Usage of Network Simulators in Machine-Learning-Assisted 5G/6G Networks

Francesc Wilhelmi, Marc Carrascosa, Cristina Cano, Anders Jonsson, Vishnu Ram, and Boris Bellalta

Abstract—Without any doubt, Machine Learning (ML) will be an important driver of future communications due to its foreseen performance when applied to complex problems. However, the application of ML to networking systems raises concerns among network operators and other stakeholders, especially regarding trustworthiness and reliability. In this paper, we devise the role of network simulators for bridging the gap between ML and communications systems. In particular, we present an architectural integration of simulators in ML-aware networks for training, testing, and validating ML models before being applied to the operative network. Moreover, we provide insights on the main challenges resulting from this integration, and then give hints discussing how they can be overcome. Finally, we illustrate the integration of network simulators into ML-assisted communications through a proof-of-concept testbed implementation of a residential Wi-Fi network.

Index Terms—Future Networks, ITU, Network Simulator, Machine Learning, Wireless Local Area Networks

I. INTRODUCTION

Beyond the fifth-generation (5G) of mobile communications systems, namely the sixth generation (6G), Artificial Intelligence (AI), and more precisely Machine Learning (ML), are expected to be pervasively included as part of the network operation, which would entail a huge leap towards optimization, automation, and self-healing. This is possible thanks to the paradigm shift driven by the softwarization of networks – achieved through Software Defined Networks (SDN) and Network Function Virtualization (NFV) – which provides the necessary flexibility to empower data-driven approaches.

The integration of ML to communications has started to be considered for the upcoming versions of 5G. This fact is supported by the content already approved by the 3rd Generation Partnership Project (3GPP) for Release 16 (2020) and Release 17 (2021) [1], which aim to continue improving the efficiency of 5G systems in many domains such as interference mitigation, power consumption, and user mobility, to name a few, and further push for Self-Organizing Networks (SON) with Big Data. Besides, we find of high relevance the contributions made by the International Telecommunication Union (ITU) Focus Group on Machine Learning for 5G and Beyond (FG-ML5G) and the Study Group 13 (SG13), which published specifications on an ML-aware architecture [2, 3].

Through the exploitation of the rich amount of available data, ML can overcome the systemic complexity inherited from novel use cases like Vehicle to Everything (V2X) communications, Machine Type Communications (mMTC), and

Francesc Wilhelmi, Marc Carrascosa, Anders Jonsson, and Boris Bellalta are with Universitat Pompeu Fabra (UPF); Cristina Cano is with Universitat Oberta de Catalunya (UOC)); Vishnu Ram is currently working as an independent researcher.

extended reality and high-quality video content delivery. These use cases comprise heterogeneous scenarios with mobility, a huge number of devices, and high-bandwidth and low-latency requirements. In particular, ML may offer substantial performance gains due to the inherent flexibility of automatically learning diverse situations, thus allowing to solve problems related to interference management, improving spatial reuse, or efficient resource allocation. The promising benefits of ML-enabled communications systems has been studied in [4]. We refer the interested reader to that work and the references therein.

While ML promises significant productivity gains, it also raises serious challenges and concerns. First of all, the successful application of ML depends on the quality of the available training data. An important challenge lies, therefore, in problems with limited or noisy data. Apart from that, dealing with non-stationary data is still an open challenge, which casts doubts on the validity of learned models. A prominent example is that of IEEE 802.11 Wireless Local Area Networks (WLANs). The typical decentralized nature of WLANs (e.g., residential deployments) complicates data collection procedures, and also leads to complex and highly non-stationary environments, where ML may fail to learn in time.

These challenges put into question the worthiness of introducing ML to networking systems. In particular, network operators and other stakeholders may have concerns regarding architectural (e.g., how to train and transfer ML models across a network) and operational aspects (e.g., how to provide trustworthy ML optimizations). While significant efforts have been put towards defining architectural solutions for ML-aware communications [1–3, 5], the implications of applying ML methods to networks have been barely studied. In this regard, we find an architectural element in [2] (the ML sandbox) for training and evaluating ML models in a safe environment. The ML sandbox has been defined to mitigate the potential side effects that ML optimizations can have on communications systems. Nevertheless, at the date of publishing this paper, the functionalities of ML sandbox remain unspecified.

