Project Report

CELTIC-NEXT AIMM Project

WP5: AI for Network Operation and Management

D5.2

Authors: Rui Inacio, Abdelrahim Ahmad (Vilicom), Keith Briggs, Shipra Kapoor, Henry Brice (BT), Arman Shojaeifard (InterDigital), Robert Piechocki, Hakan Erdol, Peizheng Li (UoB), Cliff Element (ThinkRF)

Project Acronym:	**AIMM**
Project Full Title:	**AI-enabled Massive MIMO**
Project Coordinator:	**Arman Shojaeifard (InterDigital)**
Project Duration:	**24 months (Oct. 2020 – Sep. 2022)**
Submission Date:	**30 September 2022 (M24)**
Dissemination Level:	**External**

Abstract

This report describes the progress of WP5 works in the AIMM project. The WP5 works are organised around three main activities:

- specify entire network operation and management framework.
- develop interactive and quarriable simulation environment for network functions testing
- develop hierarchical and multiagent Reinforcement Learning strategies

The work done have covered all these three tasks. Partners have worked in cooperation to develop Network Simulators to support the development of xApps/rApps, to implement a project dedicated Network Analytics Platform aimed at providing live network data and a testbed to the project partners, and finally on the development of xApps/rApps addressing the specified project use cases.

The document focus is on the work done across the whole project with emphasis on presenting results from the technology developed in WP5.

.

Executive Summary

Software-defined networking (SDN), network function virtualization (NFV), massively distributed computing, heterogeneous and cell-less system architectures are very likely to feature prominently in next-generation digital infrastructures. However, these networks of the future and their newly introduced features, significantly increase the complexity of a system. The network operation and management are becoming a very challenging task, having the traditional human-centric network infrastructure management model is no longer a viable option. It is a well appreciated strategy for newly introduced features, algorithms, policies, to be tested offline, before deployment on a target system. Different bottlenecks and drawbacks can be identified and hyperparameters tuned in a System Development Life Cycle (SDLC)-like model. Furthermore, the performance can be thoroughly evaluated before being deployed in a real system. A common practice is to do so by means of simulation tools. Recently a concept of “Digital Twin” has been proposed by various communities working on cyber physical systems. The AIMM project will develop the concept of “Digital Network Oracle” – which can be viewed as an augmented “Digital Twin” – the added capability refers to an architecture, which allows for “the twin” to be queried on instantaneous basis.

Task 5.1 covers the network management and operation function and frameworks that currently depend heavily on traditional data to helps engineers and computational models to understand how the network behaves and determine the possible causes for such behaviour. Current network management frameworks and techniques are being researched and project is defining the best approach for AI based automation within the ORAN Alliance architecture, with particular focus: to identify the requirements for data sources and its structures; topologies and interfaces; processes, policies and KPIs; and address the use cases defined in WP2.

Task 5.2 focus on the research and development of system level simulator frameworks that support the development and testing of AI algorithms. These simulators aim to mimic network functions, behaviours and network metric collection and reporting, thus offering the interfaces for the implementation of upper software layers that address the automation of network management procedures, policy enforcement, etc.

Task 5.3 is dedicated to the research and development of new Reinforcement Learning techniques to be applied on the development of xApps and rApps focused on automating network operations and creating capabilities for the network be autonomic in managing/orchestrating its resources to optimise its performance.

This document reports on the work developed on these three tasks across the entire project duration.

Table of Contents

List of Figures and/or List of Tables

Abbreviations

Abbreviation	Definition
3G	Third generation cellular
3GPP	Third Generation Project Partnership
4G LTE/LTE-A	Fourth generation cellular Long-Term Evolution/Long-Term Evolution Advanced
5G NR	Fifth generation cellular New Radio
A1	O-RAN interface between Non-RT RIC and Near-RT RIC
AAS	Active antenna system
AI/ML	Artificial Intelligence/Machine Learning
BS	Base station
CAPEX	Capital expenditure
CoMP	Coordinated Multipoint
CPRI	Common Public Radio Interface
CQI	Channel Quality Indicator
CSI	Channel State Information
CS-RS	Cell-Specific Reference Signal
CU	Centralised unit
DCI	Downlink Control Indicator
DM-RS	Demodulation Reference Signal
DPB	Dynamic Point Blanking
DPC	Dirty-paper-coding
DPS	Dynamic Point Selection
DU	Distributed unit
E2	O-RAN interface between Near-RT RIC and CUs/DUs
eCPRI	Enhanced Common Public Radio Interface
EM	Electromagnetic
eNB	eNodeB (4G LTE/LTE-A base station)
EVM	Error vector magnitude
F1	3GPP interface between CU and DU
FD MIMO	3D/full-dimension MIMO
FR1	Frequency range 1
FR2	Frequency range 2
gNB	gNodeB (5G NR base station)
HLS	Higher-layer-split
IPR	Intellectual property rights
IRS	Intelligent Reflecting Surface
KPI	Key-performance-indicator
L#	Layer number # on the protocol stack
LLS	Lower-layer-split
LOS	Line-of-sight
MAC	Medium Access Control
MDT	Minimisation of drive test
MIMO	Multiple-input multiple-output
MRT	Maximum-ratio-transmission
M-TRP	Multi transmission/reception points
Near-RT	Near-real-time
Non-RT	Non-real-time
OPEX	Operational expenditure
O-RAN	O-RAN Alliance
Open RAN	Ecosystem for open standardised interfaces implementation
PA	Power amplifier
PBCH	Physical Broadcast Channel
PDCP	Packet Data Convergence Protocol
PDSCH	Physical Downlink Shared Channel
PHY	Physical Layer
PLMN	Public Land Mobile Network
PRACH	Physical Random Access Procedure
PSS	Primary Synchronisation Signal

PUSCH	Physical Uplink Shared Channel
QoE	Quality-of-experience
QoS	Quality-of-service
RAN	Radio access network
rApp	An application designed to run on the Non-RT RIC
RAT	Radio Access Technology
RIC	O-RAN RAN Intelligent Controller
RIT	Radio Interface Technology
RLC	Radio Link Control
RRC	Radio Resource Control
RSRP	Reference Signal Received Power
RSRQ	Reference Signal Received Quality
RT	Real-time
RU	Radio unit
SINR	Signal-to-interference-plus-noise ratio
SISO	Single-input single-output
SON	Self-organising-network
SRIT	Set of Radio Interface Technologies
SSB	System synchronisation block
SSS	Secondary Synchronisation Signal
TXRU	Transceiver chain
UE	User equipment
vRAN	Virtualised RAN
X2	3GPP interface between eNBs
xApp	An application designed to run on the Near-RT RIC
Xn	3GPP interface between gNBs
ZF	Zero-forcing

1 Introduction

This report describes the research and development work, undertaken in WP5 throughout the AIMM's duration, to specify and develop a Network Operation and Management Framework which is automated using AI.

The approach taken in this workpackage is to work in three parallel workstreams:

- Research existing Network Management Frameworks, identify the main requirements, features, and procedures, in order to specify the high-level characteristics of a framework which are suitable for the development of AI-based automation within the project.
- Research system-level simulation frameworks and develop platforms to support the development and testing of AI/ML algorithms for O-RAN-compliant RIC xApps and rApps.
- Research and develop the AI algorithms which implement O-RAN-compliant RIC xApps and rApps, aimed at resolving AIMM project use cases defined in WP2.

The following sections of this document will provide detail of the work produced in the project.

2 High-level Network Operation

5G is an umbrella concept under which multiple technologies and concepts come together in different domains, to implement a seamless integration of mobile and fixed network to allow the interconnection of people, things, and information flows. Network Function Virtualization (NFV) and Software Defined Networking (SDN) are two key enabler technologies of 5G, mainly underpinning the programmability of network functions aimed at the automation of provisioning, network management, operation, and maintenance tasks. The principles of programmability and automation are crucial for the implementation of the most important improvements promised by 5G, mainly at a network performance level such as increased spectrum efficiency or reduced latency. These principles were also very important for the AIMM project, being the foundations for our work on developing Artificial Intelligence (AI) algorithms and data platforms that enable the implementation of advanced Cognitive Network Management use cases.

At an early stage the AIMM project team reviewed the existing technology trends, operational and delivery frameworks to deliver 5G and decided to adopt within the project scope of work the technology concept proposed by the O-RAN Alliance. This decision has a significant impact across all the work-packages and for the case of WP5 the impact was fundamental in shaping posterior work.

2.1 Network Operation and Management Frameworks

The early research work done by Vilicom was to identify the existing network operation and management frameworks, critically evaluate their characteristics against AIMM's objectives and requirements and select one to use in the project.

2.1.1 TMN FCAPS

The objectives, requirements, and concepts of a Telecommunications Management Network (TMN) framework were introduced in 1988 (latest revision dates from 2000) by ITU-T in [2.1], which specifies the general architectural requirements for the management plane, of Public Telecommunications Operators (PTOs), responsible for the provisioning, installation, operation and administration activities of telecommunication networks and services. This document defines a functional architecture, information architecture, physical architecture, interfaces, and relationships between these architectures for the TMN management plane.

Among the most important requirements of the TMN are the following:

- The ability to securely collect and exchange management information between the network element (NE) layer, also referred to telecommunications functional layer, and the TMN layer thus enabling every function defined for network and service management.
- The format of this management information should be convertible to achieve consistency across all TMN layers and functions.
- Ability to analyse the data and consequently perform corrective and management actions.
- All the functions, architectures and mechanisms must be technology agnostic and should evolve in time to adapt to future requirements.

The TMN framework is organised into five Management Functional Areas (MFAs):

- Fault Management
- Configuration Management
- Accounting Management
- Performance Management
- Security Management

These MFAs are described in detail in ITU-T M.3400 [2.2].

2.1.1.1 Fault Management

Fault Management is a set of functions which enables the detection, isolation, and correction of abnormal operation of the telecommunication network, the component NEs and its environment. These function-set-groups are listed in Figure 1**Error! Reference source not found.** and are defined to provide mechanisms and measurements for the execution of the maintenance phases defined in ITU-T M.20 [2.3]. These maintenance phases are the execution of Performance Measurements, Failure Detection, System Restoration, Failure or performance information, Fault Localisation, Logistic Delay, Fault Correction, Verification and Restoration.

Most of the originally proposed functions can be improved using AI and other automation mechanisms. New functions could be introduced using AI to mine and learn from the huge amounts of data produced by the network and its users.

Fault Management

Original proposed function set groups

- RAS Quality Assurance
- Alarm Surveillance
- Fault Localisation
- Fault Correction
- Fault Prevention
- Testing
- Incident Management

Newly proposed functions with the introduction of AI

- Fault Prediction

Figure 1 - Fault Management function set groups

2.1.1.2 Configuration Management

Configuration Management functions focus on the exercise of control over design and configuration aspects of the network, on the identification of NEs through logical topologies, on the collection of data from the NEs and on the provisioning of new/modified configurations. The function set group defined in this model is listed in Figure 2.

Configuration Management

Original proposed function set groups

- Network Planning and Engineering
- Installation
- Service Planning and Negotiation
- Provisioning
- Status and control

Newly proposed functions with the introduction of AI

- Automatic creation of QoS and QoE differentiation rules
- Automated Provisioning
- Automatic creation of new policies and rules
- Automatic enforcement of policies

Figure 2 - Configuration Management function set groups

2.1.1.3 Accounting Management

Accounting Management function set groups, described in Figure 3, enables the measurement of network services usage, the determination of costs to the service provider and charges to the customer for such use. It also supports the determination of prices for services. This information is heavily reliant on information collected from the many NEs and functions deployed across the network through the O&M functionality offered by Operational Support Systems (OSS) and Element Management Systems (EMS).

Accounting Management

Original proposed function set groups

- Usage Measurement (incl. billing and fraud prevention)
- Tariffs and Prices
- Collections and Finance
- Enterprise Control

Figure 3 - Accounting Management function set groups

2.1.1.4 Performance Management

The function set groups defined for the Performance Management (PM) capability provides mechanism to collect/store Quality of Service (QoS) measurement data from telecommunication equipment, component deployed across the network, that is used to monitor, evaluate, report, and describe the network behaviour through the analysis of (key) performance statistical indicators. The purpose of these processes as defined in [2.3] is to maintain service levels by identifying and correcting faults/degradations in performance and for continuous performance improvement though the optimisation of configuration and NE design. Figure 4 lists the proposed functions by the PM set group.

Performance Management

Original proposed function set groups

- Performance Quality Assurance.
- Performance Monitoring.
- Performance Control.
- Performance Analysis.

Newly proposed functions with the introduction of AI

- Prediction of Performance degradations
- Forecasting of Performance based on configuration scenarios
- Automatic virtualised resource orchestration

Figure 4 - Performance Management function set groups

2.1.1.5 Security Management

Security Management provides for the management of security across the network infrastructure which also includes mechanisms that focus on the security of all network management functional areas as specified in [2.1]. The functionalities implemented by network management plane information, procedures and tools are crucial for the assurance of QoS across the network and it is fundamental to protect its interfaces, information flows and rights of access from malicious or antagonistic users. The function set groups defined by Security Management are listed in Figure 5.

Security of Management functionality includes Security services for communications and Security event detection and reporting.

The Security services cover all the aspects related to authentication, access control, data confidentiality, data integrity, and non-repudiation that may be exercised during any communications between systems, between customers and systems, and between internal users and systems.

Security event detection and reporting implements the functions responsible for the surveillance of security breaching events and subsequent reporting to higher layers of security any activity that may be construed as a security violation (e.g. unauthorized user, physical tampering with equipment, tampering with on-line configuration of the network elements, etc.)

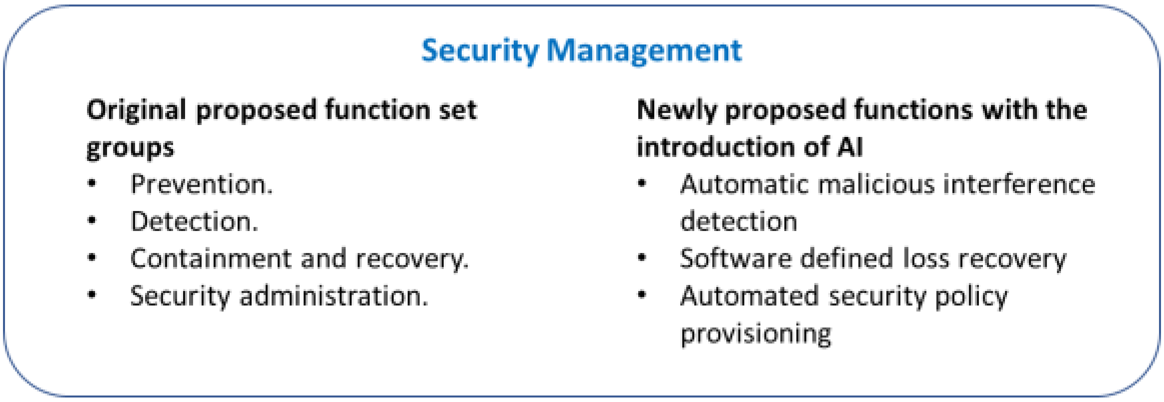


Figure 5 - Security Management function set groups

2.1.2 TMForum eTOM

The Enhanced Telecom Operations Map (eTOM), also referred by Business Process Framework, has been specified and maintained by TeleManagement Forum (TMForum) as a framework to categorise all the business activities of a telecommunications service provider.

The focus of eTOM framework is to enable end-to-end process automation of the business and operations processes that deliver information and communications services. It defines the business processes used by service providers, its interfaces, the links that connect these and the implemented use cases for the customer of each process. This framework also defines what services, resources and other types of information should be consumed by these business processes. It assumes, the automatic exploitation of the information generated across the organisation, to be a key factor for the future success of the telecommunication enterprise.

