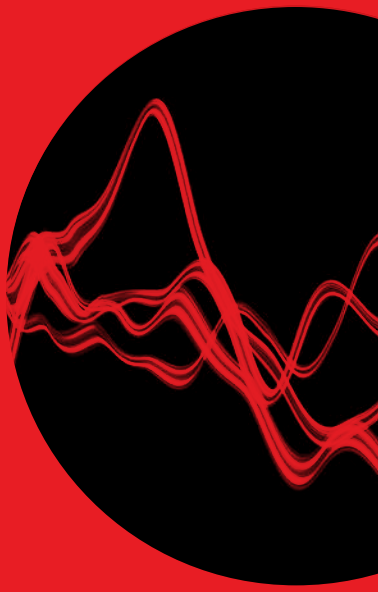
CHARTING THE FUTURE OF INNOVATION | #3·2023

ERICSSON TECHNOLOGY Review



APPROACHING AI-NATIVE RADIO-ACCESS NETWORKS

ERICSSON

Approaching AI-native RANs through generalization and scalability of learning

"AI native" is a trending concept that is gaining momentum in many industries, and ours is no exception. To maximize the benefits of making radio-access networks more AI-native while simultaneously minimizing complexity and cost, we propose a stepwise approach based on generalization and scalability of learning.

PABLO SOLDATI,
EUHANNA GHADIMI,
BURAK DEMIREL,
YU WANG,
MATHIAS SINTORN,
RAIMUNDAS GAIGALAS

A holistic vision of an AI-native radio-access network (RAN) would be a system designed for artificial intelligence (AI) algorithms, in which a single AI algorithm could learn and govern most networking operations, ranging from the physical layer to Radio Resource Management (RRM).

■ As appealing as a holistic vision of an AI-native RAN might seem, making it a reality would almost certainly break the logical boundaries of traditional communication protocols. It is also likely to be hampered by telecom industry regulations and/or the constraints imposed by the standardized systems – those that are compliant with the 3GPP (3rd Generation Partnership Project) and the IEEE (Institute of Electrical and Electronics Engineers), for example – that are used today to ensure the interoperability of radio communications across national borders and system vendors.

Ericsson has chosen to take a less abrupt approach to creating more AI-native RAN systems by embracing design principles and concepts that enable the integration of AI into RAN systems with minimal disruption. While we agree that AI algorithms should be at the center of future RAN design, we believe the paradigm shift should occur gradually within the traditional boundaries of RAN systems (RAN protocol layers and hierarchy). We think it is also important to recognize that not all RAN design aspects need AI.

Ericsson initially deployed AI for network design and hyperparameters optimization of RAN algorithms a few years ago [1], using AI to fine-tune well-established RAN algorithms whose automation would otherwise be complex and costly. One example is optimizing the block error rate (BLER) target of downlink link adaptation (DLLA) in Ericsson 4G systems.

Academic and industrial research indicates that AI could be helpful for virtually every RAN design aspect [2]. However, most of this research considers