In this paper, we devise the usage of network simulators as an important way to realize a sandbox in ML-aware communications systems. Network simulators play a crucial role both in academia and industry, and can potentially enable the paradigm shift towards ML-assisted communications. By prototyping complex problems and systems, simulators are key to evaluate new features and technologies, thus boosting innovation. In this regard, we believe that network simulators can contribute to providing reliable and robust ML mechanisms for communications. To the best of our knowledge, this is

the first work on addressing this emerging issue. The main contributions of this paper are as follows:

- We discuss the main aspects related to the reliability of ML for future communications.
- We devise the usage of simulators for training, testing, and evaluating the performance of ML models for communications.
- We showcase the potential integration of network simulators within the ITU ML-aware architecture, which is an adaptable and interoperable framework for realizing specific ML-based network functionalities.
- We provide insights on practical aspects for the integration of network simulators into ML-assisted communications systems.
- We illustrate the potential advantages of using simulators in ML-assisted networks by applying the outcome of an ML-driven simulation to a residential WLAN testbed.

II. RELIABILITY OF ARTIFICIAL INTELLIGENCE FOR COMMUNICATIONS

ML has been proved to offer outstanding results in many fields, but has also raised some concerns in terms of reliability and trustworthiness. The fact is that many ML mechanisms are seen as black boxes due to the non-linearity of their output (e.g., a prediction), especially for problems with high dimensional spaces. For instance, in Deep Learning (DL), the accuracy of a model is typically tied to its complexity in terms of the number of neurons and hidden layers. Although it is possible to get some insights on the behavior of a neural network (e.g., through visualization tools), its inherent complexity hampers interpretability, thus making outputs unpredictable.

In the telecommunications realm, ML has been successfully applied to multiple problems (see, for instance, the surveys in [6–10] and the references therein). Much of the credit resides in the extraction of useful information from large amounts of data. For instance, the authors in [8] show that autonomous Unmanned Aerial Vehicles (UAV) can be empowered by Artificial Neural Networks (ANN). In particular, on-time decisions such as the flying direction can be optimized based on the data collected (e.g., users' location, available resources, or wireless environment). This data, which may come from multiple sources, can be exploited and comprehended by the ANN for the sake of optimization.

Whereas ML can potentially improve the management and operation of networks, the uncertainty associated with the output of ML methods can lead to performance degradation. For instance, an online learning mechanism that is driven by exploration-exploitation may fail to comply with Service Level Agreements (SLAs). The fact is that exploration may lead to experiencing an undesired performance by triggering certain configuration settings. This is a critical aspect to take into consideration since many applications rely on certain minimum requirements to operate, and not meeting them could be even dangerous (for instance, consider networking applications for autonomous driving). As a result, the application of ML can raise concerns and lead to mistrust when applied to networks.

To address the potential lack of confidence generated by ML, we find two main state-of-the-art approaches: (1) *explainable AI* [11] (understand how ML algorithms behave), and (2) *safe Reinforcement Learning (sRL)* [12] (provide safe exploration mechanisms). Firstly, explainable AI is based on the interpretation of AI-based decisions, which is useful to devise the impact of potential optimizations and predict misbehavior. However, this field is not mature enough, and the existing techniques are mainly based on visualization, so they are subjective and may lead to misinterpretation. For that reason, explainable AI currently lacks applicability in ML-assisted communications, where the potential understanding of AI mechanisms should be directly translated into specific operations in the network.

Furthermore, sRL aims to minimize the negative effects that unconstrained exploration methods can produce during the learning procedure. This can be achieved either by adding extra information to the exploration mechanism (e.g., external advice) or by applying certain risk-aware criteria (e.g., exploration based on water-filling methods). While sRL is useful to mitigate the randomness of exploration, its application may provide moderate improvements and lead to slow optimization when applied to networks, which may be unacceptable in non-stationary systems. Besides, sRL is restricted only to RL mechanisms, and cannot be generalized to other approaches such as DL.