The eTOM business process framework represents the whole of a service provider's enterprise environment. At a conceptual level it can be depicted as having three major process areas, as shown in Figure 6.

- Strategy, infrastructure, and product – Covering planning and lifecycle management (associated with development and delivery).
- Operations – Covering the core of operational management.
- Enterprise management – Covering corporate or business support management.

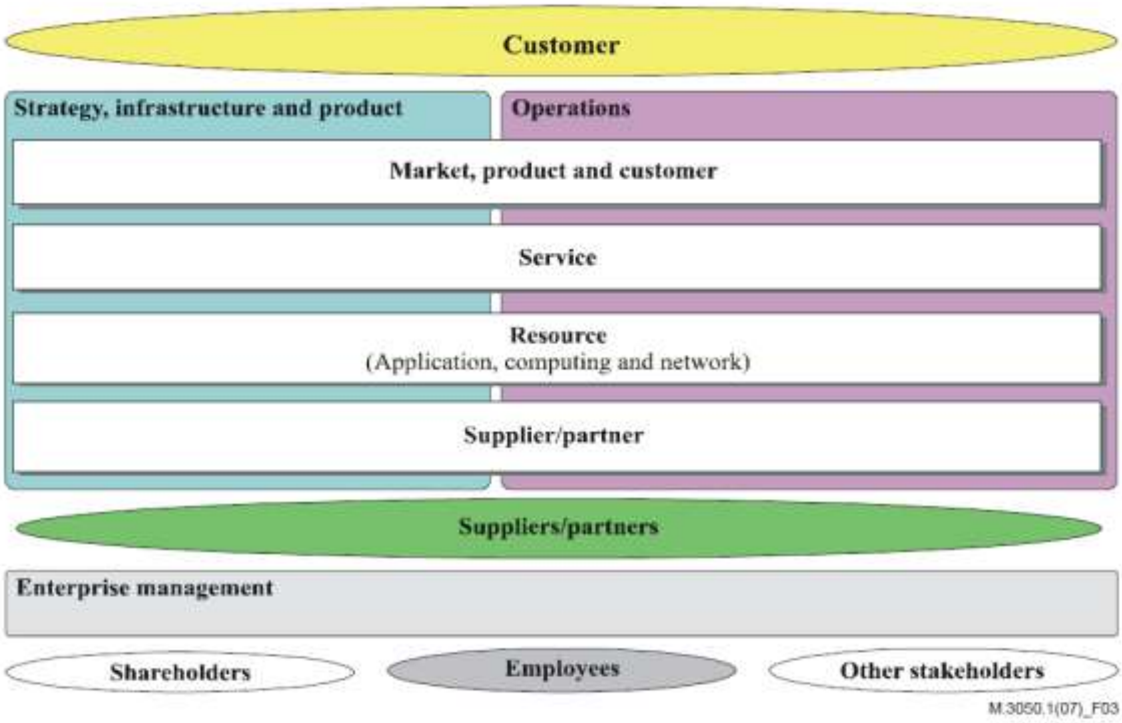


Figure 6 - eTOM business process framework conceptual structure [2.4]

The conceptual structure view provides an overall context that differentiates strategy and lifecycle processes from operations processes. It also identifies the key functional process structures in four horizontal layers across these two main process areas. Additionally, it depicts the internal and external entities that interact with the enterprise, such as customers, suppliers, partners, and overall stakeholders.

The operations process area describes the processes and activities that are the traditional core competencies of the Service Provider (SP) enterprise, which includes network operations, network management and service operations which deals with customers' support. As it can be seen from Figure 7**Error! Reference source not found.** the operations area also includes the processes of sales management and supplier/partner management.

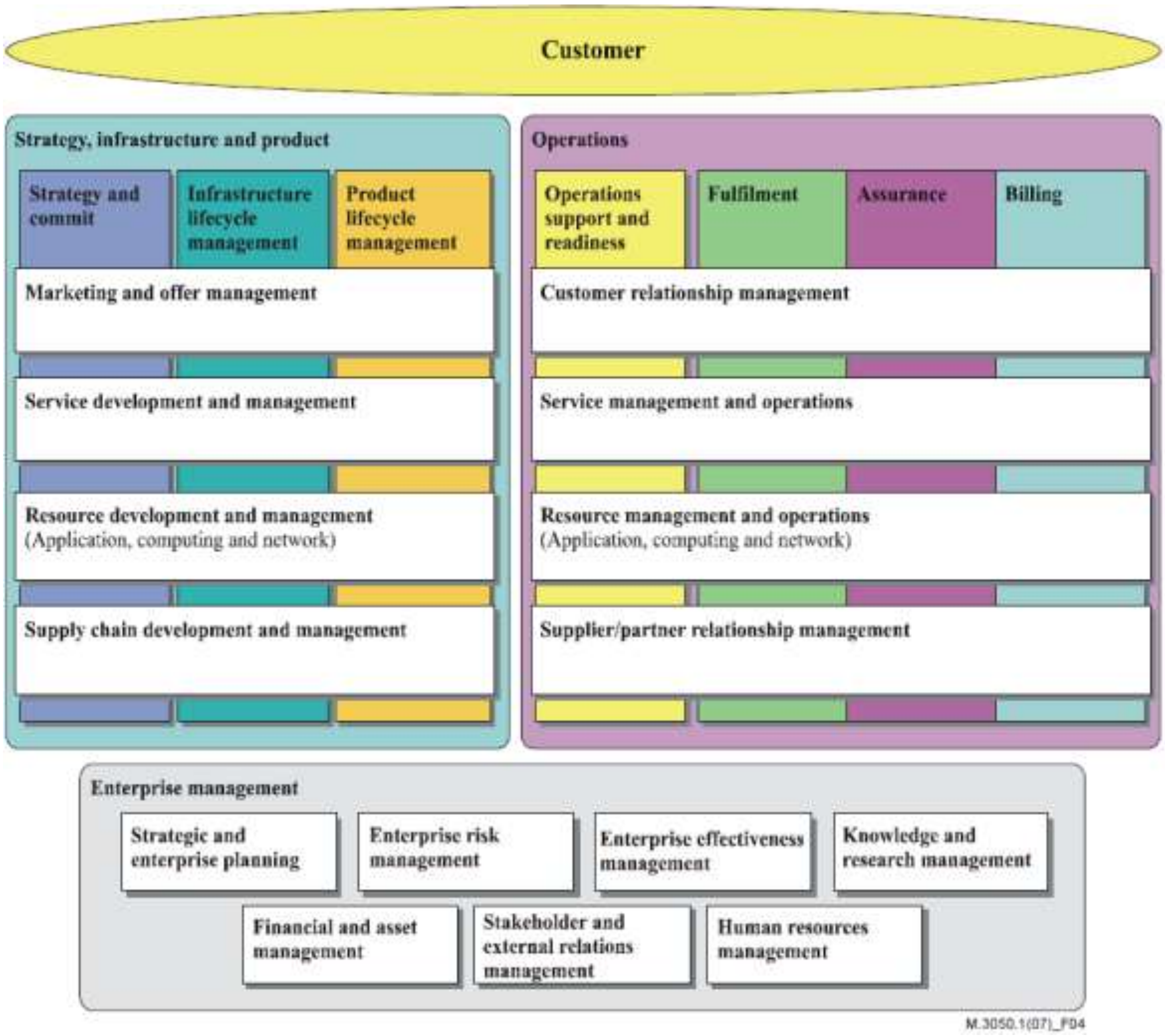


Figure 7 - eTOM level 0 view of level 1 process groupings [2.4]

On the other hand, the strategy, infrastructure, and product process area include processes focused on the definition and execution of strategies focused on the development of the network throughout its life cycle. Within the remit of this processes area is the planning and development of new technological features, product capability evolution and organisational competences crucial for the management of technology and product life cycle across networks, service, and products.

In the eTOM framework, infrastructure refers to more than just the resource (IT and network) infrastructure that directly supports products and services. It also includes the operational and organizational infrastructure required to support marketing, sales, service, and supply chain processes, e.g., customer relationship management (CRM). These processes direct and enable processes within the operations process area.

The enterprise management process area covers all the basic business processes that are required to manage any corporate business. These comprehend most of the complementary business competences and capabilities that support the core business functions on the implementation of business, technological and organisational strategies.

Each process area is further organised in to supporting functional process structures reflecting the core business and organisational competences of the mobile network operator enterprise. These functional processes are:

- Market, product and customer processes dealing with sales/channel and marketing management, product development and offer management, and finally operational processes that oversee network management, customer support, problem handling, SLA management, billing, etc.
- Service processes deal with service development, service delivery and service management competencies and capabilities.

- Resource processes focuses on the development and delivery of network and IT infrastructure as a key resource for the delivery of products and services. It is involved in technology management including aspects of architectural, design and functional evolution and operational management aspects such as provisioning, incident management and performance management.

Supplier and partner processes deal with the development and management of supply chains that underpins aspects related with infrastructure, product development and service delivery.

2.1.3 ETSI NFV MANO

The Network Functions Virtualisation Management and Orchestration (NFV-MANO) architectural framework was proposed by the European Telecommunications Standards Institute (ETSI) to manage the Network Functions Virtualisation Infrastructure (NFVI) and orchestrate the allocation of resources needed by the Network Services (NSs) and Virtualised Network Functions (VNFs). Figure 8 presents the high-level framework’s architecture.

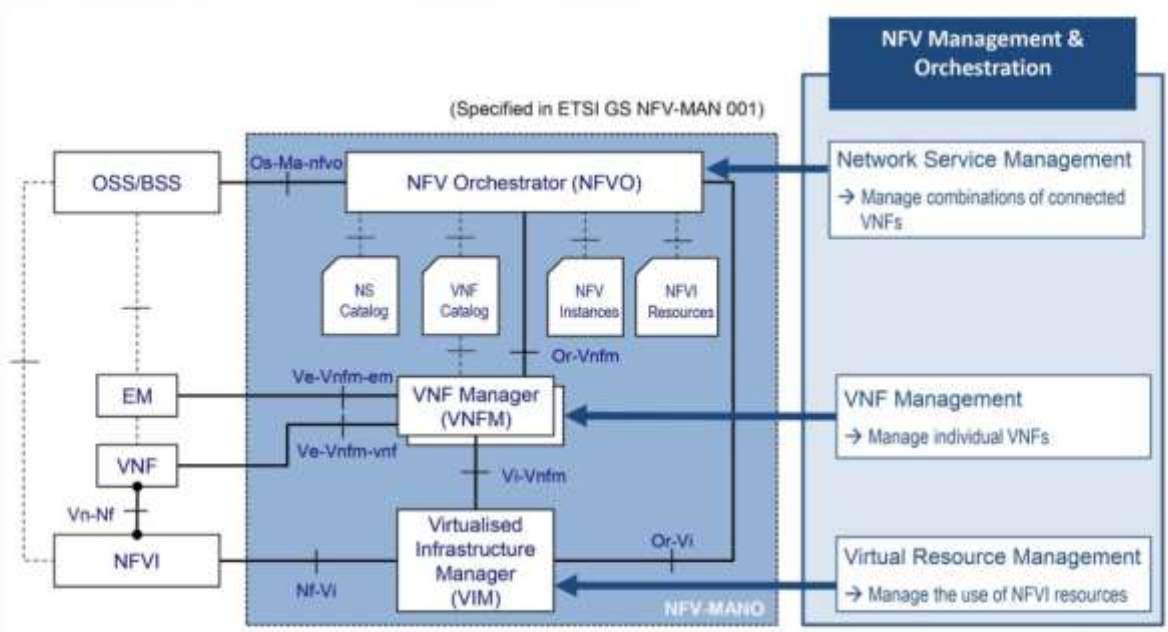


Figure 8 - NFV-MANO architectural framework [2.5]

NFV-MANO Functional Blocks	
VIM	Virtualised Infrastructure Manager • responsible for controlling and managing the NFVI compute, storage and network resources, usually within one operator's Infrastructure Domain (e.g. all resources within an NFVI-Point-of-Presence (PoP), resources across multiple NFVI-POPs, or a subset of resources within an NFVI-PoP); • it may be specialized in handling a certain type of NFVI resource (e.g. compute-only, storage-only, networking-only), or may be capable of managing multiple types of NFVI resources (e.g. in NFVI-Nodes).
NFVO	Network Function Virtualisation Orchestrator • the orchestration of NFVI resources across multiple VIMs, fulfilling the Resource Orchestration functions; • the lifecycle management of Network Services, fulfilling the Network Service Orchestration functions.
VNFM	Virtualised Network Function Manager • responsible for the lifecycle management of VNF instances; • may be responsible from one or more VNF instances; • VNFM functions are generic common functions applicable to any kind of VNF;

	• NFV-MANO also supports the case where VNF instances require a specific set of functionalities for the management of its lifecycle, which must be specified in the VNF package.
NFV-MANO Data Repositories	
NS-C	Network Services Catalogue • represents the repository of all of the on-boarded Network Services, supporting the creation and management of the NS deployment templates (Network Service Descriptor (NSD), Virtual Link Descriptor (VLD), and VNF Forwarding Graph Descriptor (VNFFGD) via interface operations exposed by the NFVO.
VNF-C	Virtualised Network Function Catalogue • represents the repository of all the on-boarded VNF Packages, supporting the creation and management of the VNF Package (VNF Descriptor (VNFD), software images, manifest files, etc.) via interface operations exposed by the NFVO.
NFV IR	Network Function Virtualisation Instances Repository • holds information of all VNF instances and Network Service instances; • Each VNF instance is represented by a VNF record, and each NS instance is represented by an NS record; • Those records are updated during the lifecycle of the respective instances, reflecting changes resulting from execution of NS lifecycle management operations and/or VNF lifecycle management operations.
NFVI RR	Network Function Virtualisation Infrastructure Resources Repository • holds information about available/reserved/allocated NFVI resources as abstracted by the VIM across operator's Infrastructure Domains, thus supporting information useful for resources reservation, allocation and monitoring purposes.
Other Functional Blocks that share reference points with NFV-MANO	
EM	Element Management • responsible for FCAPS management functionality for a VNF.
VNF	Virtualised Network Function • responsible for the virtualisation of specific network functions.
OSS/BSS	Operational Support System and Business Support System • combination of the operator's other operations and business support functions that are not otherwise explicitly captured in the NFV-MANO framework.
NFVI	Network Function Virtualisation Infrastructure • encompasses all the hardware (e.g. compute, storage, and networking) and software (e.g. hypervisors) components that together provide the infrastructure resources where VNFs are deployed; • may also include partially virtualised NFs.
Os-Ma-nfvo	Reference point between OSS/BSS and NFV-MANO functional blocks.
Ve-Vnfm-em	Reference point between EM and VNFM
Ve-Vnfm-vnf	Reference point between VNF and VNFM
Nf-Vi	Reference point between NFVI and VIM
Or-Vnfm	Reference point between NFVO and VNFM
Or-Vi	Reference point between NFVO and VIM
Vi-Vnfm	Reference point between VIM and VNFM

Network Functions Virtualisation (NFV) adds new capabilities to communications networks and requires a new set of management and orchestration functions to be added to the current model of operations, administration, maintenance, and provisioning.

In legacy networks, Network Function (NF) implementations are often tightly coupled with the infrastructure they run on. NFV decouples software implementations of Network Functions from the computation, storage, and networking resources they use. The virtualisation insulates the Network Functions from those resources through a virtualisation layer.

From this decoupling emerge a new set of entities, the Virtualised Network Functions (VNFs), and a new set of relationships between them and the NFV Infrastructure (NFVI). VNFs can be chained with other VNFs and/or Physical Network Functions (PNFs) to implement a Network Service (NS).