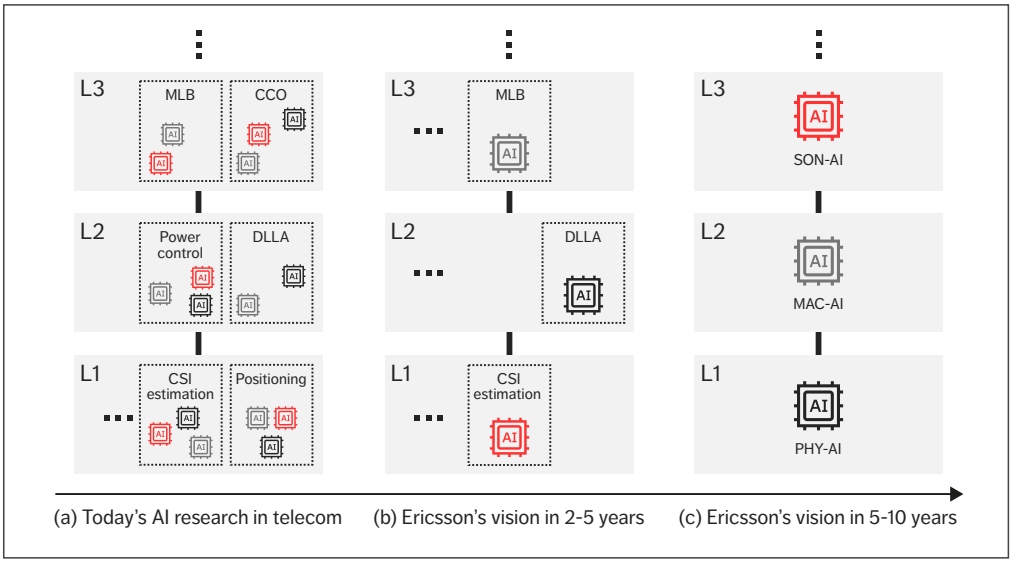


Figure 1 Ericsson's approach to integrating AI into a RAN over the next 10 years

specific RAN problems in isolation, with AI tailored to fit specific conditions or environments, such as those in the cell where training data is collected. We refer to such an approach as AI model specialization. While specialized AI models enjoy simple state reconstruction (few model inputs, for example) and less computational complexity, they do so at the cost of not having the robustness to cope with the diversity of RAN deployments and environments.

Our concern is that the adoption of solutions based on specialized AI models in live RANs could lead to an uncontrolled proliferation of AI models per radio feature, as shown on the left side of *Figure 1*, which would increase both RAN

Terms and abbreviations

3GPP – 3rd Generation Partnership Project | **AI** – Artificial Intelligence | **BLER** – Block Error Rate | **CCO** – Coverage and Capacity Optimization | **CDF** – Cumulative Distribution Function | **CRAN** – Cloud RAN | **CSI** – Channel State Information | **CSP** – Communication Service Provider | **CU** – Central Unit | **DLLA** – Downlink Link Adaptation | **DQN** – Deep Q-Networks | **DU** – Distributed Unit | **GNN** – Graph Neural Network | **HARQ** – Hybrid Automatic Repeat Request | **KPI** – Key Performance Indicator | **LCM** – Life-cycle Management | **MAC** – Medium Access Control | **MAC-AI** – Medium Access Control based on Artificial Intelligence | **MIMO** – Multiple-input, Multiple-output | **ML** – Machine Learning | **MLB** – Mobility Load Balance | **mMIMO** – Massive MIMO | **MSRBS** – Multi-standard Radio Base Station | **NF** – Network Function | **QoS** – Quality of Service | **PHY** – Physical Layer | **PHY-AI** – Physical Layer based on Artificial Intelligence | **RAN** – Radio-Access Network | **RL** – Reinforcement Learning | **RLE** – RAN Learning Engine | **RRM** – Radio Resource Management | **SEED RL** – Scalable and Efficient Deep RL | **SL** – Supervised Learning | **SON-AI** – Self-Organizing Networks based on Artificial Intelligence | **UE** – User Equipment

> WE HAVE IDENTIFIED TWO KEY ENABLERS: AI MODEL GENERALIZATION AND A SCALABLE AND VERSATILE LEARNING ARCHITECTURE

complexity and operational expenditure. Additionally, the coexistence of multiple AI-driven RAN features that are independently designed and expected to operate at the same timescale (that is, residing in the same protocol layer) and to learn from the same environment (the same cells, for example), would make the learning process difficult and potentially unstable.

The middle section of Figure 1 shows Ericsson's vision for integrating AI in RAN systems in the next 2-5 years. We are currently working to replace legacy RAN algorithms with AI counterparts that are designed to generalize across different radio conditions and environments, network deployments, types of traffic, RAN intents and so on. The intention is for these new AI algorithms to be reusable across the network. In this phase, we are focusing on few key RAN operations that are best suited to capitalize on AI – that is, those with the highest potential to boost performance.

In the longer term, which is illustrated on the right side of Figure 1, we aim to broaden the scope and reach of an individual AI algorithm to jointly solve multiple RAN operations. Initially, this could lead to the creation of a single generalized AI algorithm per RAN protocol layer that is designed to jointly learn and govern the most relevant (but not necessarily all) operations of its protocol layer.

Reliance on the traditional RAN protocol hierarchy ensures the correct separation of AI algorithms based on the timescale of RAN operations. Identifying which RAN operations within a protocol layer are the most relevant to be jointly addressed with AI is instrumental to avoid the risk of increasing the complexity of RAN products for diminishing returns. For instance, a generalized AI algorithm for the medium access control (MAC) layer may be designed to jointly control only the few most critical MAC layer operations, such as scheduling and link adaptation, but not necessarily power control, beamforming, precoding and so on.

