the complete set of peers, but just a subset of them, without hindering the quality of the final result.
- a protocol like GROUP [3,4,10] is able to create homogeneous cluster of data, without any need of collecting all the pieces of information in a unique place.

These solutions demonstrated to be very effecting in allowing efficient and effective access to distributed data, often located in remote locations. Even more, these solutions are usually conceived, developed and optimized assuming unstable communications, churn (namely, nodes that disconnect from the network without providing any kind of notification).

A further interesting aspect of these technologies are their ability of managing very different kind of dynamic data, produced at a very different paces, depending on the application and context in which such information is generated.

As matter of fact **P2P computing-based approaches** could be good candidates for providing technological solutions supporting the achievement of Smart UAV-2-UAV computing, by enabling an efficient and effective communication between UAVs. In spite of the significant amount of research to be conducted to achieve a proper adaptation of these solutions to the UAVs, they can be a valid starting point.

2.2 Modelling and Orchestrating

Another important item in the achievement of a distributed, decentralized support to realize the smart UAVs, is the proper way to adopt for the overall modelling of the orchestration affecting the achievement of ML-based solutions. In particular, the way in which is such orchestration/interaction model the overall management of communication, collaboration and data exchange taking place among the UAVs. This is the keystone on which the decentralization is build and achieved. There are not much approaches in literature that specifically deal with such issues.

Federated learning [18,32] is probably the most effective approaches, existing so far, that is conceived and build on the idea of achieving a decentralized ML solution. It consists of a machine learning approach where the goal is to train models that are conceptually centralized but computed in a decentralized way, involving data distributed over a huge amount of computing nodes each with unreliable and relatively slow network connections. Federated learning considers learning algorithms in which, at each round, each node independently computes an update to the current model based on its local data, and communicates this update to a central server, where client-side updates are aggregated to compute a new global model.

In its current form and adoption, the typical clients are mobile phones, so communication efficiency is of utmost importance. In fact, in the paper in which such solution is presented, the authors discuss two ways to reduce the uplink communication costs able to reduce the upload communication required to train a reasonable model by two orders of magnitude w.r.t. other existing approaches.

Federated learning is nowadays the best candidate to be the starting point on which to build the orchestration and modelling of decentralized ML targeting UAVs. By adopting this approach, the ML models driving UAVs could benefit from a large knowledge base at the cost of a limited amount of communication.

2.3 Programmability

After defining the way UAVs follow to communicate one each others, and the information that are expected to exchange in this process, a further complex issue arise: How to program smart UAVs? Namely, how could be achieved their programmability without charging the programmers of the complex, error-prone, development of the entire process discussed so far, aimed at the management of the actual interaction occurring among UAVs? How this could be obtained in a high-level way?

In the scientific literature there is not much work specifically focused on UAVs, however, there are many interesting approaches dealing with pretty similar problems organized in a pretty similar way. It is the case of the solutions aimed at describing, programming and evaluating systems composed by large sets of interacting entities. There are essentially two main sectors in which these issues have been faced so far:

- Agent based models [6,14,19,26]. An agent-based model is a computational model aimed at simulating the actions and interactions resulting by the interplay of autonomous agents, focusing on the assessment of their effects on the system as a whole. It combines elements of game theory, complex systems and evolutionary programming.
- Multi-agent systems [11,25,31,35]. A multi-agent system is a computerized system composed of multiple interacting agents embedded with some form of intelligence, operating within a given target environment. Multi-agent systems are used to solve problems that are difficult for an individual agent or a single system to solve.

Both these approaches, can offer useful hints for the definition of programming models targeting fleets of smart UAVs. By following these agent-oriented views to organise the computation it could be possible to have a well-organized way to structure and organize the development process following a well-studied, high-level and highly-tested approach, that match the underlying deployment architecture of UAVs.

From a more technological viewpoint, one tool that could be considered for the programming of UAVs is Akka [7,13,27]. Akka is a free and open-source toolkit aimed at easing the construction of concurrent and distributed applications on the Java Virtual Machine (that is more and more supported also by the single-board devices installed on UAVs).

Akka supports multiple programming models for concurrency, but it emphasizes actor-based concurrency, essentially porting to an imperative-based languages level the features that were used to characterise Erlang [1].

2.4 Even More Efficient: Let's Go to the Edge

As aforementioned, in several ways and forms, the achievement of ultimate smart UAVs requires effective and efficient collaboration among UAVs, proper modelling of the algorithms and effective way to program them.

The overall, implicit, assumption has been that in this way UAVs can interact in a fully decentralized way to achieve their goal. As an alternative, we pointed out how federated learning could ensure, at cost of a very reduced communication footprint the generation (possibly storing it on a cloud) and assessment of a global ML model, resulting by the single actions performed by each single UAV. This essentially bi-partite the potential system organizations between: (i) totally decentralized models and (ii) cloud-based models.