Since NS, PNFs, VNFs, NFVI and the relationships between them did not exist before the emergence of NFV, their handling requires a new and different set of management and orchestration functions that is introduced by the NFV-MANO framework.

The NFVI resources considered, by this framework, are both virtualised and non-virtualised resources, supporting fully virtualised network functions and partially virtualised network functions.

Virtualised resources in-scope are those associated with virtualisation containers, and are offered for consumption through appropriate abstract services, for example:

- Compute nodes including machines (e.g. hosts or bare metal), and virtual machines, as resources that comprise both CPU and memory.
- Storage, including volumes of storage at either block or file-system level.
- Network, including networks, subnets, ports, addresses, links and forwarding rules, for the purpose of ensuring intra- and inter-VNF connectivity.

The management and orchestration of virtualised resources should handle NFVI resources (e.g. in NFVI Nodes) in NFVI Points of Presence (NFVI-PoPs). Management of non-virtualised resources is restricted to provisioning connectivity to PNFs, necessary when a NS instance includes a PNF that needs to connect to a VNF, or when the NS instance is distributed across multiple NFVI-PoPs or N-PoPs.

The virtualised resources are managed to provide to VNFs, at each instant, with the resources they need. Allocation and release of resources is a dynamic process, in response to consumption of those services by multiple VNFs and should provision for specific and differentiated network requirements such as bandwidth and latency as an example.

While the management and orchestrations function for virtualised infrastructure are VNF-unaware, resource allocations and releases may be needed throughout the VNF lifetime. An advantage of NFV is that with increasing load VNFs can dynamically consume services that allocate additional resource when scaling-out is triggered.

Management and orchestration aspects of specific VNFs include traditional Fault Management, Configuration Management, Accounting Management, Performance Management, and Security Management (FCAPS) model. The decoupling of Network Functions from the physical infrastructure underpinning it, requires a new set of management functions focused on the creation and lifecycle management of the virtualised resources allocated for the VNF, processed that is referred to as VNF Management.

VNF Management functions are responsible for the VNF's lifecycle management including operations such as:

- Instantiate VNF (create a VNF using the VNF on-boarding artefacts).
- Scale VNF (increase or reduce the capacity of the VNF).
- Update and/or Upgrade VNF (support VNF software and/or configuration changes of various complexity).
- Terminate VNF (release VNF-associated NFVI resources and return it to NFVI resource pool).

During the VNF on-boarding process the VNF Management function reads from a deployment template the requirements necessary to realize such VNF and captures, in an abstracted manner, the requirements to manage its lifecycle.

During the lifecycle of a VNF, the VNF Management functions may monitor KPIs of a VNF, if such KPIs were captured in the deployment template. The management functions may use this information for scaling operations. Scaling may include changing the configuration of the virtualised resources (scale up, e.g., add CPU, or scale down, e.g., remove CPU), adding new virtualised resources (scale out, e.g., add a new VM), shutting down and removing VM instances (scale in), or releasing some virtualised resources (scale down).

The VNF Management services each VNF, without interfering with the logical functions performed by the VNF, by exposing its functions in an open, well known abstracted manner other functions.

The services provided by VNF Management can be consumed by authenticated and properly authorized NFV management and orchestration functions.

2.1.4 ORAN RIC

The O-RAN OAM Architecture, shown in Figure 9, identifies management services, managed functions and managed elements supported in O-RAN, including the interworking between service management and orchestration and other O-RAN components. The requirements, for this architecture, are derived from end-to-end OAM including the initial provisioning of O-RAN service across VNFs and PNFs, and data collection from O-RAN Managed Elements and O-Cloud. The architecture identifies the interfaces between O-RAN Service Management and Orchestration and Managed Elements for different models and example deployment options.

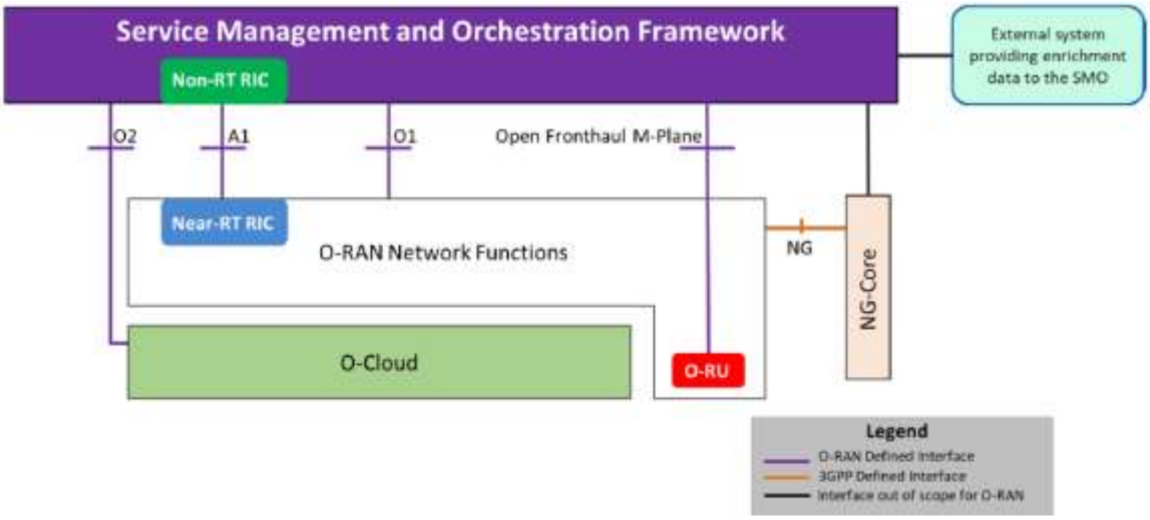


Figure 9 - High-level ORAN Architecture [2.6]

network management functionalities for the RAN and may also be extended to perform Core Management, Transport Management, and end-to-end Slice Management. This function is responsible to provide: the FCAPS interfaces and procedures to be used by the multiple O-RAN VNFs; the Non-RT RIC for RAN optimisation; and O-Cloud Management, Orchestration and Workflow Management.

SMO interfaces	
A1	Interface between the Non-RT RIC in the SMO and the Near-RT RIC for RAN Optimization
O1	Interface between the SMO and the O-RAN Network Functions for FCAPS support
M-plane	Open Fronthaul M-plane interface between SMO and O-RU for FCAPS support
O2	Interface between the SMO and the O-Cloud to provide platform resources and workload management

Service Management and Orchestration (SMO)

SMO offers the FCAPS functionality through the O1 interface. The ORAN standards, adapted the original FCAPS function set and created a new list of processes to be executed by O1 interface:

- Fault Management (FM)
- Configuration Management (CM)
- Performance Management (PM)
- File Management
- Communications Surveillance (Heartbeat)
- Trace
- Physical Network Function (PNF) Discovery

- PNF Software Management

The Open Fronthaul M-plane interface also implements FCAPS procedures to the O-RU. The procedures include:

- “Start-up” installation
- SW Management
- Fault Management (FM)
- Configuration Management (CM)
- Performance Management (PM)
- File Management

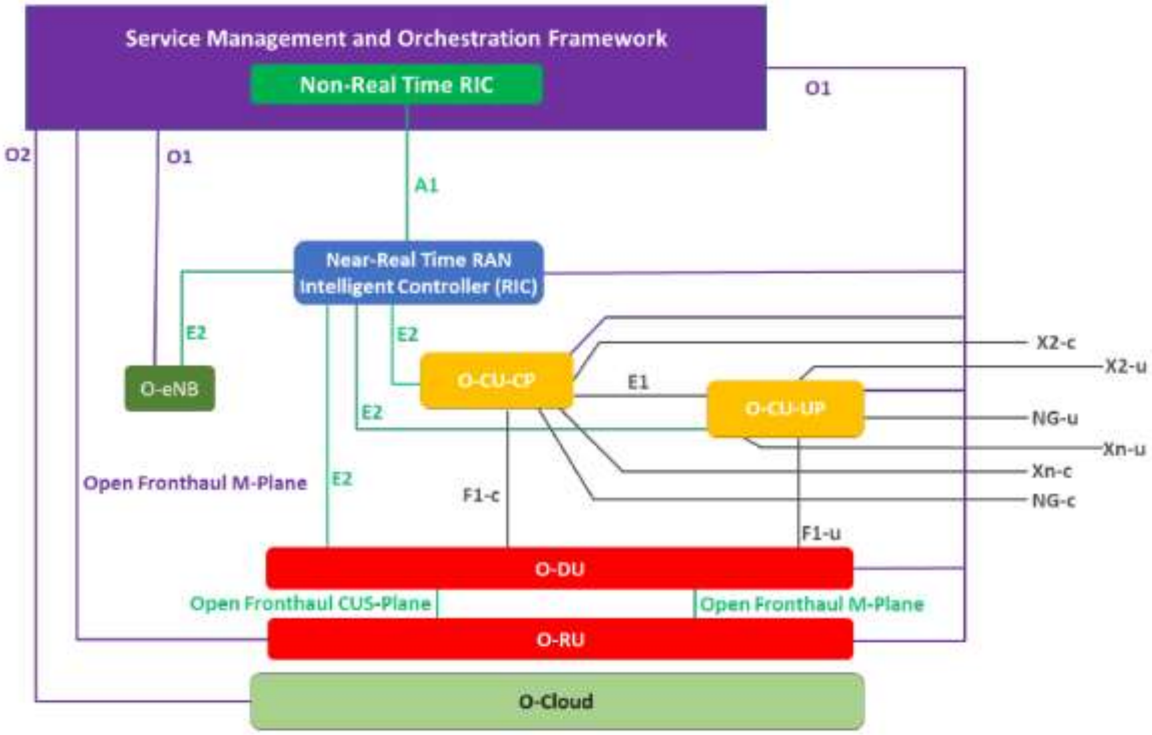


Figure 10 - Logical Architecture of ORAN [2.6]

Non-RT RIC

Another important functionality offered by the SMO is the Non-RT RIC that implements the A1 interface towards the Near-RT RIC.

Non-RT RIC is a functional platform designed to implement automated policy-based RAN optimisation activities, by running and managing Machine Learning (ML) models and data enrichment that facilitates analytics, decision making and learning by AI/ML algorithms.

Near-RT RIC

The O-RAN Near-RT RIC, as shown in **Error! Reference source not found.**, is a logical function that enables near real-time control and optimization of E2 nodes functions: O-eNB; or O-CU and O-DU. This interface implements controls loops for the gathering of data and the executions of optimisation actions via xApps. The frequency of execution of such control loops may vary in order of 10ms to 1s. All the actions taken by the Near-RT RIC are guided by the policies and enriched data coming from the Non-RT RIC.

Network Functions

ORAN architecture follows 3GPP architecture and interface specifications to the possible extent [2.6]. The main network functions implemented in ORAN are related with the 5G-NR gNB, and consist of one Open-RAN Centralised Unit (O-CU) that connects to and controls at least one Open-RAN Distributed Unit (O-DU) via the F1 interface. The O-DU connects to at least an Open RAN Radio Unit (O-RU), via an Open Fronthaul CUS-Plane, to implement the Uu interface. The Open Fronthaul M-plane exist for the implementation of the Management Plane towards the O-RU via the O-DU.

O-CU

The O-CU terminates both the User Plane (UP) and Control Plane (CP) of the NG interface towards 5G Core Network. It also terminates the Xn interfaces between gNBs. O-RAN NR also supports multi-RAT Dual Connectivity and for this reason the O-CU also supports the S1 interface towards EPC and X2 towards eNBs.

Depending on the implement bearer split option (Figure 11) the O-CU will terminate protocol layers.

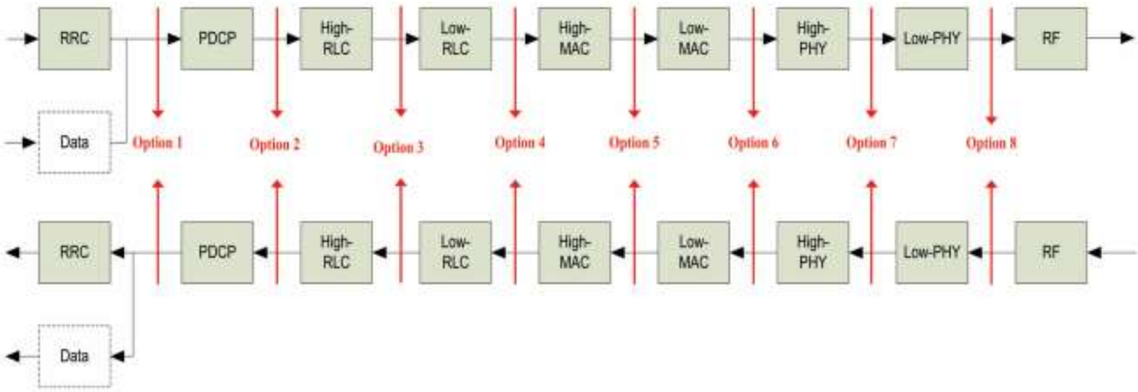


Figure 11 - 3GPP Function Split between central and distributed unit [2.7]

O-CU-CP

The O-CU-CP is the logical component of the O-CU that deals with all Control Plane related protocol stack, signalling messages and interfaces:

- The O-CU-CP is connected to each O-DU through F1-C interface;
- The O-CU-CP is connected to each O-CU-UP through E1 interface;
- One O-DU is connected to only one O-CU-CP;
- The O-CU-CP terminates the E2 interface to Near-RT RIC;
- The O-CU-CP terminates O1 interface towards the SMO;
- The O-CU-CP terminates NG-c interface to 5G Core (5GC);
- The O-CU-CP terminates X2-c interface to eNB or to en-gNB in EUTRAN-NR Dual Connectivity (EN-DC);
- The O-CU-CP terminates Xn-c to gNB or ng-eNB;

This function is responsible for all the mobility related procedures. According to O-RAN specifications this function is responsible for the termination of the PDCP and RRC protocols of the CP to implement the Signalling Radio Bearers (SRBs) towards the User Equipment (UE).

O-CU-UP

The O-CU-UP is the logical component of the O-CU that deals with all User Plane related protocol stack, traffic and interfaces:

- The O-CU-UP is connected to the O-DU through the F1-U interface;
- The O-CU-UP is connected to each O-CU-CP through E1 interface;
- One O-CU-UP is connected to only one O-CU-CP;
- One O-CU-UP can be connected to multiple O-DUs under the control of the same O-CU-CP;
- The O-CU-UP terminates E2 interface to Near-RT RIC;
- The O-CU-UP terminates O1 interface towards the SMO;
- The O-CU-UP terminates NG-u interface to 5GC;
- The O-CU-UP terminates X2-u interface to eNB or to en-gNB in EN-DC;
- The O-CU-UP terminates Xn-u to gNB or ng-eNB.

O-DU

The O-DU terminates the F1 interface towards the O-CU, the Open Fronthaul CUS-plane, and M-plane towards the O-RU, the E2 interface towards the Near-RT RIC and O1 interface to the SMO.

One O-DU can support multiple O-RUs.

This function typically deals with the lower protocol layers of the stack and therefore is normally responsible for a great deal of RRM functions, RLC&MAC scheduling, and handling of the High-PHY functions of the fronthaul.

O-RU

The O-RU implements the Uu interface and terminates the Open Fronthaul Interfaces (CUS-plane and M-plane) and is a physical node. It also connects to the SMO via the O1 interface.

O-Cloud

O-Cloud is a cloud computing platform comprising a collection of physical infrastructure nodes that meet O-RAN requirements to host the relevant O-RAN functions (i.e., Near-RT RIC, O-CU-CP, O-CU-UP, and O-DU), the supporting software components (such as Operating System, Virtual Machine Monitor, Container Runtime, etc.) and the appropriate management and orchestration functions.