Looking further ahead, the integration of AI technology in RAN systems may become closer to the holistic vision of an AI-native RAN where a single AI agent may learn to control several (if not most) RAN operations, possibly bridging across the traditional protocol layers. The feasibility and the blueprint of this vision is uncertain as it relies on the evolution of several technological areas, ranging from AI algorithms to computing, storage and RAN technologies. Furthermore, the telecommunication industry may also need to embrace more advanced and disruptive design concepts, that may lead to revisiting some traditional design pillars that have shaped our industry for more than three decades.

In our efforts to conceptualize a more AI-native RAN in the nearer term, we have identified two key enablers: AI model generalization and a scalable and versatile learning architecture.

Enabler 1: AI model generalization

A fundamental goal of machine learning (ML) is to achieve model generalization – that is, to train a function that can cope with conditions not directly observed during training yet inferable from training data. For AI in RAN systems, we extend this concept into three generalization domains:

1. RAN environment
2. RAN intents
3. RAN control tasks.

While each generalization domain brings us one step closer to a more AI-native RAN, the untapped potential of AI resides in a design that integrates all generalization dimensions, paving the way toward the vision of a single AI algorithm governing and replacing multiple RAN operations at once.

Generalization over the RAN environment

Training AI models to generalize over the RAN environment, rather than specializing to fit specific conditions (such as a cell environment), provides the necessary robustness to deploy a single AI model for a given task across the entire network. This improves scalability and reduces the complexity of the AI functionalities and operations. For example, model life-cycle management (LCM) becomes less complex when there is only one AI model per radio feature or RAN protocol layer (which is ideal).

AI generalization unlocks the potential for a single AI model to learn from the distributed experience of thousands of network entities. Distributing training-data generation over space (with thousands of network entities each contributing a few training samples) and time drastically reduces the impact of the learning process on RAN performance, such as for reinforcement learning (RL) algorithms taking suboptimal actions during training in live networks.

The RAN environment can be characterized by a combination of static and semi-static information that describes the network deployment and configuration, along with more dynamic information about the radio environment, traffic, load, specific user equipment (UE) conditions and so on. At Ericsson, we have investigated two main design approaches for generalizing AI models over the RAN environment. The first relies on traditional feature engineering, where domain knowledge is applied to identify the most critical information to reconstruct the RAN environment for specific AI applications. Techniques such as feature sensitivity analysis make it possible to filter out redundant information (based on correlation metrics, for example) to reduce model complexity.

A more systematic approach is to separately learn an embedding of static and semi-static RAN environment characteristics and combine them with dynamic RAN environment information that is specific to individual use cases. Graph neural networks (GNNs) [3] are a prime candidate for embedding the static and semi-static aspects of the RAN environment. These include network configurations and relations between different RAN entities (such relations between one cell and a set of direct or indirect neighboring cells), which may characterize wider relations between network entities beyond the reach of traditional feature-engineering approaches. Such embedding could be learned and reused across various use cases (from L1 to L3, for example). At Ericsson, this technique has been investigated in the context of L3 functionalities, such as secondary carrier prediction and antenna tilt control for coverage and capacity optimization (CCO).

Generalization over RAN intents

RAN performance cannot be defined by a single key performance indicator (KPI) such as throughput, spectral efficiency, latency, packet loss, reliability or resource utilization. Instead, it is defined using multiple, sometimes competing, KPIs. Therefore, many control problems in RAN, such as RRM, do not offer a single optimum-for-RAN KPI but rather a Pareto frontier of RAN KPIs, each trading off against another.

Ericsson's approach to generalizing the AI model for RAN intents includes the use of multi-objective RL, which enables a single AI model to learn the Pareto frontier of multiple RAN KPIs. In execution, the same AI model could be used to achieve the best KPIs to meet specific requirements of use cases, such as fulfilling different quality-of-service (QoS) requirements (including 5G QoS Identifier values) for different UE traffic types.

Generalization over RAN control tasks

Generalization over RAN/RRM control involves designing an AI algorithm to jointly control multiple transmission parameters by solving multiple RAN/RRM tasks simultaneously. This is instrumental in reducing the number of AI-based control loops operating at the same timescale and ultimately realizing our vision of a single AI algorithm per RAN protocol layer.