Given the lack of general mechanisms and procedures for providing trustworthy ML-aware communications, we devise the potential usage of network simulators for training, testing, and evaluating the effect of ML models before being applied to operative networks. In particular, simulators can provide diverse functionalities to enhance the confidence level of future ML-assisted networks:

1) **Validate the output of ML models:** a simulator can test and evaluate the output of a certain ML optimization before being applied to a production environment.
2) **Assess the impact of ML models on networks:** simulators can be used to study the effect that a given ML optimization has on the rest of the network. The whole procedure can be simulated together if the simulator includes ML functionalities, which is the case, for instance, of ns-3 and Komondor. To put an example, the authors in [10] propose an innovative end-to-end solution to jointly reconstruct the physical layer of a given transmission through DL. In this regard, the usage of simulators can help at devising the effects that the ML-driven approach may produce on the rest of the network layers.
3) **Generate training data:** sometimes, training data extracted from network devices can be sparse, limited, incomplete, or incoherent. To address this, simulators can generate synthetic data, which would broaden the available training data sets. However, assessing the quality of synthetic data sets can be challenging for operators, especially concerning complex problems that cannot be modeled accurately. For that reason, it is important to monitor the effects of applying ML models trained with synthetic data on operative networks.
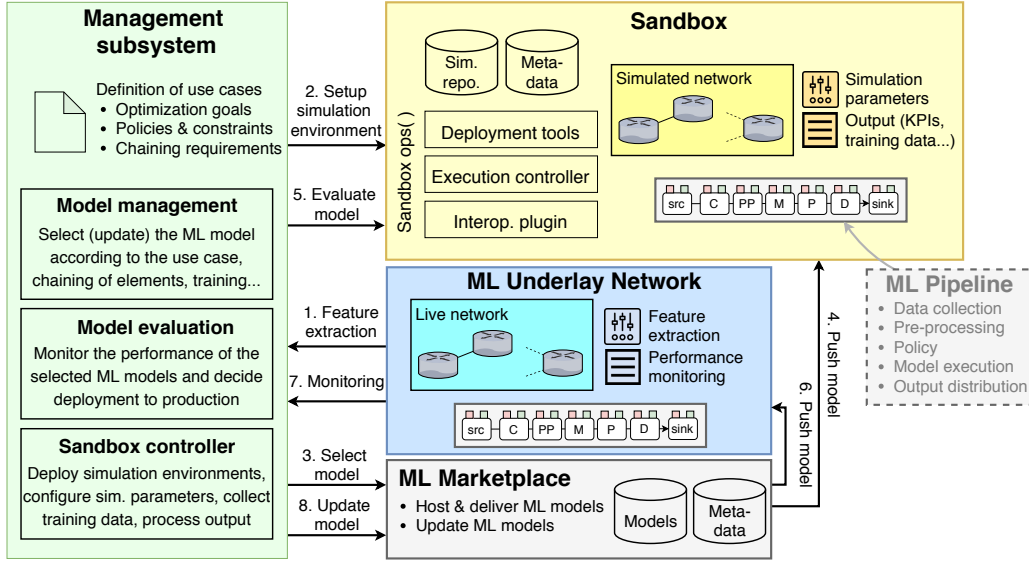
Fig. 1: Architectural elements and procedures for evaluating the output of ML models.

4) **Train ML models:** ML models can also be trained in a simulation environment. As an example, consider the case where online learning is simulated along with the network operation.

5) **Complement ML models:** simulators can also contribute to filling the intersection between model-based and data-driven approaches. The fact is that simulators can act as *experts* to assist the operation of ML algorithms. As an example, random initialization is typically employed for ML methods, which sometimes leads to converging to suboptimal saddle points. By adding additional knowledge from simulations, the learning procedure can be improved.

III. NETWORK SIMULATORS TO ENABLE ARTIFICIAL INTELLIGENCE IN COMMUNICATIONS

In this Section, we describe the architectural aspects of integrating network simulators into ML-assisted communications. Besides, we analyze the key requirements, challenges, and opportunities that emerge from the proposed integration.