2.2 Network Topology

Network Topology (NT) is defined in [2.8] as the “principle arrangement, ordering or relationships amongst objects and components used in describing a network, without regard to their actual occurrence in any real network”. A topological component as an architectural building block, is used to describe the network in terms of the topological relationships between network functions within the same network layer. These topological components may represent network entities, detailing network functions, protocol functions and entities and interface functions and entities. Figure 12 is an example of a topological representation of an ORAN-based architecture.

NT representations play a central role in the implementation of any Network Management and Operation practice because it provides structure to processes like data collection, data visualisation, troubleshooting analysis. In the case of using of AI for the automation and enhancement of management and operational procedures, NT provide a reference for the correct definition of data structures, feature engineering, statistical analysis, rules, and policies. This set of reference elements are an essential part of AI algorithm development, testing, validation, and regular functionality in a production environment.

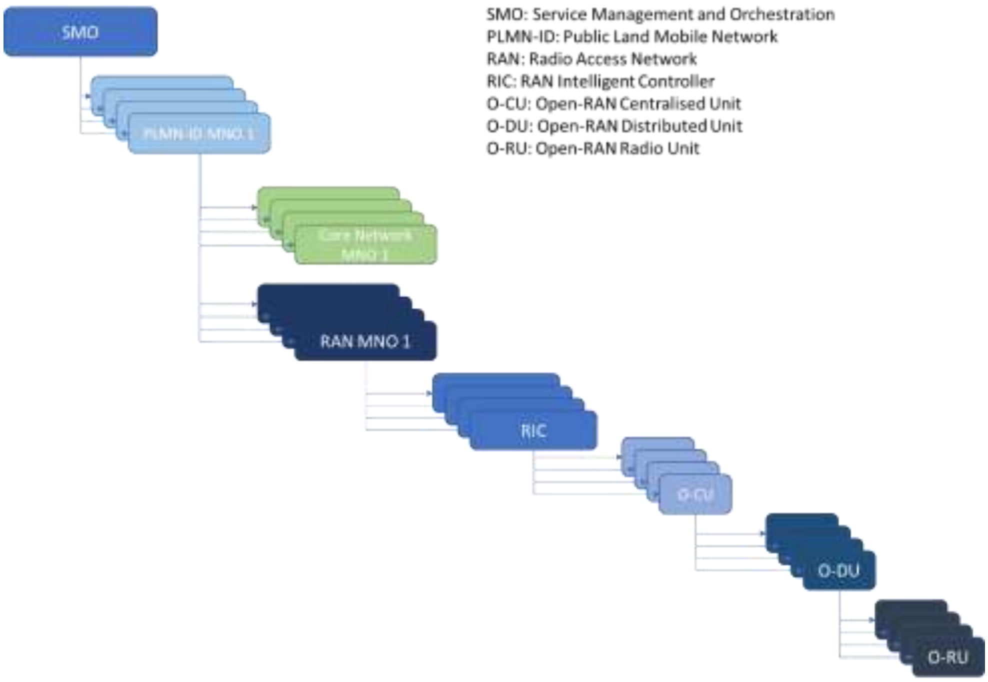


Figure 12 - Example of a Network Topology representing the implementation of an ORAN based network.

2.3 Data Sources

As previously discussed, O-RAN architecture follows FCAPS framework for the implementation of O&M procedures across the different Network Functions (physical and virtualised). FCAPS has also been a common feature across the different network management frameworks that we have studied.

This makes us to assume that TMN FCAPS is the most suitable framework to be used in the future work within AIMM's Work Package 5.

Using the FCAPS framework, provides guidance on the expected information types and data sources for the execution of network management related procedures. Having a clear inventory of the possible data sources, data types and structures is an important step for structuring the development of xApps, rAPPs and other RIC related functionalities as it can be seen from Figure 13 where data plays a central and fundamental role on the RIC architecture.

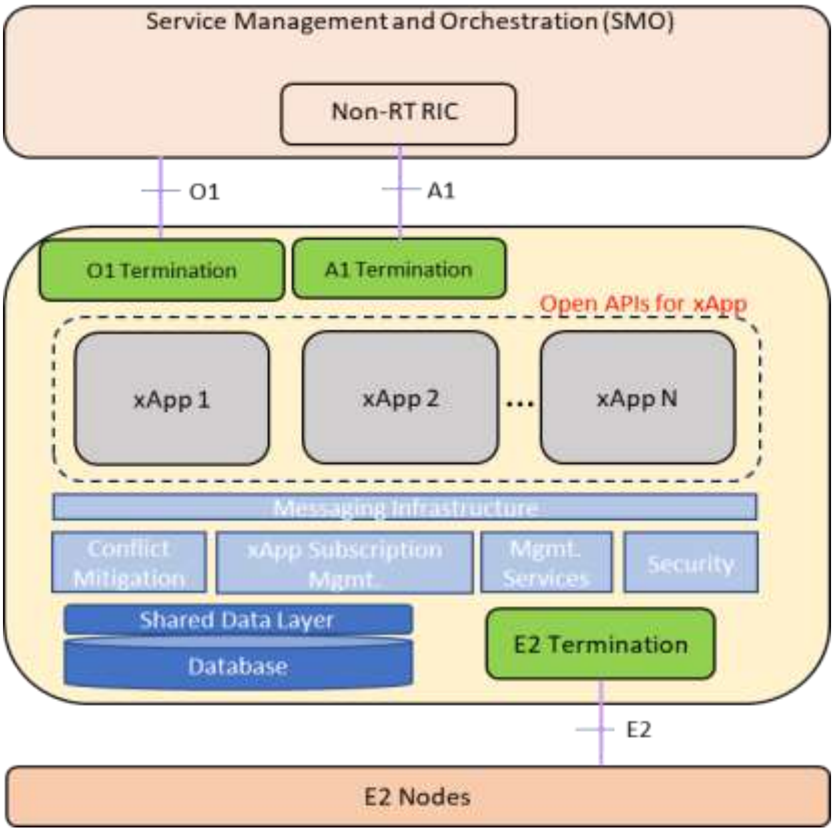


Figure 13 - Near-RT RIC Internal Architecture [2.6]

2.4 Network Analytics Platform

The consortium worked together to understand the challenges that present to the development cycle of Machine Learning (ML) and in specific Reinforcement Learning (RL) from O-RAN systems. From the view of algorithms development, firstly, in O-RAN, data related to model training is difficult to obtain and process. Standard interfaces defined in the O-RAN structure, such as E2, can access DU, CU and other components to collect information inside the network. This data comes, by default, in raw format and without a schema which is not suitable to be directly consumed by AI algorithms. To effectively implement AI on-top of O-RAN and its interfaces, the multiple raw data sources need, on a first stage, to be collected validated, enriched, transformed, and stored onto an integrated data pool, that prepares it to be processed by data engineering processes (such as application of business rules, creation of KPIs, feature engineering, linkage of data tables according to network topology mapping, etc.) that ultimately enables the application of the algorithms according to the addressed use cases.

On the other hand, an O-RAN network is built on-top of other system components such as IP network and Cloud server infrastructure. The operation and maintenance of these system is crucial for the whole network performance and should be integrated in a holistic network management process that addresses all the components.

2.4.1 Network Platform Components and Data Pipeline

The project has benefited from the consumption of data from a real-life network based on O-RAN standards. The centralized units (CUs), distributed units (DUs), and remote radio units (RRUs) are the main NEs deployed as VNFs, and they produce FCAPS data that describe the network behaviour.

The FCAPS data is a fundamental building-block of the AI development cycle, being part of the training and validation process. Once deployed the AI will also use this data on its normal operation.

This O-RAN network was used as a testbed offering real-live network traffic allowing to bridge the gap between purely computer simulated environments and a real live network, and despite being based on standardized interfaces, it presents the typical limitations of the real-world where features and capabilities are delivered as per a staggered development roadmap and market priorities of each network vendor. This helps the AIMM project partners to understand in more detail, the inherent challenges that will be presented when trying to deploy the AIs developed in a purely computer simulated environments to real-life networks through a xApp/rApp product solution.

Given the standardisation of the RAN Intelligent Controller (RIC) functionality is still ongoing in the O-RAN Alliance, we have implemented an OSS/BSS and Analytics platform, depicted in Figure 14, for the collection, storage, processing, and analysis of the FCAPS data from all the VNFs and PNFs. This platform offers functionalities like the RIC such as:

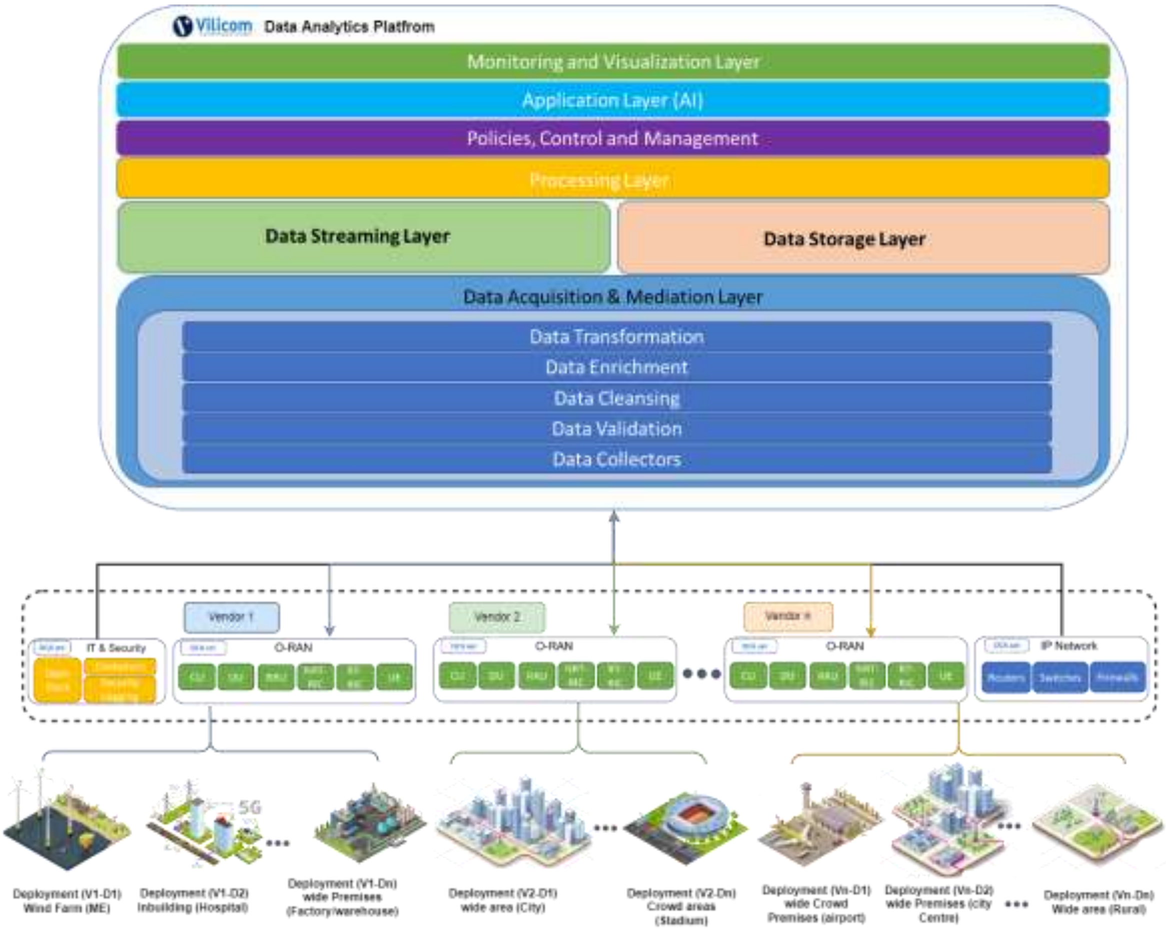


Figure 14 - Vilicom's Analytics Platform to implement Automated Network Management capabilities.

2.4.1.1 Data Collection Agents (DCA)

A DCA is a software application deployed across the network layer i.e. within, or alongside, network NEs and network element managers (NEM), that interact with existing APIs and the NEs. These agents use the standard APIs to collect the standard FCAP dataset. We have identified insufficiencies in these standard APIs and for that we have design these DCAs to be able to connect directly to the NEs and implement data collection processes that have been designed according to the use case.

In an RAN network there are network domains that are implemented using equipment and technology that do not offer open and/or standard APIs. For that reason, it is necessary to develop specific DCA designed to interact with the specific NE API or protocol, etc.

The DCA also has a function of data preparation right from the source, to allow for an efficient and effective data integration coming from multiple and diverse data sources, by normalising the data applying the conventions that have been defined in the system.

The DCA is also responsible for the logging of all its actions and to perform initial data validation procedures. This function is important to trace end-to-end the data pipeline and assist the upper layer of the data mediation stack.

These applications are deployed directly on the management plane of the NE/EM or on any adjacent servers. These have been designed to listen and track the data generated on these sources and capable to pull the logs and send them instantly to the Data Acquisition and Mediation Layer (DAML).

2.4.1.2 Data Acquisition & Mediation Layer (DAML)

The data acquisition and mediation layer (DAML) is the main component in the Data Pipeline and it is responsible for collecting the data by coordinating the DCAs in the south-band interface, data processing and implementing the north-bound interface to the upper layers. This layer is a cluster-based system designed according to Big Data requirements and best practices [2.9], allowing the system to scale and support ultra-dense networks.

After data is collected from the DCAs, the DAML receive it in its raw format, requiring it to be prepared before going through validation and cleansing processes. The DAML needs to add the schema information to the data stream and link it with the network topology. This preparation processes increases the efficiency of the system by reducing the complexity of the data validation, data cleansing.

The DAML is responsible for the data validation and data cleansing processes that consist in validating the data against the expected schema, identifying duplicate records, or missing records, and coping with latency on the data source in making the data records available.

It also prepares the dataset for an optimal application of the data enrichment processes, that would fail if applied directly to the raw data due to missing network topology information details in the file or data structure.

The data enrichment and data transformation functions are tightly coupled with the Data Storage and Processing layers because it prepares the data stream to match the schemas of the data lake and of other consuming applications.

At the end of the DAML cycle, the data offered to the upper layers, is fully integrated, normalised, enriched and transformed according to the system conventions, thus simplifying development of the data lake and of any processing applications (including AI applications). The DAML layers can be continuously improved and extended to consume more – in quantity and diversity – data sources and to offer the data on the north-bound interface in any format, type and frequency that is optimal to the layers consuming the data stream, e.g. the support of multiple AI applications: that consume different metrics, focusing on different parts of the network (different NEs, protocols, etc.); and that might require data in real-time and as opposed to other AIs that consume non-real time data or even historical data.

The DAML coordinates with the DCAs to securely collect the data by implementing encrypted data pipes. It creates one uniform data flow, between each DCA and the upper layers.

2.4.1.3 Data Storage Layer (DStgL)

This layer contains one of the main components of the entire architecture which is the data lake. The data lake is the place where the data is stored to be made available to the upper layers, most importantly the Processing and Application layers. It is designed upon a scalable private cloud object storage; it provides the means to manage and store big datasets that come in diverse formats and structures and enables high throughput and fast access to the data.

The policies, business rules, network topology and other metadata required by the Policies, Control and Management Layer are stored in a dedicated relational database that is managed by the Data Storage Layer.

Business Intelligence techniques and the development of ML/AI applications rely heavily upon wide and diverse historical datasets, for trend analysis, statistical analysis and for ML/AI in specific for model training, testing and validation. This demands for many computational resources and requires DSL to be designed and implemented using big-data best practices [2.10], to deliver optimal access to large scale datasets.