At the MAC layer (L2), for instance, a single AI algorithm, rather than two, could jointly solve resource scheduling and link adaptation. However, solving broader multi-decision-making problems

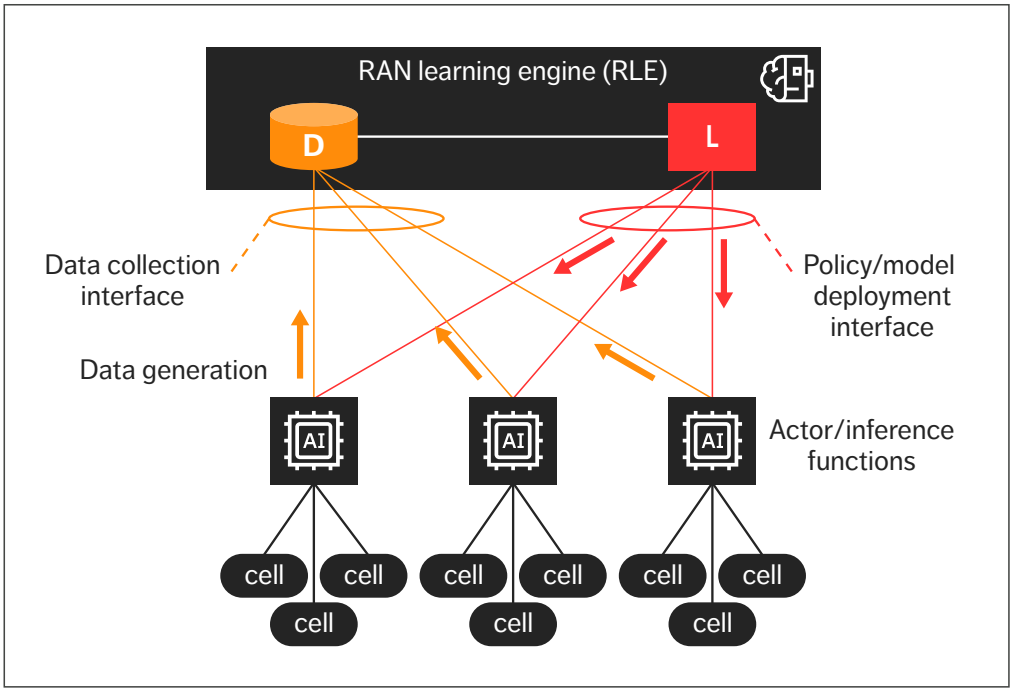


Figure 2 Our vision for the generic functional framework of a RAN learning architecture

with a single AI algorithm is only viable if it leads to a substantial performance boost, as it requires solving a more complex problem within stringent time limitations for the execution of AI models and entails more advanced hardware capabilities.

Enabler 2: A scalable, versatile learning architecture

The second enabler that we consider to be essential to realizing a more AI-native RAN is an scalable, versatile learning architecture that is, possibly, agnostic to the specific type of AI algorithms to be trained, such as RL and supervised learning (SL). Designing AI algorithms to generalize in any of the previously mentioned domains poses new challenges and opportunities. In addition to designing an appropriate AI algorithm to solve the task at hand, a major challenge for AI generalization in RAN is finding a scalable solution for training the AI model, as well as for the generation, collection and management of the necessary training data.

Since the publication of the seminal DQN algorithm [4], several architectural works have gradually addressed the scalability and efficacy of the learning for RL algorithms [5-10]. The Impala architecture [7] achieved the first milestone by exploiting off-policy learning combined with a decoupled distributed actor-learner architecture to capitalize on, rather than suffer from, the lag between when the actors generate actions and when the learner estimates the gradient. A distributed RL architecture advocating a similar actor-learner decoupling for the RAN applications was independently proposed [11]. Such principles have recently been further explored by other RL architectures, including APEX [8], R2D2 [9] and

SEED RL [10], which pushed the boundaries of learning scalability and efficacy to new levels.

Despite being referred to as distributed RL architectures, these methods exploit the actor-learner decoupling through a single centralized learning function supporting several distributed actor functions, all of which contribute to learning a single RL policy. (Note that because an RL policy is typically represented by a functional approximator, such as an AI model, the terms RL policy and AI model are often used interchangeably when referring to RL algorithms.)

Actors asynchronously receive a policy update and evaluate the policy, possibly on several parallel environments (in parallel simulations, for example). This massively scales policy evaluation and training data generation, enabling more frequent policy updates, which leads to faster and more efficient learning. For instance, the R2D2 architecture [10] achieved a fivefold to twentyfold performance improvement over traditional DQN with a corresponding two-and-a-half fold to fiftyfold training time reduction, respectively.