However, recently is getting momentum the idea of blurring this distinction by introducing a hierarchy of computing devices standing in the middle between clients and the cloud. Depending on the pervasiveness of these additional devices, their aim and focus, we can distinguish between Edge [5,15] (additional devices only at the edge of the network) and Fog [20] computing (additional devices placed along all the network path from the cloud to the clients).

More specifically, it can be envisioned for fleets of UAVs the possibility of relying on properly defined and placed "base stations" aimed at caching data, performing complex computation, etc. The challenge will be to make these stations "invisible", meaning that UAVs (and their "application" programmers) do not need to be aware of the existence of such station; it would be the system itself able to manage the offloading of the computation, the management of the caching and all the activity requested to optimize the work of UAVs by means of the adoption of Edge computing.

The adoption of such deployment strategies can provide useful benefits in terms of energy saving and reduced network latencies.

3 Further, Non-functional, Challenges

Located at downstream of these challenges, there is also yet another key element to take into account: the actual testing of these solutions. Regardless the usual solutions, the behavior of these systems needs to be evaluated and assessed in a realistic, physical, environment before allowing the flight to the developed approaches.

One example of the current existing solution on this extent is the Flying Machine Arena (FMA) developed by the ETH of Zurich. FMA is a portable space devoted to autonomous flight. Measuring up to $10 \times 10 \times 10$ m, it consists of a high-precision motion capture system, a wireless communication network, and custom software executing complex algorithms for estimation and control.

The motion capture system can locate multiple objects in the space at rates exceeding 200 fps. While this may seem extremely fast, the objects in the space can move at speeds in excess of 10 m/s, resulting in displacements of over 5 cm between different snapshots. This information is then joined with other data and models related to the system dynamics in order to predict the state of the objects into the next future.

The system uses this knowledge to determine what should be the commands for the vehicles, that should be executed to achieve their desired behavior. Then, via wireless links, the system sends the commands to the vehicles, which execute them with the aid of on-board computers and sensors such as rate gyros and accelerometers.

Somehow this recalls the idea of federated learning, but in this context is not necessarily related to the achievement of the orchestration itself but mainly to its following evaluation and estimation.

4 Conclusion

In this paper we present a set of relevant challenges to make possible the decentralized execution of ML-based solutions on fleets of UAVs. Along of such challenges we also provide some suggestions about the potential solution to adopt for addressing such issues. As matter of fact a direct adoption may not be possible in all the cases, but represent a useful starting point for investigating and developing solutions addressing the aforementioned challenges.

The presentation mainly focused on the collaboration among UAVs, the definition of a decentralized ML model, the problem related to their programmability and the possibilities deriving from exploiting solutions based on Edge and Fog computing.

Finally the paper briefly observed that another key aspect in the achievement of ML-based solutions for UAVs is related to their actual evaluation, on an actual setup in a real environment. Even in this case the paper reports an existing solution that is worth to consider for the validation of next generation smart UAVs.

References

1. Armstrong, J., Virding, R., Wikström, C., Williams, M.: Concurrent Programming in Erlang. Prentice Hall, Hertfordshire (1993)
2. Baraglia, R., Dazzi, P., Guidi, B., Ricci, L.: GoDel: Delaunay overlays in P2P networks via gossip. In: IEEE 12th International Conference on Peer-to-Peer Computing (P2P), pp. 1–12. IEEE (2012)
3. Baraglia, R., Dazzi, P., Mordacchini, M., Ricci, L.: A peer-to-peer recommender system for self-emerging user communities based on gossip overlays. J. Comput. Syst. Sci. **79**(2), 291–308 (2013)
4. Baraglia, R., Dazzi, P., Mordacchini, M., Ricci, L., Alessi, L.: GROUP: a gossip based building community protocol. In: Balandin, S., Koucheryavy, Y., Hu, H. (eds.) NEW2AN/ruSMART 2011. LNCS, vol. 6869, pp. 496–507. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22875-9_45
5. Beck, M.T., Werner, M., Feld, S., Schimper, T.: Mobile edge computing: A taxonomy (2014)
6. Benenson, I., Torrens, P.M.: Geosimulation: Automata-based Modeling of Urban Phenomena. Wiley, Chichester (2004)
7. Bernhardt, M.: Reactive Web Applications: Covers Play, Akka, and Reactive Streams. Manning Publications Co., Greenwich, CT (2016)
8. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Lechevallier Y., Saporta G. (eds.) Proceedings of COMPSTAT 2010, pp. 177–186. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-7908-2604-3_16
9. Carlini, E., Coppola, M., Dazzi, P., Laforenza, D., Martinelli, S., Ricci, L.: Service and resource discovery supports over P2P overlays. In: International Conference on Ultra Modern Telecommunications & Workshops, ICUMT 2009, pp. 1–8. IEEE (2009)
10. Dazzi, P., Felber, P., Leonini, L., Mordacchini, M., Perego, R., Rajman, M., Rivière, É.: Peer-to-peer clustering of web-browsing users. In: Proceedings of LSDS-IR, pp. 71–78 (2009)
11. Ferber, J.: Multi-agent Systems: An Introduction to Distributed Artificial Intelligence. vol. 1. Addison-Wesley, Reading, MA (1999)
12. Gallington, R.W., Berman, H., Entzminger, J., Francis, M.S., Palmore, P., Stratakes, J.: Unmanned aerial vehicles. Future aeronautical and space systems (A 97–26201 06–31), Reston, VA, American Institute of Aeronautics and Astronautics, Inc., Progress. Astronautics and Aeronautics. **172**, 251–295 (1997)
13. Gupta, M.: Akka Essentials. Packt Publishing Ltd, Birmingham (2012)
14. Helbing, D.: Agent-based modeling. In: Social self-organization, pp. 25–70. Springer (2012)
15. Hu, Y.C., Patel, M., Sabella, D., Sprecher, N., Young, V.: Mobile edge computing-a key technology towards 5G. ETSI White Paper **11**(11), 1–16 (2015)
16. Kaashoek, M.F., Karger, D.R.: Koorde: a simple degree-optimal distributed hash table. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 98–107. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45172-3_9
17. Kestur, S., Davis, J.D., Williams, O.: BLAS comparison on FPGA, CPU and GPU. In: 2010 IEEE Computer Society Annual Symposium on VLSI, pp. 288–293, July 2010
18. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: strategies for improving communication efficiency (2016). arXiv preprint arXiv:1610.05492