On the other hand, feature engineering and RL related tasks often require high-speed access to many disparate data sources to build and optimise the ML models, this requires high availability of some of data in great quantities and diversity. For this we have designed the data lake following the “Cold, Warm and Hot” approach [3] that is the optimal design meeting our needs.

From the research and deployment stages, we have concluded that despite the structured nature of most of the data in our system, traditional database technologies and Hadoop Distributed File System (HDFS) technology are not suitable to be used in our platform.

Traditional database technologies cannot cope with the required high Input-Output Operations Per Second (IOPS) across many datasets, making it very difficult to optimise the performance at the scale of the datasets being handled by the platform. It also limits the ability to perform complex querying across many and significant sized datasets and hence reducing the value of analytical processes consuming data from implementations using this technology.

In the case HDFS, the limitations are associated with:

- the cost of maintaining the original dataset and two more replicas, that demands extra storage resources, when compared to the implemented data lake technology design;
- Its inability to handle online data streams, mostly when the number of simultaneous data streams is high;
- its inefficient design in handling small files, that wastes resources when storing this data;
- its requirements of high maintenance effort.

The data lake is directly accessible by the other layers such as DM, Processing and AI Layer through a high throughput network. The design behind this storage system allows us to easily store petabytes of data and serve applications regardless of the data access requirements.

2.4.1.4 Data Streaming Layer (DStreamL)

The Data Streaming Layer (DStreamL) handles the continuous flow of information, inside each pipeline, managing multiple data sources in an integrated pipeline stream delivering these to real-time processing applications and analytical visualisations. This type of real-time data source streaming provides a valuable benefit to the business, by guarantying that the data is made available to the PL in the right format and standard, and a continuous low latency and scalable pipeline bandwidth are achieved.

There are many important applications that use streamed data, and they solve complex business problems. In this work we’ve implement an anomaly detection model that requires a continuous stream of data being fed with low latency. This anomaly detection model is an example of an application where the data is processed in real-time before even being stored in the data lake. These use cases are important when we consider the case of near-RT RIC and RT-RIC where the latency of the decision-making process must be kept in order of magnitude of milliseconds, and we cannot afford having the data stored before being processed.

On the other hand, working with applications that have a continuous data processing requirements is not an easy task. There are many things to be considered when such applications are needed in business. Fault tolerance, complex data sources and complex network systems with extensive topologies [2.8] are some of the challenges that must be addressed in the design of the DStreamL.

For this reason, we’ve designed the DStreamL to be distributed across a cluster of multiple nodes deployed in different geographies, where data is replicated so that the system becomes:

- resilient to failures
- scalable to increasing data streaming throughput and user demand on the PL.

2.4.1.5 Processing Layer (PL)

The Processing Layer (PL) is composed by multiple applications deployed over a containerised environment that scales-up with the increased demand from the services of the upper layers such as the Application and Visualisation Layers.

The PL handles mainly three types of jobs, the distributed real-time computation, distributed batch processing and jobs related with AI models such as environment states, reward calculation, AI model training/testing, etc.

AI and ML applications are complex and hard to develop, maintain, optimise, and deploy because of its iterative and multi-staged life cycle. Complexity arises mostly from the stages that involve feature engineering, model training, model testing/validation and production deployment. On the other hand, the RL has more components to consider which are the environment, reward calculation, and the agents which make deploying these applications more challenging.

As emerged in ML-Ops practices, the main enhancement to solve the challenges of AI lifecycle is to containerize all stages. The PL has been designed and implemented to follow this principle and overcome this challenge.

The PL allows the deployment and execution of services that underpin AI applications throughout its entire life cycle. In addition, to this it also implements all the services that involve data processing such as KPI calculation, real-time processing, alarm processing, online monitoring notifications, rule enforcement and data preparation for visualisation. This layer works in tandem with the lower layers such as DStgL and DAML, to provide a containerised environment that simplifies the deployment and management of resource-intensive applications and guarantees high-throughput access to the data pool through dedicated and purpose-built data streams.

This layer will help to encapsulate the works in subphases where the task could be updated separately without affecting other phases, we illustrate some of the main jobs in this layer as follow:

- KPI (Key Performance Indicators) calculation:

  Network key performance indicators are calculated based on formulas defined within the 3GPP (3rd Generation Partnership Project) standards. These specification documents (e.g., for 5G follow 3GPP TS 28.554) contain the KPIs description and formulas. These KPIs need an elevated level of domain expertise to develop and deploy across the data pool. The purpose of these KPIs, includes but is not limited to, the monitoring and troubleshooting of the network performance and long-term trend analysis of its performance. However, they are valuable features to build AI models and to reflect environment status. By abstracting this layer, we intend to save time and reduce complexity. The KPIs are calculated per raw, hour, day, week, month, and year. The results are eventually stored with the collected performance metrics for usage by the AI engineers.

- Feature Engineering and Real-time data processing:

  The processing layer will also run applications that process streams and batches of data. This layer is where the feature engineering process is done. This abstraction considers the requirement of the RL-Ops.

- RL/ML related components:

  The components needed to train, test and validate the AI application. These containers and the related applications are integrated in the whole platform so that they are able to cooperate with other containers and services offered in the processing layer.

Additionally, the processing layer, can run environment simulators images and integrate them to the data pipeline.

2.4.1.6 Policies and Control Layer (PCL)

The Policies and Control Layer (PCL) is composed by a set of configuration methods, services and metadata that define and implement the business rules, object hierarchy and relationship across that are relevant for the functionality implemented across the Data Mediation, Data Storage and Processing Layers.

The O-RAN FCAPS data, produced across the multiple VNFs and interfaces, is the most representative and important data type in this platform. This data being structured, it is not generated its raw format, with the whole information that is required its representation and to be integrated with other data sources. This layer contains the rules, metadata, and methodologies necessary for the efficient and effective implementation of the cycles the DAML, DStgL and DStreamL, allowing to create the structures to validate, cleanse, enrich and store the data in an optimal format.

The network topology metadata and methods are fundamental for the linkage of the different managed objects and data structures, thus enabling the cross-layer analysis between network performance events and external events described by data sources - that are external to the O-RAN network – that are relevant for the analytical process, e.g., UE-based data that describes QoS and QoE events through detailed metrics and logs.

On the other hand, this layer also stores the policies and rules that control some aspects of the system's cognitive capabilities, such as identification of abnormal behaviour and the respecting self-healing action/decision. These policies and rules can be defined by:

- SMEs through processes of data engineering, feature engineering and/or analytical engineering;
- and by automated analytic processes, possibly based in AI/ML applications that identify rules/decisions that after being validated and accepted by SMEs are later deployed on to production.

2.4.1.7 AI Application Layer (AI)

The AI layer is where the development, initial training and validation of the AI model happens. It allows to implement online training through real-time data consumption and offline model validation generating results/decisions that are not implemented rather validated by the developers and the subject matter-experts. It also allows to monitor logs and track performance of the AI jobs and related application images mostly for testing and debugging purposes.

2.4.1.8 Data Visualization Layer

This layer is mostly dedicated to implement Business Intelligence functions that allow for Subject Matter Experts to access the data, and produce graphical reports and dashboards, thus providing a visual interface to monitor the overall system performance.

Through this layer it is possible to access reports and dashboards that inform about the performance of the different system components through the monitoring of dedicated measurements. The components that are monitored are:

- O-RAN network equipment, VNFs, protocols, interfaces, and functions: this allows for the Network Management SMEs to evaluate network performance, identifying opportunities of optimisation, trends of systemic behaviour and evaluate the impact that AI algorithms might have on the overall system performance
- Internet Protocol (IP) network equipment.
- Operational Technology (OT) infrastructure that implements the private cloud that underpins the operations of Vilicom's ORAN-based network.
- AI application decision-making logging: this allows for the DevOps, MLOps and RLOps engineers to evaluate the performance of these applications during the entire life-cycle from training to operations. It also allows to visually report the results of correlation and causation analysis, emphasised on the evaluation of the applications decision on the system performance.

2.4.2 Network Platform System Development and Versioning

The platform design considers the RL-Ops approach [2.11] by encapsulating the main phases in AI life cycle to a micro service approach to continue the development and deployment of the platform.

The use of a Continuous Integration and Continuous Delivery (CI/CD) pipeline [2.15] (Figure 15 and Figure 16) approach allows to abstract the development stages of each platform layer from the production environment, which contributes to continue deploying upgrades to the productions system without disrupting the system. It allows for the simplification and automation of the testing and deployment activities whilst offering strict control, which contributes to the fostering of collaborative work between multiple teams underpinning the simultaneous development & deployment of many AI applications.

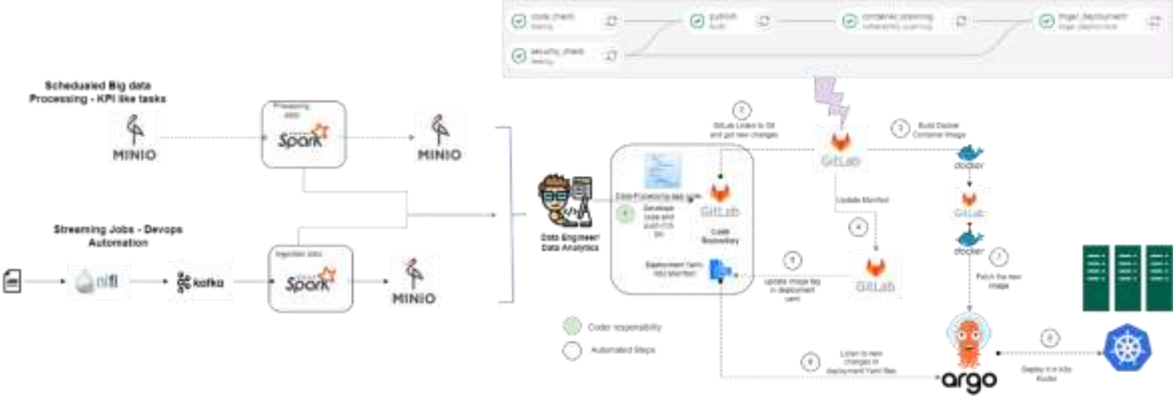


Figure 15 - Architecture of implemented CD/CI Pipeline for Data Engineering

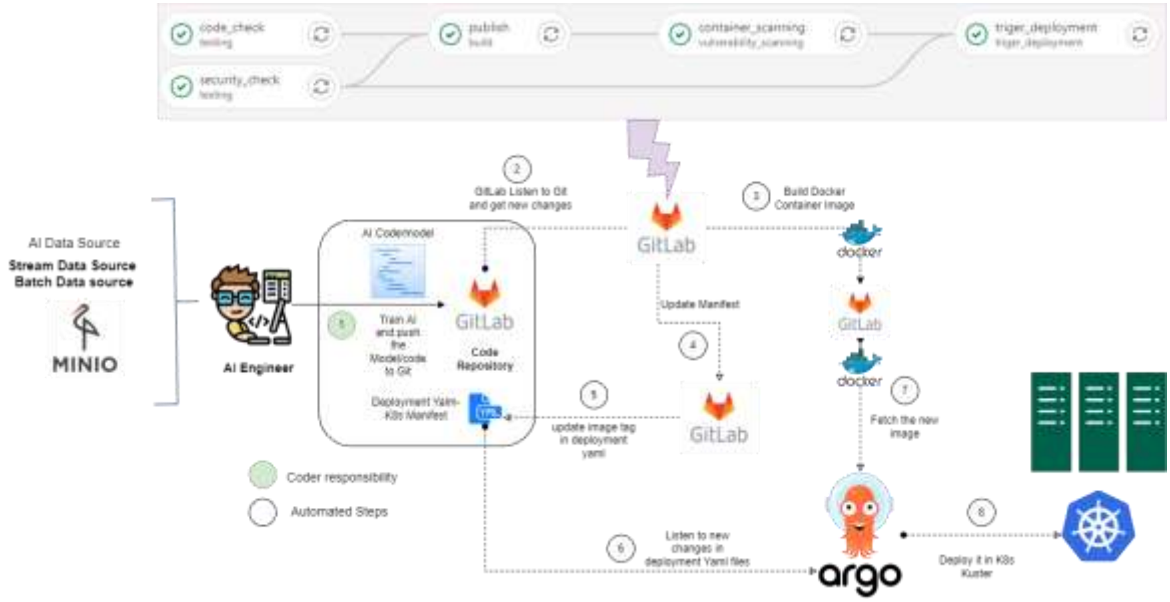


Figure 16 - Architecture of implemented CD/CI Pipeline for AI models

2.5 Vilicom's Analytics Platform

ORAN Alliance technological approach relies on a programable architecture [2.1] that has data as its core asset. The amount of generated/processed data and the wide range of functionality enabled by this architecture, increases substantially the complexity of the processes responsible for handling the entire development lifecycle, especially in complex network deployments with multiple vendors, multiple RATs, and multiple tenants. Vilicom platform, in Figure 14, has been designed and prototyped to deal with challenges in managing Big-Data life cycle, deploying analytical processes, deploying automations and ML/AI models that work coherently across multiple virtualised network instances overcoming the complexity of multiple network tenants, RATs, vendors, and domains. Vilicom Analytics platform deals with the challenges presented to the development cycle of Machine Learning (ML) and specific Reinforcement Learning (RL). The platform is designed to be compatible with Big Data requirements and easy to scale when the network scales up.

2.6 Main Benefits of Vilicom's Analytical Platform

There are multiple benefits stemming from the development of this platform, from efficiencies in data integration, reducing cost through the automation of data processing and decision making and value created through the discovery of new knowledge that adds onto the existing business intelligence.

There are benefits to all operational and management plane related processes through the integration of multiple data sources, that are generated by the network functions and interfaces of the multiple in-operation Public Land Mobile Network (PLMN) tenant instances, into the same data platform/stack with a common pipeline structure, whilst maintaining strict compliance to the network segmentation, security and data confidentiality principles, that are guaranteed by the interworking of the data storage layer (DStgL), data processing layer (DPL) and data policy & control layer (PCL).

The data integration capability implemented in the analytical platform allows for the industrialisation of the mediation process after the initial development of generic data collection and mediation software components. This capability reduces substantially the effort associated with the deployment of data mediation components specific to network instances from unsupported equipment vendors, since it requires simple configuration changes to adapt the generic functionality to support network functions and other software components from vendors and RATs that are not yet supported by the platform.

The deployment of new PLMN instances using network functions, from supported vendors, is also streamlined and improved by this integrated approach since the generic mediation layer components are easily deployed and scaled-up, with minor configuration changes, across the cloud infrastructure using the principles of containerisation and resource orchestration.

As depicted in Figure 17 the savings in terms of efforts associated with each stage of the data analytics pipeline, decrease from the more generic set of processes and activities to the more specific and use case driven activities, due to a lower effort in application/process implementation activities once the data has been structured and modelled properly from the initial design of each data source pipeline.

Among the benefits introduced by this design is that this integration of the data, generated by the cloud wide network functions, into an application programming interface (API) offering multivendor and multi RAT: Performance Management (PM) Key Performance Indicators (KPIs); Fault Management (FM) alarms; and Configuration Management (CM) objects; to the development cycle of Machine Learning (ML) and Artificial Intelligence (AI) models, reduce the data collection bottleneck, accelerates the whole process of model training, model validation and model deployment to production [2.9]. The API allows the adaptation of the data structures and models consumed by the ML/AI development cycles according to the specific use case and problem statement for which each model is addressing.