Our solution to address the challenges of AI generalization is a RAN learning architecture inspired by the most advanced RL architectures, as illustrated in ***Figure 2***. There are two main elements:

1. A centralized RAN learning engine (RLE) function
2. Distributed actor functions (for control purposes, for example) and/or inference functions (for insight generation, for example).

Despite its resemblance to some RL architectures [8], the RAN learning architecture in Figure 2 offers a versatile functional framework with a broader AI algorithmic scope, ranging from RL algorithms for RAN control to SL to provide RAN insights. It also supports unsupervised learning, offline RL and/or transfer learning by relaxing the typical conditions of the RL learning loop. Even federated learning could be supported with some functional modification based on the Gorila architecture [5], for example.

The RLE is a centralized support function that provides AI learning services and data services to different AI-driven RAN features. These "AI features" are executed by distributed actor/inference functions, possibly at different RAN protocol levels and timescales, by means of two interfaces for model/policy deployment and data collection, respectively. A natural implementation of the RLE function is in a cloud environment, for both multi-standard radio base station (MSRBS) systems and cloud RAN (CRAN) systems hosting actor/inference functions.

The deployment of actor/inference functions executing an AI feature in the RAN architecture is use-case specific – for instance, internal to RAN network functions (NFs) like a gNB central unit (CU) for mobility use cases or a distributed unit (DU) for PHY and MAC use cases. On the other hand, the centralization and unification of the AI support functionalities within the RLE makes its deployment in the RAN architecture use-case agnostic, thus allowing the flexibility needed for a lean integration of AI technology in RAN systems, with reduced complexity and operational expenditure.

The RLE function is responsible for the entire learning and data management pipeline. For ease of description, we decouple its functionalities into two main categories: a centralized learning engine (L) and a centralized data engine (D). The centralized learning engine provides AI learning services (including training, validation and testing, as well as AI model LCM management services) for actuation/inference functions deployed in RAN nodes. The learning engine may also control the generation of training data (including exploration) from the underlying distributed actor/inference functions. The centralized data engine provides data services related to the storage and management of training, monitoring and other types of data; the storage and version control of algorithmic objects (including trained models and learning data structure); the storage and version control of unit tests; feature store functionality and so on.

Actor and inference functions are broadly responsible for AI model execution, the generation and collection of training data, the execution of unit tests and some AI model LCM support

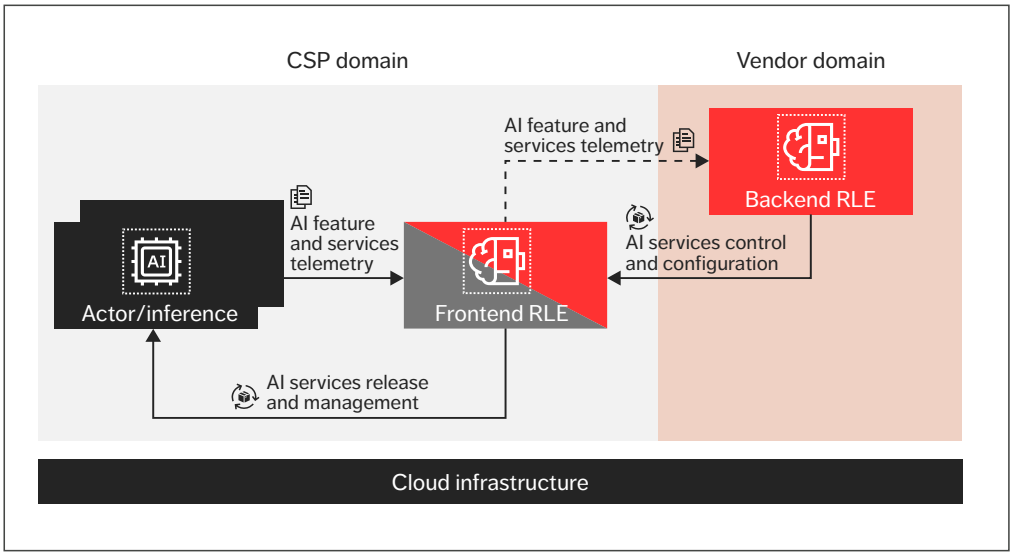


Figure 3 Example of RAN architecture deployment in a cloud infrastructure shared between vendor and CSP

functionalities, such as local performance monitoring, model inaccuracy detection and fallback operations. However, certain AI algorithms will require specific functionalities. An RL actor, for instance, would handle action selection (including exploration) and actuation, reward computation and so on. An SL inference function would need to provide automated data labeling.