A. Architectural Integration

Most of the existing simulation platforms have no relation with AI/ML techniques, nor have any integrated module for evaluating and training ML models. Moreover, current simulated network functionalities are typically too specific (e.g., simulate the effect of multiple antennas on the PHY layer performance), and seldom support open interfaces, as a result of being developed by focused academic or industrial organizations. To enable the next generation of ML-based communications systems, it is imperative to design interoperable mechanisms between network simulators and ML mechanisms. For that purpose, we find of high relevance the ITU ML architecture defined in [2].

The ITU ML architecture defines a set of logical components, interfaces, and procedures to realize ML-assisted communications. For a complete overview of the ITU architecture,

we refer the interested reader to the work in [4], which proposes a realization for future IEEE 802.11 WLANs, an important part of the 5G/6G ecosystem in unlicensed bands. In particular, the ML-aware architecture is composed of the following elements:

- **Management subsystem:** this element is responsible for the management and orchestration of the ML operation in a network. The responsibilities of this module range from data collection to ML model deployment and monitoring.
- **ML underlay network:** network at which the ML optimization is applied.
- **ML sandbox:** isolated domain for reproducing the behavior and operation of live networking systems.
- **ML marketplace:** container of ML models that are applied to operative or simulated underlay networks.
- **ML pipeline:** set of elements that interact with underlay networks to perform the ML optimization.

To integrate simulators into the loop of ML-assisted networks, we target interoperability as the principal driver. Interoperability allows building end-to-end ML pipelines in simulated network underlays, thus allowing the integration of network simulators in the ML-aware architecture. The fact is that we find a plethora of proprietary and open-source network simulators (e.g., ns-3, OMNET++, OPNET, NetSim, Komondor) for characterizing multiple types of scenarios, technologies, and network functionalities.[1]

To address interoperability, we devise the set of components and procedures to be held at the sandbox that is depicted in Fig. 1. In this regard, the management subsystem must deploy, configure, and interact with the simulated functionalities that are required by the ML use case. Notice that, given the diversity of simulation tools (stored and maintained in a

---

[1]Besides networking aspects, other specific phenomena can be simulated. For instance, Simulation of Urban MObility (SUMO) and UnderWater simulator (UWsim) simulate vehicular urban mobility and underwater physical effects, respectively, along with OPNET and ns-3 simulators.

repository), an interoperability plugin is required to translate simulator-specific commands into standardized operations.

The sandbox operations can be enabled by the softwarization of networks in 5G/6G systems. In the first place, this change of paradigm enables all the procedures related to data collection and processing, thus allowing to train and evaluate data-driven models in the sandbox. Besides, virtualized elements and standard interfaces can be reused to build and adapt the network in the simulation domain. For instance, the Open Radio Access Network (O-RAN) project provides an intelligent and virtualized RAN for open hardware. According to that, a RAN controller can be used to handle both the operative network and the equivalent simulation environment.

To illustrate the potential applications of simulators within the ML-aware architecture, Fig. 1 also provides an example where the output of an ML model is evaluated at the sandbox before being applied to the operative network. The involved procedures are as follows:

1) The management subsystem extracts features from the underlay network.
2) Based on the characteristics of the underlay network and the ML use case, the simulation environment is prepared through sandbox's deployment tools.
3) The management subsystem selects the ML model from the marketplace, according to the use-case metadata, the optimization goals, and the available models.
4) The ML model is pushed into the sandbox and applied to the simulated network.
5) The ML model is evaluated in the simulator. Evaluation of other ML models may be considered upon unsuccessful results.
6) Once the evaluation is done, the ML model is pushed into the operative network, where the ML optimization takes place.
7) The network performance is monitored, as well as new data is gathered.
8) The information obtained from monitoring is used to update the ML models and/or metadata in the marketplace.

The ML output evaluation procedure allows devising the potential benefits and drawbacks of using a certain ML model in a network. The fact is that ML outputs can sometimes look surprising from the perspective of a network operator, and their effect on the network may be unknown a priori. Unpredictability is further noticeable for complex problems from which there is limited knowledge.

B. Practical Integration Aspects

Integrating simulators into ML-assisted networks entails a set of challenges concerning execution, interoperability, and portability aspects.