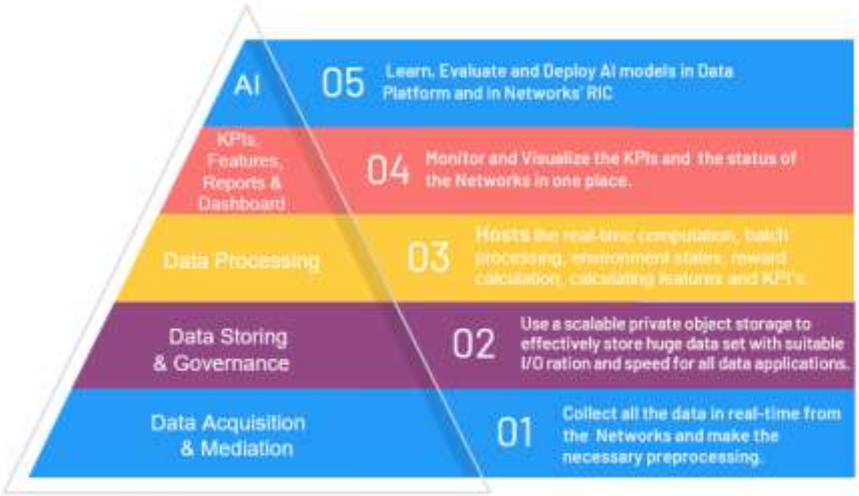


Figure 17.The main platform phases with respect to efforts

- **All in one platform**
  - Unified platform for all data applications regardless of the data access type. This will support applications that need real-time access or big batch access to build stream applications, analytical logic, dashboards or even AI models. This feature is built to be compatible with Big Data Specification [2].
- **Access to all data from one place**
  - Since different network instances might be deployed based on different software vendors, which increases the complexity of the processes and activities related with the network management plane, the fully integrated nature of this platform's design allows to enable ML/AI development for multi-vendor and multi-RAT (with all its specific) scenarios. This significantly reduces the effort of operating and maintaining the cloud infrastructure, by effectively underpinning the automation of processes and streaming of the data pipelines across the entire analytical life cycle.

- **Centralized/Federated AI applications**
    - As all the relevant data is hosted in this platform it becomes more affordable to develop and deploy AI applications. Collecting more variant data will help build more robust models. This design also supports working with multivendor Radio Intelligent Controller (RIC) environments and implements some functions of the Non-Real Time RIC.
    - It also supports the development of Federated Learning applications by hosting the global model in the processing layer.
- **Less work for new deployments**
    - The generic nature of the data collection and mediation layer components, allows to repurpose existing interfaces by changing its configuration. Once it’s done, it is possible to consume data from new deployments using software components from vendors and RATs that are new to the platform without additional effort.

2.7 AI deployment use cases

In general, there are many possible ways to deploy AI in production environments, however, centralized and distributed Artificial Intelligence are yet considered the main enabler in 5G networks [2.12]. In this section, we introduce the two main methods to deploy AI in ORAN architecture using this platform.

Case 1: AI Federated Learning

Federated learning (FL) algorithm [2.13] is a communication and computation efficient framework for distributed learning approaches. FL algorithm exploits the possibility of keeping data local and collecting model updates/gradients from deployed AI workers to update a global/centralised version of each model. Therefore, there are several benefits of implementing FL in O-RAN architecture.

Federated Learning in Vilicom platform

Vilicom Cloud platform offers an example of using AI application in ORAN with multiple vendors and multiple deployments. It’s possible to face a case where we have some application interested in solving a specific network management problem like power optimization. On the other hand, it may not be feasible to develop “n” AI models for “n” networks/cases. This issue arises when there are big differences between the characteristic of each network. For example, a network deployed in industrial settings address substantially different requirements, through specific design and configurations, to those from networks deployed in a city centre or a sports arena/stadium. The FL framework can play a relevant role in solving such problem relying on local data. In each case, a model deployed in a RIC instance (possibly multiple RICs, from multiple vendors, across several deployments) and managed by Vilicom’s analytical platform, where a global model is deployed and maintained. This global model is, constantly updated by adjustments coming from the distributed/local deployed models, and it is used as a template for the deployment of new local instances to manage local network deployments. Figure 18 depicts a deployment scenario of FL in O-RAN using Vilicom’s Analytical Platform to manage its entire life cycle.

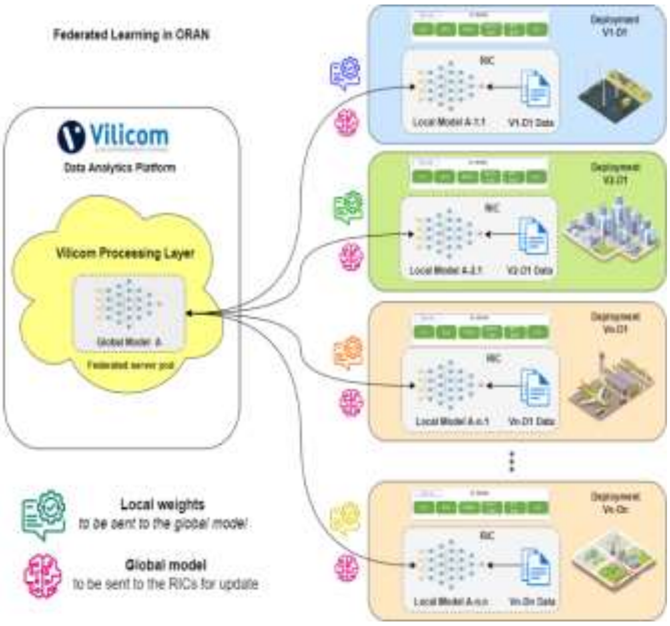


Figure 18. Federated learning in ORAN managed by Vilicom's platform.

Case 2: Centralized AI learning

Vilicom platform is also designed to support the development of centralized AI applications benefiting from a consistent and coherent data acquisition and mediation layer, across every data source. This simplifies the AI development lifecycle for non-real-time applications that require enormous amounts of historical data. Anomaly detection is an example for these types of AI applications.

Figure 19 shows the cycle of building centralized AI applications in ORAN architecture using Vilicom Data platform. The model is executed inside the Vilicom Processing Layer that acts as a global non-real-time RIC function.

The innovation here is that model can be trained, tested, and validated in one place, Vilicom's platform, and the final accepted model can be put into production within the same layer or in instances specific to each one of the deployed PLMNs.

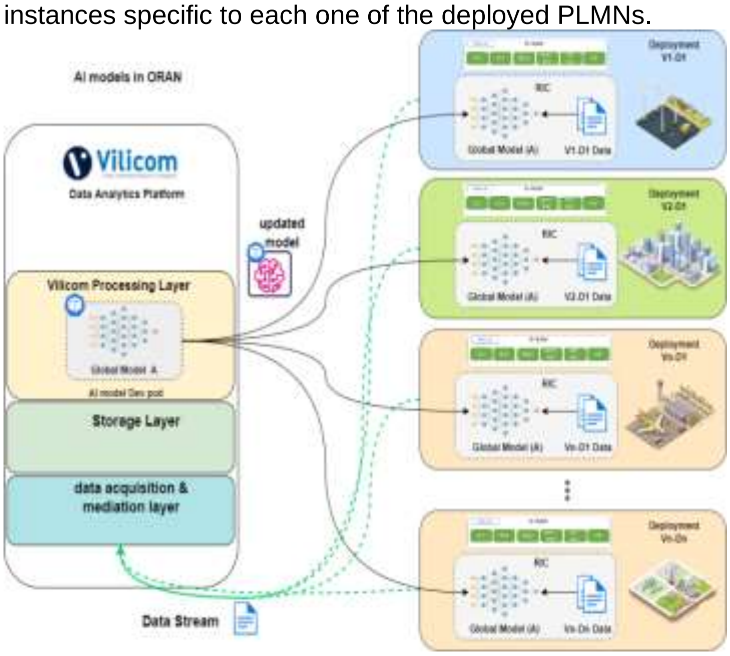


Figure 19. Centralized learning in ORAN managed by Vilicom's platform.

2.8 Use case and Results

The work on this task culminated in the development of a solution to a real operational challenge in a live ORAN network following the IBM data science life cycle methodology [2.14], as shown in Figure

20. The steps are categorized into three main phases that needs people from different skill set and knowledge. The main purpose of this proof of concept is to test the benefit of the platform in a practical exercise.

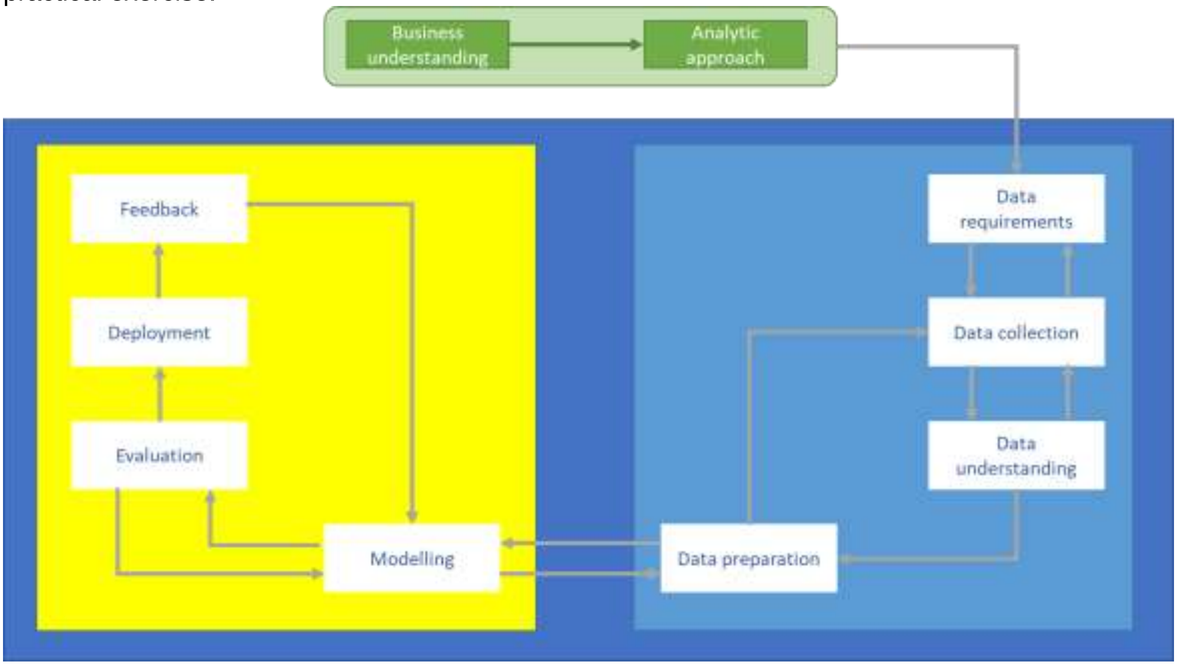


Figure 20 - IBM Data Science Methodology - Main Phases [2.14].

The three main stages of this methodological process are:

- Business related (Highlighted in green),
- Data related phase (Highlighted in light blue)
- AI related phase (Highlighted in yellow).

Vilicom RAN Optimisation engineers, worked as Subject Matter Experts (SME), alongside Data Scientist, providing business knowledge that was fundamental in defining the problem statement and an analytical approach that could be consistent with the data sources available.

At a second stage a data engineers together with Business SMEs determine what type of data is required to implement the analytical approach defined to address the problem statement. This is often done by identifying the FCAPS data sources that are available coming from the NEMs or directly from the NEs. In some cases, it is necessary to create agents to survey specific network behaviours that are not captured in the FCAPS information, mainly through the processing of UE logs and associated analytical procedures.

In this project, the entire set of data engineering processes and activities have been implemented in Vilicom's platform which is responsible for the data collection, preparation, and processing of all the data sources across all the data pipelines. Several data pipelines have been implemented to address different use cases or problem statements. An example of different pipelines being created to address a specific use case is the need to segregate, for security and confidentiality reasons, throughout the analytics platform the data that is generated by the set of CNFs – that implement NEs and NEMs – of each specific public MNO PLMN-ID that is a tenant on Vilicom's Private Cloud.

On the other hand, inside each PLMN domain, different network optimisation problem statements might require dedicated data pipelines to feed specific deployed AI-models or Visualisations.

The analytical platform resolves the complexity of managing the development, deployment, and operation of all these different data pipelines through a process of CI/CD pipelining [2.15].

The third stage of the data science methodology proposed by IBM [2.14], has also been implemented through the architecture and toolset available in Vilicom's Network Analytics Platform. AI engineers work alongside Data Engineers and business SMEs to define and implement the AI-models' lifecycle.

The people involved in this project, throughout the three stages, didn't have shared skills and shared expertise, but the implementation, in the analytical platform, of the automations that implement the data pipeline facilitates the development lifecycle of data driven applications, by codifying and formalising all the methodology activities.

2.8.1 Business/Operational analysis stage

The problem statement, addressed by the work of this project, describes the need to create a real-time anomaly detection model that enables engineers monitoring the network performance to take decisions to resolve performance degradation events. The existing FCAPS data generated by the NEs and NEMs was not fit for the purpose of this analysis because of its reporting granularity – both for time and topology domain. The degradations were not noticeable in five-minutes aggregated data and neither at the cell level aggregation. For this reason, the SMEs and Data Engineers involved in this work identified that the best data source should be UE logs and API metrics. They defined that through the consumption of this data a set of trend and pattern analysis rules should be implemented to generate alarms to be displayed at the operational awareness dashboards.

2.8.2 Data stage (feature engineering)

Most of the effort to implement this process is spent at this stage (as depicted in Figure 17) to automate the data and feature engineering processes that prepares the data for suitable consumption by the data driven applications and visualisation layers.

The analytics platform is designed to automate these procedures to underpin the work of AI-model development by reducing the complexity of integrating and deploying new micro data processing applications.

Figure 21 depicts the functional architecture of the analytics platform detailing the technological components used in the data engineering stage. These architectural components implement the bottom five-layers of the analytics platform (Figure 14).

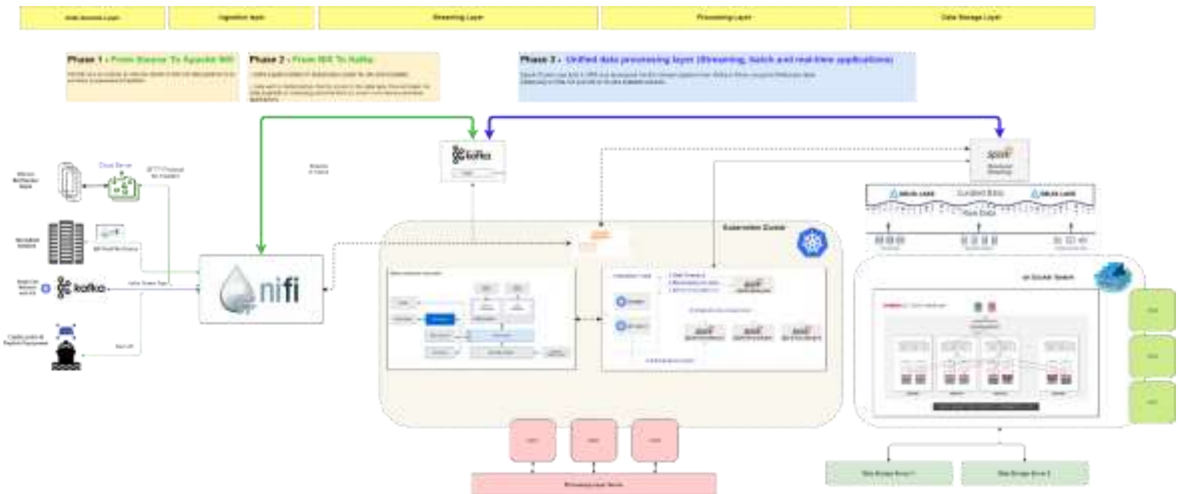


Figure 21 – building data processing applications for data driven apps (including AI)

The use of the analytics platform, and its CI/CD pipeline, in this stage allowed for the simplification of the whole process, by providing a framework and toolset (implementing the DCA, DAML, DStgL, DStreamL and PL layers), that:

- simplifies and fastens the deployment of existing data collectors across pipeline domains and the development of collectors for any type of new data sources
- integrates the data making it easier for data exploration and quicker understanding of the data structure and its content
- and allows the flexibility/agility to build the data validation, data cleansing and data enrichment tasks preparing the data to be consumed by the upper layers
- reduces the overall amount of effort involved in this stage of the data science methodology process
- produces quality output that is ready to be used by the AI engineers throughout the lifecycle of the AI-model.