Before an AI algorithm is deployed in a communication service provider's (CSP's) network, it is first designed and fully tested in a digital representation of the real network that is known as a network digital twin. The digital twin not only reproduces the deployment and configuration of the network and the radio propagation, traffic mix and user mobility of the environment condition, but also implements the same RAN learning architecture as shown in Figure 2. Following the same architecture enables both the training of performant AI models and the portability of the developed AI algorithm from the digital twin to the CSP's network with minimum overhead, thereby reducing the time to market.

Providing AI services in the RAN

The introduction of a RAN learning architecture that relies on a centralized RLE function makes it possible to efficiently provide the support services that are necessary to integrate AI algorithms in RAN systems.

We broadly distinguish AI services to reflect different phases of the lifecycle of an AI feature. Service type 0 refers to the initial release of an AI feature, which may be delivered in a RAN system as a software package upgrade. An incremental release in RAN nodes can allow safe integration of the new AI feature, which at this stage may be trained either in simulations or with data available to the vendor from proprietary testbeds or field trials, for example. In some cases, a CSP that is introducing an AI feature might want to train it with data from the CSP's own network. This would be supported by a process referred to as service type I. Finally, we refer to the continuous LCM of the AI feature while operational in a CSP network as service type II, which includes lifecycle managing the AI model, along with more advanced functionalities as algorithmic or design upgrades.

This simple and modular structure of AI services enables flexible customization on a per-CSP and per-AI-feature basis. A CSP could initially decide to introduce a new AI feature requiring only a standard service type 0. However, the service agreement may later be modified to include continuous AI services of type I or type II, with either full or partial service functionalities.

While there are numerous ways to deploy the learning architecture in RAN systems, only a few of them meet our criteria for a more AI-native RAN and enable the system vendor to retain responsibility for providing AI services during all phases of an AI feature lifecycle. The most efficient way to provide the continuous and flexible AI services of type I (feature training) and type II (feature LCM) is to maintain the learning functionalities as close as possible to the training data itself. To that end, the deployment of the RAN architecture must support two scenarios:

1. The provision of AI services from within the CSP network – that is, when the CSP data cannot be transferred to the vendor domain
2. The provision of AI services directly from the vendor domain – that is, when the CSP data is made available in the vendor domain.

We achieve such flexibility in design by deploying the two RLE function instantiations illustrated in *Figure 3*: a backend RLE in the vendor domain and a frontend RLE in the CSP network. In the example in Figure 3, the vendor and the CSP share the same cloud infrastructure, although this is not a prerequisite.

The two RLE instantiations are functionally almost identical, but with some crucial differences. The backend RLE is lifecycle managed by the vendor and retains the responsibility to control and configure the AI services for all AI features used in a CSP network. The frontend RLE is deployed and lifecycle managed by the CSP and it is responsible to release and administer the AI services in the CSP's own network, as it has direct access to the network functions hosting the actor/inference functions used by specific AI features.

The execution of type I (feature training) and type II (feature LCM) AI services is handled by the RLE function that has direct access to the CSP data. Therefore, when the CSP data is made available to the vendor domain (the light red area in Figure 3), the backend RLE also executes model training and LCM services within the vendor domain. Further, it instructs the frontend RLE to administer the services in the CSP network (AI model deployment and so on).

If the CSP data is not available in the vendor domain, the frontend RLE executes and administers the AI services within the CSP domain (the gray area in Figure 3) under the supervision and control of the backend RLE. Depending on which RLE instantiation provides continuous AI services of type I or type II, the "AI feature and service telemetry information" transferred from the frontend RLE to the backend RLE may include monitoring information of training and LCM services, AI models trained in the CSP domain, or training data generated in the CSP network.