First of all, to test, train, and evaluate ML models in simulators, it is important to reproduce the behavior of the target operative network. The main handicap lies in the plethora of existing simulation tools, each one with specific functionalities and execution requirements. In this regard, it is required to handle simulation-related metadata to maintain information such as the supported technologies and network functionalities,

the maturity of simulation blocks (e.g., beta release), and the potential number of domains the simulators can span (e.g., from the core to access network). Metadata can, therefore, enable the automated deployment and configuration of network simulators according to the ML use case requirements. For instance, an update of policies should be reflected in the simulation domain, so that operators' requirements can be fulfilled.

To carry out deployment operations, a great disadvantage is that simulators are written in multiple programming languages (e.g., C/C++, Java) and are supported by different specific platforms. In this regard, containerization (e.g., via Docker) can boost portability and allow network operators to deploy simulators flexibly. Apart from that, parallelization is important to determine, for instance, the number of ML pipeline nodes and simulated network functionalities that the simulator can support at any instant. With knowledge on supported capabilities, the simulated functionalities can be adapted to the use case. For instance, short execution and configuration times can serve to empower ML-driven real-time applications. First, we consider the time it takes for the simulator to generate a given output, which may indicate the tractability of simulating large-scale scenarios. Second, fast reconfiguration of network simulators would allow following potential changes on the operative network (e.g., user demands, available resources, policies, etc.).

Concerning pluggable ML functionalities, built-in ML modules can boost the procedures for simulating the behavior of ML mechanisms or training ML models in a simulator. A few existing simulators support ML functionalities. Two examples are the framework connecting ns-3 with OpenAI Gym [13], and the agent-based implementation in Komondor. The ns3-gym Gym framework exemplifies the interconnection of network simulators with ML platforms, and opens the door to new integrations with other well-known ML libraries such as TensorFlow or Acumos. In ns3-gym, the synchronization between the simulator and the ML components is achieved through a discretized mapping of the simulation's information (e.g., channel status, nodes number, traffic demands) with the states, actions, and rewards to be used by an ML algorithm. The whole procedure is carried out in execution time.

When it comes to interoperability, an important aspect lies in the degree of flexibility of simulators for interacting with the components of the ML-aware architecture. Interoperability is, therefore, meant to enable a seamless integration of intelligent network functionalities in the communication network. For that, it is imperative that the simulated network functionalities are managed using the same operation and maintenance mechanisms as for the network functionalities in the ML underlay. In this regard, the interoperability plugin is crucial to handle the different simulated networks in execution time, thus allowing for standardized functionalities such as *start* or *stop*. In particular, the features that may facilitate the interoperability of out-of-the-box simulators are the support for Command-Line Interface (CLI) execution mode, the level of monitoring supported (real-time, batch, model-based, etc.), automation of data collection, and in applying the ML output in the simulator (e.g., reading from log files vs. API-based

interface with ML functions).

C. Accuracy of Network Simulators

The degree of reliability of a network simulator depends on its accuracy on reproducing the actual phenomena. In other words, simulations must be as close as possible to reality. This topic was previously addressed in [14], where the authors defended that simulators do not really fit the actual behavior of networks, based on experimental results in a MANETs testbed. Nevertheless, it was also shown that simulation results can serve as a good upper-bound for testbed setups.

In general, network simulators accurately reproduce the behavior of protocols in higher levels of the TCP/IP stack. However, they can fail at characterizing complex physical phenomena such as radio propagation, antenna radiation, or energy consumption. As a result, network simulators typically provide accurate qualitative performance results, and help to predict the behavior of real networks under certain circumstances. In contrast, some results may lack quantitative precision, thus deviating from the exact performance that would be experienced in real networking systems. Alternatively, hybrid approaches can be employed for simulating certain layers (e.g., MAC) while taking advantage of the actual interactions that occur in real implementations. Unfortunately, and to the best of our knowledge, there is little literature on this topic.

IV. USE-CASE: POWER CONTROL IN RESIDENTIAL WLANs

To illustrate the potential of integrating simulators into ML-assisted networks, we provide a testbed implementation of an IEEE 802.11 WLAN that suffers from starvation due to the high sensed interference of a residential environment. To address this problem, a joint ML-based solution is simulated, validated, and then provided to the testbed devices.