19. Ligtenberg, A., Bregt, A.K., Van Lammeren, R.: Multi-actor-based land use modelling: spatial planning using agents. Landscape Urban plann. **56**(1), 21–33 (2001)
20. Luan, T.H., Gao, L., Li, Z., Xiang, Y., Wei, G., Sun, L.: Fog computing: Focusing on Mobile Users at the Edge (2015). arXiv preprint arXiv:1502.01815
21. Marzolla, M., Mordacchini, M., Orlando, S.: Resource discovery in a dynamic grid environment. In: Sixteenth International Workshop on Database and Expert Systems Applications, pp. 356–360. IEEE (2005)
22. McMahan, B., Moore, E., Ramage, D., Hampson, S., Aguera y Arcas, B.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282 (2017)
23. McMahan, B., Ramage, D.: Federated Learning: Collaborative Machine Learning without Centralized Training Data (2017)
24. Mordacchini, M., Dazzi, P., Tolomei, G., Baraglia, R., Silvestri, F., Orlando, S.: Challenges in designing an interest-based distributed aggregation of users in P2P systems. In: International Conference on Ultra Modern Telecommunications & Workshops, ICUMT 09, pp. 1–8. IEEE (2009)
25. Olfati-Saber, R., Fax, J.A., Murray, R.M.: Consensus and cooperation in networked multi-agent systems. Proc. IEEE **95**(1), 215–233 (2007)
26. Pahl-Wostl, C.: Actor based analysis and modeling approaches. Integrated Assessment, **5**(1) (2005)
27. Roestenburg, R., Bakker, R., Williams, R.: Akka in Action. Manning Publications Co., Greenwich, CT (2015)
28. Rowstron, A., Druschel, P.: Pastry: scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Guerraoui, R. (ed.) Middleware 2001. LNCS, vol. 2218, pp. 329–350. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45518-3_18
29. Sarma, S., Muck, T., Bathen, L.A.D., Dutt, N., Nicolau, A.: SmartBalance: a sensing-driven linux load balancer for energy efficiency of heterogeneous MPSoCs. In: 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), pp. 1–6. IEEE (2015)
30. Scardapane, S., Wang, D., Panella, M.: A decentralized training algorithm for echo state networks in distributed big data applications. Neural Netw. **78**, 65–74 (2016)
31. Schumacher, M.: Multi-agent Systems. Objective Coordination in Multi-agent System Engineering: Design And Implementation, pp. 9–32 (2001)
32. Smith, V., Chiang, C.-K., Sanjabi, M., Talwalkar, A.: Federated Multi-task Learning (2017). arXiv preprint arXiv:1705.10467
33. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: a scalable peer-to-peer lookup service for internet applications. SIGCOMM Comput. Commun. Rev. **31**(4), 149–160 (2001)
34. Trunfio, P., Talia, D., Papadakis, H., Fragopoulou, P., Mordacchini, M., Pennanen, M., Popov, K., Vlassov, V., Haridi, S.: Peer-to-peer resource discovery in grids: models and systems. Future Gener. Comput. Syst. **23**(7), 864–878 (2007)
35. Van der Hoek, W., Wooldridge, M.: Multi-agent systems. Found. Artif. Intell. **3**, 887–928 (2008)
36. Woithe, H.C., Kremer, U.: TrilobiteG: a programming architecture for autonomous underwater vehicles. In: Proceedings of the 16th ACM SIGPLAN/SIGBED Conference on Languages, Compilers and Tools for Embedded Systems 2015 CD-ROM, LCTES 2015, pp. 14:1–14:10, New York. ACM (2015)