Concept validation

In the pursuit of AI generalization for RAN features, we implemented the APEX architecture [8] for off-policy RL with actor-learner decoupling and considered DLLA as a use case to develop, validate and evaluate our design principles. We designed an AI algorithm that directly controls the modulation and coding scheme index for individual user transmissions to replace the legacy DLLA procedure. The introduction of a highly scalable learning architecture that capitalizes on distributed policy evaluations from multiple actors, each handling multiple parallel simulations, proved instrumental in boosting learning efficiency. Without altering the algorithm design itself, moving from a single actor to a 15-actor setup led to a twentyfold reduction in the training time and improved performance by 20 percent.

Most importantly, our approach made it possible to generalize our AI-driven DLLA solution over the RAN environment, intents and control parameters, which are more data-hungry to train than a single

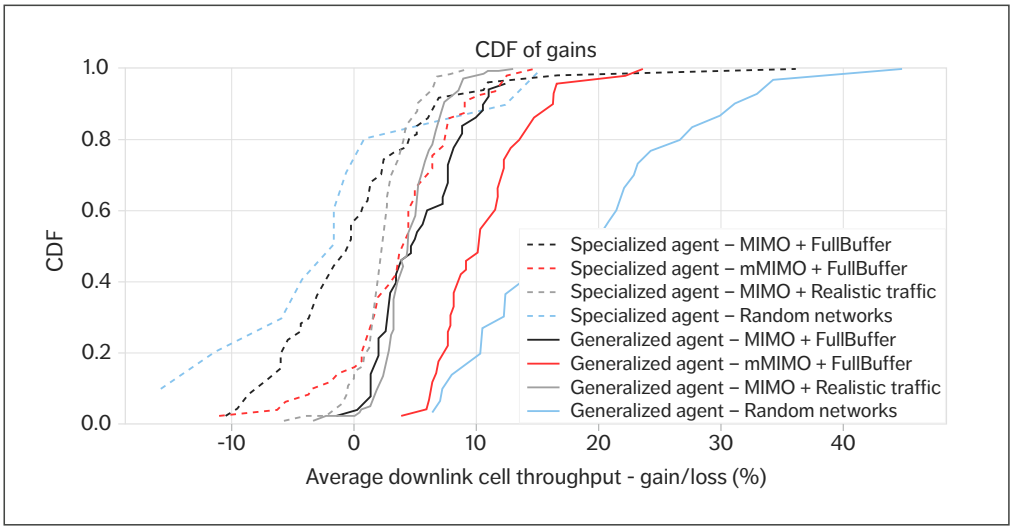


Figure 4 Cumulative distribution function (CDF) of average cell downlink throughput gain/loss over legacy DLLA evaluated for several realizations of the different benchmark scenarios

specialized AI model (but not necessarily compared to training thousands of specialized AI models).

To generalize DLLA over the RAN environment, we must capture characteristics of the RAN deployment surrounding UEs (including static and semi-static deployment and configuration information) and dynamic information characterizing the radio environment and UE state (including channel state information, HARQ (hybrid automatic repeat request) process and UE buffer).

Figure 4 shows that the generalized AI algorithm outperforms the legacy DLLA algorithm (which has a 10 percent BLER target) in relevant benchmark scenarios, comprising conditions, parameters and deployments unseen during training. Average cell throughput gains of well over 20 percent are observed in most benchmark scenarios, with smaller gains observed in scenarios where the legacy DLLA is expected to perform well. Figure 4 also shows that training an AI model to specialize in certain conditions leads to a less robust design that cannot cope with environmental changes.

Conclusion

Our research has identified two key enablers for efficiently integrating and systemizing artificial intelligence (AI) in future radio-access network (RAN) systems. Firstly, AI algorithms must be designed with the goal of generalization across the entire RAN environment, including intents and control tasks. Secondly, RAN systems must be empowered with an advanced and scalable learning architecture that can support a variety of AI algorithms and enable efficient learning at scale from thousands of distributed network entities. This architecture can be achieved with a centralized RAN learning engine that is able to support thousands of distributed actor/inference functions. While the deployment of actor/inference functions in RAN systems is use-case dependent, our analysis indicates that only certain deployments of the RAN learning engine lead to a lean integration of AI in RAN systems, resulting in the best AI performance and services.