A. From Testbed to Simulation Domain

The considered testbed implementation comprises two overlapping Basic Service Sets (BSSs) in a residential environment, which are characterized by being highly dense and uncoordinated. The decentralized nature of WLAN deployments in a neighborhood may lead to high interference, which can be extremely variable due to the heterogeneous usage of the network and the complex physical phenomena that can occur. The non-stationarity characteristic of residential environments is one of the most critical aspects to be considered when designing dynamic solutions for improving network performance. Hence, the usage of network simulators can contribute to reducing the performance losses originated by the transitory phases of training (e.g., exploration in online learning).

Our proposed testbed-simulator integration is illustrated in Fig. 2,where a tested ML solution is provided to the testbed devices by a simulated version of it. Two identical BSSs are deployed in a high-density residential scenario. However, since they are positioned at different locations, they are subject to different interference conditions, and so experience different performance. The characterization of the WLAN testbed is

done with the IEEE 802.11ax-oriented Komondor simulator, which includes the operation of agents for simulating the behavior of ML mechanisms when plugged into wireless nodes.[2]
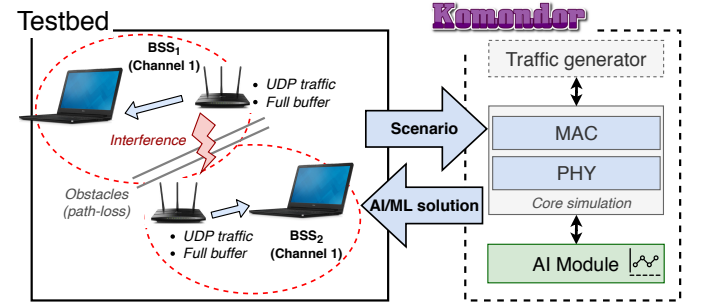

Fig. 2: Use case application of the Komondor simulator to apply ML in a testbed WLAN.

Through the procedures that have been previously illustrated in Fig. 1, the testbed scenario is first characterized in the simulator by gathering parameters such as the location of nodes, path-loss effects, or the traffic load. As an example of the characterization of the testbed in the simulator, consider the path-loss model selected, which is chosen based on the degree of similarity with respect to testbed measurements. After preparing the simulation environment, the ML model is applied in the simulator for the sake of improving a certain performance metric. Finally, the optimized ML-based configuration is passed and applied to the real devices whose performance is expected to be enhanced.

B. Machine-Learning-based Transmit Power Control

To improve the performance of the target WLAN, we simulate a Multi-Armed Bandits (MABs) mechanism for Transmit Power Control (TPC), as previously done in [15]. We take an online learning approach to address the complexity of spatial interactions in WLANs, where the effect of tuning the transmit power can be hindered. Accordingly, the MABs framework is useful to reduce the complexity of the problem and effectively improving the performance at a low computational cost.

This use case is particularly revealing since the transmit power is a critical parameter to be freely adjusted, and trying several configurations before finding the best performance may lead to unpredictable effects during the transitory regime. Moreover, commercial equipment typically offers a high delay when changing the transmit power or other parameters such as the primary channel. As a result, network simulators can play a crucial role in palliating the negative impact that exploration can have in communications.

Figure 3 illustrates the temporal throughput obtained by each BSS when simulating the MAB method for tuning the transmit power. Also, the performance that would be obtained by both BSSs when using the default configuration is illustrated. As shown, both BSSs experience an unstable transitory

regime before reaching a stable state whereby performance is improved. Among the set of input transmit power levels, the most popular one to be used by both BSSs is 7 dBm, which, based on simulation results, is expected to improve the average throughput by 88.48%.
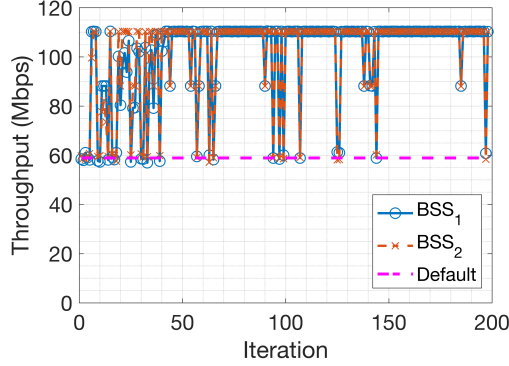

Fig. 3: Simulated throughput evolution after applying MABs for tuning the transmit power in an OBSS. Each learning iteration corresponds to 5 seconds in the simulation.