In the context of our project, at this stage, the platform was collecting data in real-time from several UEs of the live network and preparing it to be processed using the previously defined analytic approach.

The output of this pipeline process, to the upper layers, are the following parameters: RSRP, RSRQ, DL throughput, UL throughput, timestamp, cell id, PCI, latency, longitude, latitude and of the UE distance from the site providing the service.

2.8.3 AI Development and Deployment stage

This stage encapsulates an iterative process that communicates with the data engineering stage and implements the AI-model necessary to achieve the desired outcome as defined in the business/operational analysis stage.

Specifically for this project it was developed a system that is described in Figure 22, to implement a Long Short-Term Memory Neural Network (LSTM NN) model [2.16].

The raw data (RSRP, RSRQ, etc.) is first processed by a training features generation module. Then, considering that the data set is unbalanced, the model uses a weighted sampler, which means that the batch sampling process samples the data according to the proportion of each label. After the batch sampling, the data is fed to the sequential LSTM layer, followed by a full connected layer (FCN). The output of this FCN is the signal of alarm: ‘0’ means “No Action Needed”, while ‘1’ means “Action Needed”.

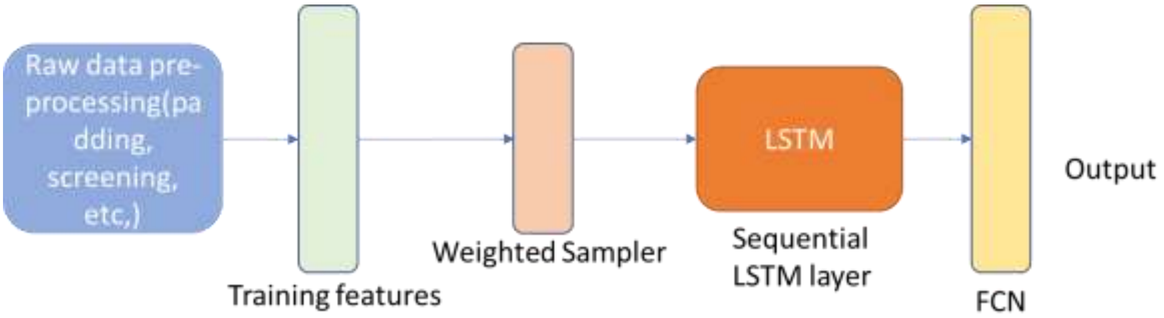


Figure 22 - The system diagram of the proposed LSTM NN model.

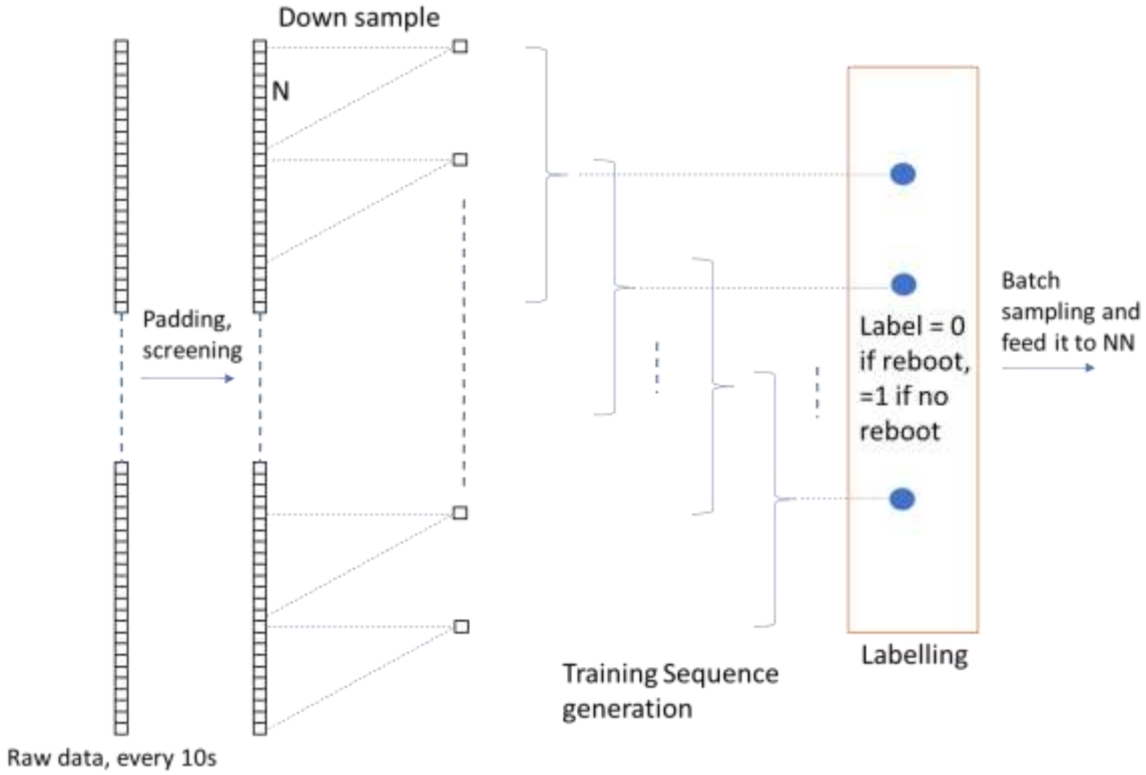


Figure 23 - The details of the raw pre-processing data module as part of the entire LSTM-model.

Figure 23 details the data pre-processing module. As the kernel of the designed NN is the LSTM, the raw data must be processed as sequential data in the time-domain. This process consists of 4 steps:

(1) paddling and screening to fill up the missing records and remove the mistaken records.

(2) downsampling to calculate the mean value of the records of two different time-series, coming from two modems of the same device.
(3) training sequence generation to wrap the raw data to a sequence, which can be fed directly into the LSTM;
(4) label the sequence according to the historical records of actions taken by the engineers. Figure 24 shows the labelling process of the time series collected from the modems “cell 1” and “cell 2” will be labelled independently.

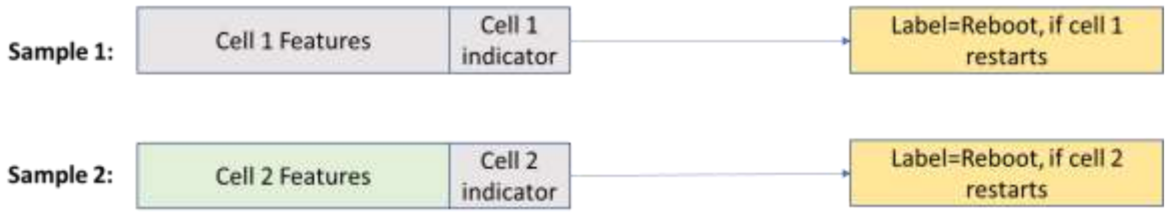


Figure 24 - The labelling process

The development, validation and deployment of the AI model was done in the platform, thus significantly improving the overall process by reducing the complexity for the AI engineer and reducing the development time, because it abstracts the data engineering activities into a separate readily available process.

When the model passes the validation, the deployment is done instantly by leveraging the CI/CD pipeline as shown in Figure 16.

The outcome of this stage is a validated LSTM model that was acceptable to be deployed in production automatically.

2.8.4 The final product

2.8.4.1 LSTM model

The final product is a docker image containing the last updated LSTM model with two functions to consume data and write results. This image is a result of collaboration between the data engineering stage and AI-model development stage where the AI engineer train, test, validate and accepts/updates the AI model, whilst the Data Engineer (using DevOps practices [2.18]) adjusts/adapts the two functions - input source function and output sink function - as depicted in Figure 25.

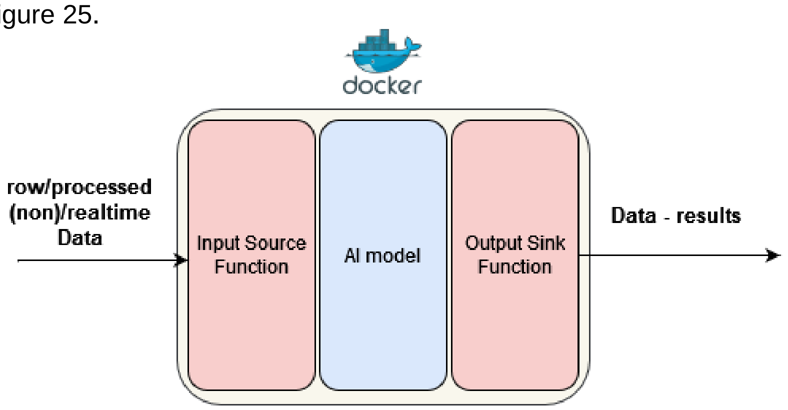


Figure 25 - Final AI artifactory (The Docker Image). Tasks for Data Engineer are in red, Task for AI Engineer are in blue.

The input function collects the data in stream of raw periods of ten seconds and prepares it through statistical aggregation into data records of one minute granularity to feed the AI model. The AI model then predicts the action that will be handled by the output function where the data is stored as per the correct schema and in the correct lakehouse [2.19] table to be visualized instantly in the dashboard and thus feed the recommendation to the network operations engineer.

2.8.4.2 Visualization Results and Anomalies’ Alarming

The Visualisation dashboard, in Figure 26, takes a single-pane-of-glass [2.20] approach to network performance monitoring, combining a set of relevant network Key Performance Indicators (KPIs) to monitor the Quality of Service (QoS) offered by the network, UE Key Quality Indicators (KQI) to monitor user experience and Radio Frequency (RF) metrics to monitor coverage levels and quality, to provide context when analysing the anomaly alarms predicted by the LSTM model.

In this same dashboard we also publish the results of a heuristic application that has been developed to validate that the AI-model prediction results follow the engineering rules defined by the network operations SMEs.

This approach will be followed for any other AI models that we develop and deploy in the future.

Figure 26 – Single-pane-of-glass dashboard for AI alarms, PM and KPIs.

Figure 27 shows the data visualisation for the UE location that is traveling in the vessel in the sea. The colour code follows a gradient between recent data records in bright red and older records in light blue.

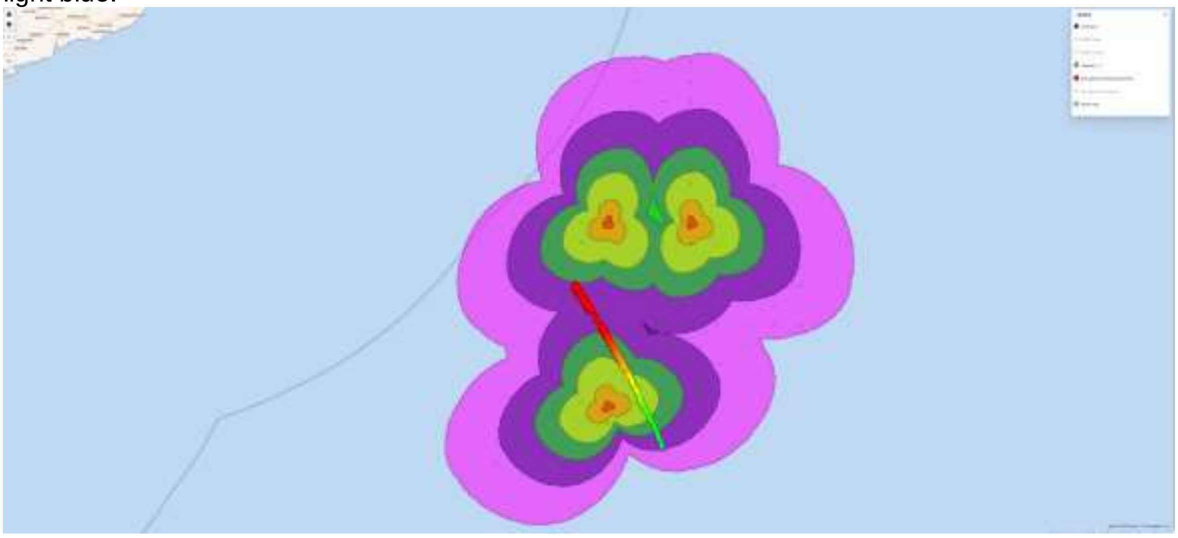


Figure 27 - UE movement in the coverage area. the UE symbol colour refers to collection time.

Figure 28 is another example of a single-pane-of-glass dashboard that integrates geographical information system (GIS) data, with RF metrics, the UE logging data and the AI-model predicted data to allow the network operations engineer to understand the trends and validate the model results.

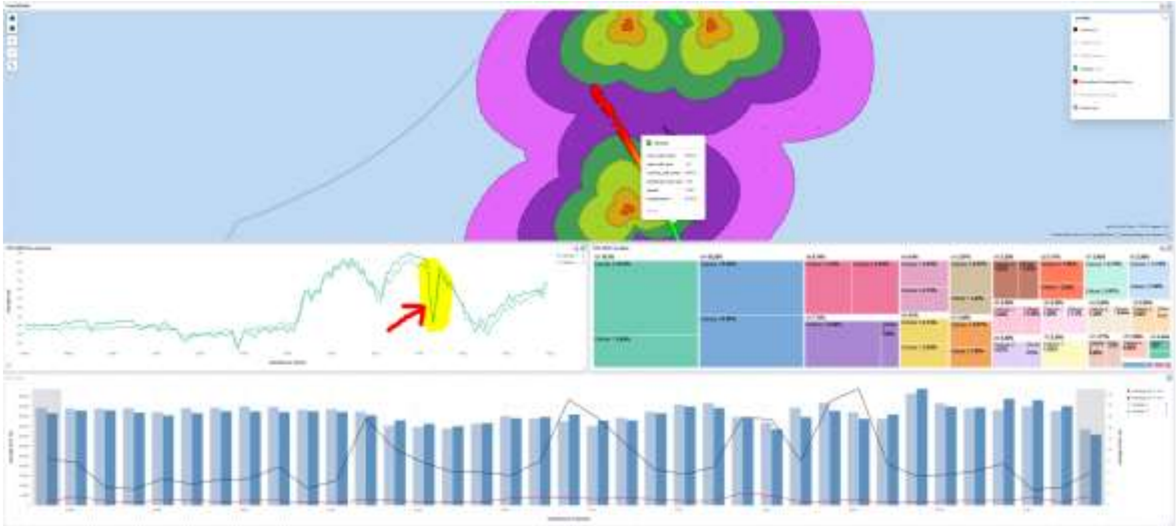


Figure 28 - Visualizing Alarms with PM, KPIs and data from the UE. The abnormal case was detected and highlighted in yellow.

Any changes/updates committed to the data pipeline, KPIs, the AI-model or the dashboards will be automatically deployed via the CI/CD pipeline in seconds, following a GitOps approach [2.21].

These results validate the importance of the concept and architecture presented in Figure 14.