Further reading

- **Ericsson Technology Review, AI-enabled RAN automation, available at:** *https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/ai-enabled-ran-automation*
- **Ericsson blog, Don't take AI and ML for granted - understand why they're critical for your RAN automation, available at:** *https://www.ericsson.com/en/blog/2022/4/dont-take-ai-and-ml-for-granted---understand-why-theyre-critical-for-your-ran-automation*
- **Ericsson, AI-powered radio-access networks, available at***: https://www.ericsson.com/en/ai/ran*
- **Ericsson, 5G RAN, available at:** *https://www.ericsson.com/en/ran*

References

1. **Ericsson Technology Review, Enhancing RAN performance with AI, January 20, 2020, Calabrese, F.D.; Frank, P; Ghadimi, E; Challita, U; Soldati, P, available at:** *https://www.ericsson.com/4ac66f/assets/local/reports-papers/ericsson-technology-review/docs/2020/enhancing-ran-performance-with-ai.pdf*
2. **IEEE Communications Surveys & Tutorials, Vol. 21, issue 4, Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues, June 2019, Sun, Y, et al., available at:** *https://ieeexplore.ieee.org/document/8743390*
3. **IEEE Transactions on Neural Networks 20 (1), 61-80, The Graph Neural Network Model, January 2009, Scarselli, F, et al., available at:** *https://ieeexplore.ieee.org/document/4700287*
4. **NIPS, Playing Atari with Deep Reinforcement Learning, 2013, Mnih, V, et al., available at:** *https://arxiv.org/pdf/1312.5602v1.pdf*
5. **ICLM, Massively Parallel Methods for Deep Reinforcement Learning, January 2015, Nair, A, et al., available at:** *https://arxiv.org/abs/1507.04296*
6. **ICML, Asynchronous Methods for Deep Reinforcement Learning, January 2016, Mnih, V, et al., available at:** *https://arxiv.org/abs/1602.01783*
7. **MLR, IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures, 2018, Espeholt, L, et al., available at:** *https://arxiv.org/abs/1802.01561*
8. **ICLR, Distributed Prioritized Experience Replay, September 2018, Horgan, D, et al., available at:** *https://arxiv.org/abs/1803.00933*
9. **ICLR, Recurrent Experience Replay in Distributed Reinforcement Learning, September 2018, Kapturowski, S, et al., available at:** *https://www.deepmind.com/publications/recurrent-experience-replay-in-distributed-reinforcement-learning*
10. **ICLR, SEED RL: Scalable and Efficient Deep-RL with Accelerated Central Inference, 2020, Espeholt, L, et al., available at:** *https://arxiv.org/abs/1910.06591*
11. **IEEE Communications Magazine 56 (9), 138-145, Learning radio resource management in RANs: Framework, opportunities, and challenges, 2018, Calabrese, F. D. et al., available at:** *https://ieeexplore.ieee.org/document/8466370*

THE AUTHORS

Pablo Soldati

◆ joined Ericsson in 2018 as a standardization and concepts researcher for radio networks and AI. In his current role, he defines solutions and strategy for the integration of AI in radio access networks. He holds a Ph.D. in telecommunications from KTH Royal Institute of Technology in Stockholm, Sweden.

Euhanna Ghadimi

◆ is a RAN data scientist working with AI algorithms and systemization within Business Area Networks. His research interests include AI, optimization theory and wireless networks. Ghadimi joined Ericsson in 2018. He holds a Ph.D. in telecommunications from KTH Royal Institute of Technology.

Burak Demirel

◆ joined Ericsson in 2020 and is currently a senior researcher whose work focuses on AI, RL, control theory and cyber-physical systems. He holds a Ph.D. in automatic control from KTH Royal Institute of Technology.

Yu Wang

◆ joined Ericsson in 2010 to drive research activities in RRM, network management and telecom data analytics. He is currently a concept developer, focusing on the development of AI/ML-based radio network automation solutions. Wang holds a Ph.D. in communication engineering from Aalborg University, Denmark.

Mathias Sintorn

◆ is an expert in traffic handling and service performance within Business Area Networks. He joined Ericsson in 1998. In his current role, he defines the long-term evolution of the RAN architecture, specifically in the area of RAN automation. Sintorn holds an M.Sc. in engineering physics from Uppsala University, Sweden.

Raimundas Gaigalas

◆ joined Ericsson in 2005 and has been working with concept and simulator development in RRM Layer 2 and the systemization of commercial features in various RAN products. He currently serves as a developer in Cloud RAN with a focus on AI/ML functions in the distributed unit. Gaigalas holds a Ph.D. in mathematical statistics from Uppsala University.

ISSN 0014-0171
284 23-3390 | Uen

Ericsson
SE-164 83 Stockholm, Sweden
Phone: +46 10 719 0000