Finally, we provide some insights on the time it takes the simulator to bring up results for the testbed. To include the operation of simulators in future networks (especially for real-time applications), it is very important to find an equilibrium between the stability of the output and the time it takes to generate it. Figure 4 shows the variability of the simulation results, for different simulation time values. The execution time is also displayed. As observed, the higher the simulation time, the higher the stability is. However, this is paid with execution time, which varies for different network simulators.
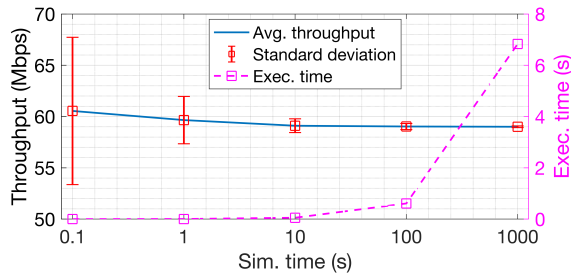

Fig. 4: Execution time versus variability of the results in Komondor simulator.

C. Testbed results

Now, we show the results of applying the configuration suggested by the simulator on the testbed. Figure 5 compares the performance of applying the ML-based configuration (both BSSs use a transmit power equal to 7 dBm) with that used by default (i.e., 23 dBm).

As shown, both BSSs improve their throughput significantly by using the configuration suggested by the simulator. While BSS_1 improves its throughput by 76.16 %, BSS_2 experiences a 93.98 % improvement. Besides, based on the lower number of observed outliers, we notice a higher stability in terms of throughput variability, especially for BSS_1. Note, as well, that BSS_2 experiences a higher number of outliers, which are originated by the high channel variability found in the residential environment where the tests were performed.
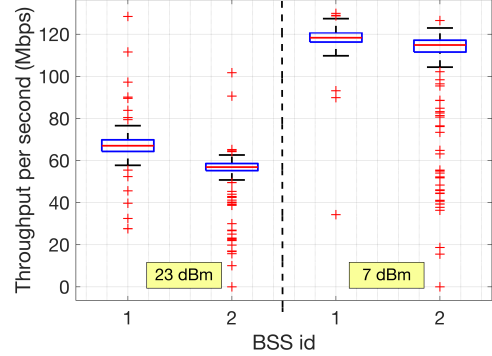

Fig. 5: Performance comparison of default (23 dBm) and ML-based (7 dBm) configurations at the testbed WLAN.

V. CONCLUDING REMARKS

Future communications are expected to evolve towards automated systems enabled by ML. However, the application of ML to networking systems can generate instability and degrade KPIs. To address that, we envision the integration of sandbox environments for ML-assisted networks. In particular, we find network simulators of great utility for training, testing, and evaluating the performance of ML models before being deployed to production environments. In this article, we proposed the utilization of network simulators for future ML-based communications and provided an architectural integration for improving the level of confidence in ML within ML-aware network architectures. Besides, we provided insights into the most prominent challenges resulting from such an integration. Finally, our testbed results in a residential IEEE 802.11 WLAN showed how network simulators allow mitigating the negative effects of directly applying ML in the operative network.

Network simulators are expected to contribute to filling the gap between AI and communications. Nevertheless, a lot of effort is still needed with regards to the architectural integration of simulators into ML-assisted networks. The most important challenges lie in the definition and implementation of standardized interfaces. Concerning practical implementations, we left as future work the further evaluation of ML models in network simulators. In particular, different ML models and simulation tools can be compared in terms of output accuracy, execution time, or computation needs.

REFERENCES

[1] 3GPP TR 23.791 V16.2.0 (2019-06), "Study of Enablers for Network Automation for 5G," 2019.

[2] ITU-T Rec. Y.3172, "Architectural framework for machine learning in future networks including IMT-2020," 2019.

[3] ITU-T Rec. Y.3174, "Framework for data handling to enable machine learning in future networks including IMT-2020," 2019.

[4] Wilhelmi, F. et al., "A flexible machine-learning-aware architecture for future wlans," *IEEE Comm. Mag.*, vol. 58, no. 3, pp. 25–31, 2020.

[5] ETSI GS ZSM 002 V0.13.5 (2019-07), "Draft Zero-touch network and Service Management (ZSM); Reference Architecture," 2019.

[6] Klaine, P. V. et al., "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Comm. Surveys & Tutorials*, vol. 19, no. 4, pp. 2392–2431, 2017.

[7] Zhang, C., Patras, P. and Haddadi, H., "Deep learning in mobile and wireless networking: A survey," *IEEE Comm. Surveys & Tutorials*, vol. 21, no. 3, pp. 2224–2287, 2019.

[8] Chen, M. et al., "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Comm. Surveys & Tutorials*, vol. 21, no. 4, pp. 3039–3071, 2019.

[9] Liang, L. and Ye, H. and Li, G. Y., "Toward intelligent vehicular networks: A machine learning framework," *IEEE IoT Journal*, vol. 6, no. 1, pp. 124–135, 2018.

[10] O'Shea, T. and Hoydis, J., "An introduction to deep learning for the physical layer," *IEEE ToCCN*, vol. 3, no. 4, pp. 563–575, 2017.

[11] Samek, W., *Explainable AI: interpreting, explaining and visualizing deep learning*. Springer Nature, 2019, vol. 11700.

[12] Garcıa, J. and Fernández, F., "A comprehensive survey on safe reinforcement learning," *JMLR*, vol. 16, no. 1, pp. 1437–1480, 2015.

[13] P. Gawłowicz and A. Zubow, "ns-3 meets OpenAI Gym: The Playground for Machine Learning in Networking Research," in *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2019, pp. 113–120.

[14] Cavin, D., Sasson, Y., and Schiper, André, "On the accuracy of manet simulators," in *Proceedings of the second ACM international workshop on Principles of mobile computing*, 2002, pp. 38–43.

[15] Wilhelmi, F. et al., "Potential and pitfalls of multi-armed bandits for decentralized spatial reuse in WLANs," *JNCA*, vol. 127, pp. 26–42, 2019.

Francesc Wilhelmi (francisco.wilhelmi@upf.edu) holds a B.Sc. degree in Telematics Engineering (2015) and an M.Sc. in Intelligent and Interactive Systems (2016), both from Universitat Pompeu Fabra (UPF). He is currently pursuing a Ph.D. in Information and Communication Technologies at UPF.

Marc Carrascosa (marc.carrascosa@upf.edu) obtained his B.Sc. degree in Telematics Engineering (2018) and a M.Sc. in Intelligent and Interactive Systems (2019) from Universitat Pompeu Fabra (UPF). He is currently a PhD student in the Wireless Networking Research Group in the Department of Information and Communication Technologies (DTIC) at UPF. His research interests are related to performance optimization in wireless networks.

Cristina Cano (ccanobs@uoc.edu) holds a Ph.D. (2011) in Information, Communication and Audiovisual Media Technologies from Universitat Pompeu Fabra (UPF). She has been a research fellow in the Hamilton Institute of the National University of Ireland, Maynooth (2012-2014), in Trinity College Dublin (2015-2016) and in Inria-Lille in France (first half of 2016). Currently, she is an associate professor at Universitat Oberta de Catalunya (UOC).

Anders Jonsson (anders.jonsson@upf.edu) is the director of the Artificial Intelligence and Machine Learning group at Universitat Pompeu Fabra (UPF). He received his Ph.D. in computer science in 2005 from the University of Massachusetts Amherst, USA, and has been at UPF ever since.

Vishnu Ram (vishnu.n@ieee.org) worked for Motorola/Nokia/Siemens in advanced technologies teams for 21 years. He was a Scientific Advisory Board Associate (SABA) member of Motorola Networks. He has published several drafts in IETF, contributed to ETSI, 3GPP in his role as a senior specialist (Radio Resource Management). He is currently working as an independent researcher.

Boris Bellalta (boris.bellalta@upf.edu) is an Associate Professor in the Department of Information and Communication Technologies (DTIC) at Universitat Pompeu Fabra (UPF). He is the head of the Wireless Networking research group at DTIC/UPF.