2.9 Future Work

Vilicom's Analytics Platform is now part of our managed services business unit, and it is used to automate operational processes in our Service Operation Centre (SOC) and Network Operation Centre (NOC) and assist our engineers on their decision-making process.

For the future we are looking to extend the number of operational use cases and processes to automate, through the development of more AI-models and Visualisation models.

Further research and development efforts will be put into the automated provisioning in the network of the decisions taken by the AI-models, then closing the cycle of intent driven network optimisation applications and creating a network model that is more cognisant.

2.10 Section 2 - References

[2.1] ITU-T M.3010 Principles for a Telecommunications Management Network https://www.itu.int/rec/T-REC-M.3010

[2.2] ITU-T M.3400 Telecommunications management network - TMN management functions, 02/2000, https://www.itu.int/rec/T-REC-M.3400/en

[2.3] ITU-T M.20 Maintenance philosophy for telecommunication networks, https://www.itu.int/rec/T-REC-M.20-199210-I/en

[2.4] ITU-T M.3050.1 (02/2007) Enhanced Telecom Operation Map (eTOM) – The Business Process Framework https://www.itu.int/rec/T-REC-M.3050.1/en

[2.5] ETSI GS NFV-MAN 001 V1.1.1 (2014-12) Network Functions Virtualisation (NFV); Management and Orchestration https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf

[2.6] ORAN-WG1 Architecture Description - v01.00.00 https://www.o-ran.org/specification-access

[2.7] 3GPP TR 38.801 V14.0.0 (2017-03) Specification # 38.801 (3gpp.org)

[2.8] ITU-T M.3017 Framework for integrated management of hybrid circuit/packet network https://www.itu.int/rec/T-REC-M.3017

[2.9] Y. Roh, G. Heo and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," in IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 4, pp. 1328-1347, 1 April 2021, doi: 10.1109/TKDE.2019.2946162

[2.10] Wo L. Chang, Nancy Grady. 2019. NIST Big Data Interoperability Framework: Volume 1, Definitions. National Institute of Standards and Technology, US Department of Commerce.

[2.11] P. Li, J. Thomas, X. Wang, A. Khalil, A. Ahmad, R. Inacio, S. Kapoor, A. Parekh, A. Doufexi, A. Shojaeifard et al., “RLOps: Development Life-cycle of Reinforcement Learning Aided Open RAN,” arXiv preprint arXiv:2111.06978, 2021.

[2.12] T. Subramanya and R. Riggio, "Centralized and Federated Learning for Predictive VNF Autoscaling in Multi-Domain 5G Networks and Beyond," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 63-78, March 2021, doi: 10.1109/TNSM.2021.3050955

[2.13] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communcation-efficient learning of deep networks from decentralized data,” pp. 1–11, Feb. 2017. [Online]. https://arxiv.org/pdf/1602. 05629

[2.14] Following the data science methodology - IBM Developer

[2.15] What Are CI/CD and the CI/CD Pipeline? | IBM

[2.16] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997 Nov 15;9(8):1735-80. doi: 10.1162/neco.1997.9.8.1735. PMID: 9377276.

[2.17] F. A. Gers J. Schmidhuber and F. Cummins "Learning to forget: Continual prediction with LSTM" Proc. Int. Conf. Artif. Neural Netw. pp. 850-855 Sep. 1999.

[2.18] What is DevOps? | IBM

[2.19] Armbrust, M., Ghodsi, A., Xin, R., & Zaharia, M. (2021, January). Lakehouse: a new generation of open platforms that unify data warehousing and advanced analytics. In Proceedings of CIDR. https://www.cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf

[2.20] Sukhija, N., Bautista, E., James, O., Gens, D., Deng, S., Lam, Y., & Lalli, B. (2020, November). Event management and monitoring framework for HPC environments using ServiceNow and Prometheus. In Proceedings of the 12th International Conference on Management of Digital EcoSystems (pp. 149-156). https://doi.org/10.1145/3415958.3433046

[2.21] What is GitOps? | GitLab

3 Interactive environment development

3.1 5G-NR system-level simulator

Recent developments in the field of Artificial intelligence (AI) provide new capabilities of generating automated solutions for network management functions. Specifically, Reinforcement Learning (RL) is an approach for dynamically controlling and solving Markov Decision Processes. An RL intelligent agent learns to make sequential decisions by interacting with the environment. Other options include neural network and deep learning methods. To gather information and train the intelligent agent, an accurate simulation framework for network management is necessary.

The 5G-NR simulation and development environment, also known as “AIMMSim”, is a system-level simulator which emulates a full cellular radio system following 5G concepts and channel models. Further it is a discrete event simulation framework that maintains a queue of pending events internally which is invisible to the programmer. All functions and classes have default arguments appropriate to the simulation of a 5G macrocell deployment at 3.5GHz.

The intention is to have an easy-to-use and fast system simulator written in pure Python with minimal dependencies. It is especially designed to be suitable for interfacing to AI engines such as ‘TensorFlow’ or ‘PyTorch’, and it is not a principal aim for it to be extremely accurate at the level of the radio channel. For the latter task, pre-computed look-up tables (based on simulated channel models) are used to obtain fast run-times. If a more precise link-level model is required, a simulator such as ns-3 can be used. The AIMM simulator normally operates without a graphical user interface, and simply writes logfiles for subsequent analysis. The default logfile format is tab-separated columns, with purely numerical data. These files can then be easily processed with shell utilities such as cut, head, tail, etc., or read into Python or R scripts, or, if all else fails, even imported into spreadsheet programs. However, a custom logger can create a logfile in any desired format.

The AIMM project addresses two aspects of AI in the RAN. The first, “bottom-up” approach, is to use AI to optimise the air-interface performance and enable the practical implementation of antenna structures and network architectures. The second, “top-down” approach, is to incorporate data collection features coupled with AI functionalities in order to facilitate RAN intelligence and automation at the system level. Work on the top-down approach has been largely based on simulation techniques. Typical requirements considered whilst developing AIMMSim framework are:

1. The simulation framework needs to be interactive. The agent learns how to perform under different environment states. The large number of different instances should be imported by an RL intelligent agent to be trained. The simulation framework should also be able to receive and perform the policy generated by the RL agent.
2. The framework needs to offer self-evaluation mechanisms for the states. In other words, the RL agent needs to receive feedback on how well is doing. This information needs to either come from the simulation framework itself, or from an individual evaluation module.
3. A desirable feature is parallel sampling ability. To train an RL agent, the exploration of the environment is necessary. If parallel sampling is possible for the simulation framework, the training process of the RL agent would be more efficient.
4. The simulation framework needs to rely on the O-RAN interface for interacting with the RL agent. This means any RL agent developed on the simulation will be one step closer to real-world deployment. Furthermore, this clears the avenue for combining training data from both the simulation and the real-world as the agent can be trained on both interchangeably.
5. The simulation framework needs to rely on the OpenAI Gym interface. OpenAI Gym is the de facto environment wrapper for RL research and development. It ensures proper interaction with the RL agent. Most RL algorithms are written with this interface in mind so ensuring it is implemented here will expedite the development cycle.

3.1.1 Architecture

The following factors have influenced the overall software architecture:

1. The software architecture should closely mimic the real system, with a class for each type of network component.
2. The components should exchange traffic in a similar way to the real system. However, “traffic” here is an abstraction; there is no concept, for example, of IP packets, or of resource blocks at the physical layer. These constraints are imposed to get

sufficient speed from the simulator, to get as many ML training episodes as possible, in each given time.

3. There should be a RIC module (radio intelligent controller), at the top level of management. The AI or ML methods will operate solely in the RIC, effectively as xApps.
4. The simulation technique should be the discrete-event method. In the core of the simulator, a queue of pending events in maintained. Most events will be periodic (such as UE reporting), and an easy-to-use framework is provided for this. The discrete-event method has negligible overheads and allows easy mapping to simulated time to real time.
5. Subbanding (division of the channel into sub-channels which may be dynamically reallocated between cells) is implemented on all Cell objects, but the number of subbands may be set to 1, effectively switching off this feature.
6. All simulations take place in three spatial dimensions, for example, to allow modelling of high office buildings. Some simple capabilities for accounting for wall losses in indoor scenarios are provided.
7. Dynamic features of a specific simulation are handled by a Scenario class. This can, for example, move users according to some mobility model.
8. UE handovers between cells will be handled internally by a heuristic based on RSRP (received signal reference power), as in real system. This is implemented in the MME class. However, for research into smart or AI-based handover strategies, this default heuristic can be overridden.
9. In fact, all modules can be overridden or have their default behaviour modified if desired, using the usual subclassing technique.

Software design considerations

The following factors influenced the software design:

1. The core simulator should be monolithic (meaning that only one import will be needed by applications) but will not implement plotting or post-simulation analysis. These can be done better by existing tools.
2. The output of a simulation run will be a logfile in a standard format (by default, tab-separated columns). The lines in the logfile are constructed and formatted by an instance of the Logger class.
3. For testing and debugging purposes, a realtime plotter is provided as a separate program. This reads and plots the logfile as it is generated, through a shell pipeline.
4. Python was chosen for portability, ease of development, and ease of interfacing to existing AI software.
5. Extensive use of Numerical Python (numpy) means that most of the code is running at the level of compiled C code. Sufficient speed is thus attained.
6. External dependencies are kept to a minimum; essentially the only one is simpy to handle the event queue, but little of its capabilities are in fact used, and simpy could easily be replaced by a small local module.
7. Sensible defaults are provided for all system parameters, such as operating frequency, channel bandwidth, etc.
8. Implementations are provided for several 3GPP standard channel models.
9. Extensive online documentation, with a full set of tutorial examples, is provided at https://aimm.celticnext.eu/simulator/.

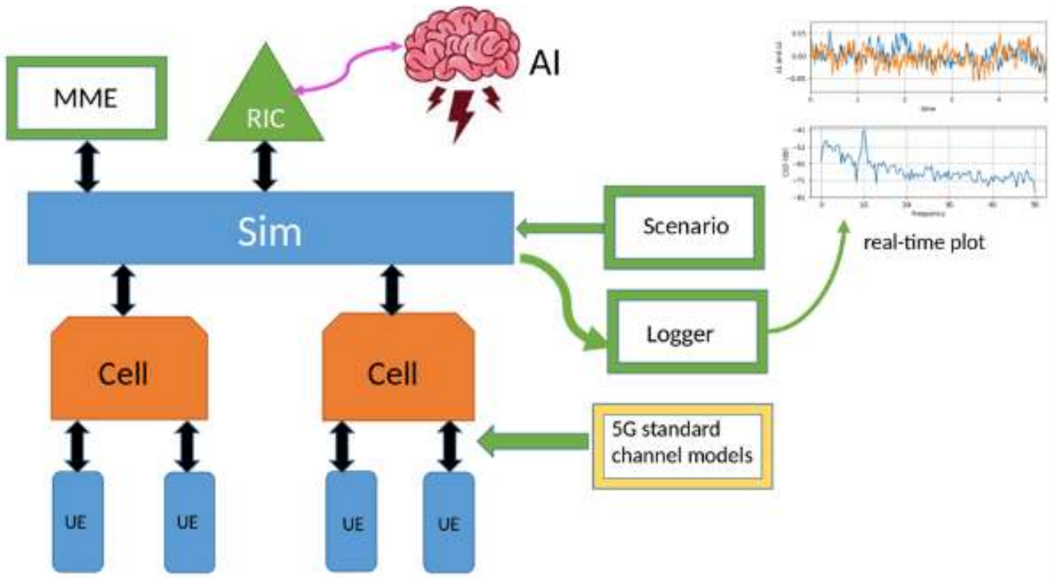


Figure 29 - AIMMSim Block Structure

Outline of usage principles

The basic steps required to build and run a simulation are:

1. Create a Sim instance. The represents the complete simulation.
2. Create one or more cells with make_cell(). Cells are automatically given a unique index, starting from 0.
3. Create one or more UEs with make_UE(). UEs are automatically given a unique index, starting from 0.
4. Attach UEs with the method attach_to_best_cell().
5. Create a Scenario, which typically moves the UEs according to some mobility model, but in general can include any events which affect the network.
6. Create one or more instances of the Logger class.
7. Optionally create a RIC, possibly linking to an AI engine.
8. If necessary, create a custom Logger class by subclassing.
9. Start the simulation with sim.run().
10. Plot or analyse the results in the logfiles.

A complete simulation code demonstrating these principles is in Figure 30.

```
from numpy.random import standard_normal
from AIMM_simulator_core import Sim,Logger,Scenario,MME,np_array_to_str

class MyScenario(Scenario):
  def loop(self,interval=10):
    while True:
      for ue in self.sim.UEs: ue.xyz[:2]+=20*standard_normal(2)
      yield self.sim.wait(interval)

class MyLogger(Logger):
  # throughput of UE[0], UE[0] position, serving cell index
  def loop(self):
    while True:
      sc=self.sim.UEs[0].serving_cell.i
      tp=self.sim.cells[sc].get_UE_throughput(0)
      xy0=np_array_to_str(self.sim.UEs[0].xyz[:2])
      self.f.write(f'{self.sim.env.now:.2f}\t{tp:.4f}\t{xy0}\t{sc}\n')
      yield self.sim.wait(self.logging_interval)

def hetnet(n_subbands=1):
  sim=Sim()
  for i in range(9): # macros
    sim.make_cell(xyz=(500.0*(i//3),500.0*(i%3),20.0),power_dBm=30.0,
      n_subbands=n_subbands)
  for i in range(10): # small cells
    sim.make_cell(power_dBm=10.0,n_subbands=n_subbands)
  for i in range(20):
    sim.make_UE().attach_to_strongest_cell_simple_pathloss_model()
  sim.UEs[0].set_xyz([500.0,500.0,2.0])
  for UE in sim.UEs: UE.attach_to_strongest_cell_simple_pathloss_model()
  sim.add_logger(MyLogger(sim,logging_interval=1.0))
  sim.add_scenario(MyScenario(sim))
  sim.add_MME(MME(sim,verbosity=0,interval=50.0))
  sim.run(until=2000)

if __name__=='__main__':
  hetnet()
```

Figure 30 - AIMM Sim complete code example.

3.1.2 System Evaluation and Benefits

The AIMM system-level simulator allows easy construction of large-scale 5G network simulations, with a clean interface (through the RIC class) into standard AI software packages. Because the RIC class has privileged access to internal cell data, as well as permission to set operating parameters in cells, it is the right place to place any AI or ML components. Furthermore, current developments such as implementing xApps with communication via Google protobuf can be accommodated by putting a simple translation layer in the RIC. Thus, the current design is essentially agnostic regarding messaging protocols. Current enhancements being planned include a tracking of energy consumption in each network component, allowing use in green radio projects. At the completion of the AIMM project in September 2022, it is intended to release the code as open source.

3.2 AIMMSim as Digital Twin

Assessing customer experience and network quality of service is of utmost importance for global mobile operators as it provides the ability to optimise network performance based on current needs and demand. A comprehensive investigation of the potential of DT, in particular, a one-way DT for 5G Radio Access Network (RAN) and beyond employing network data for a range of use cases has been performed. The developed one-way DT also referred as “AIMMSim” has been utilised to study performance issues, generate valuable insights, and produce data-driven policies in dynamic as well as static environments. These informed policies generated are envisioned to assist human-in-the-loop such as network planners and/or radio engineers to resolve edge cases, false positives taking better network decisions in the real world. These simulations provide more control on validating machine learning models’ predictions to avoid bad AI decision-making as it could be costly for the business.

The ML technique implemented is reinforcement learning (RL), a branch of AI and class of ML that employs a reward and punishment policy to enable an agent to learn a solution to a decision problem by interacting with its environment purely through trial-and-error. The motivation to develop RL enabled algorithms is the ability to learn a solution without any prior knowledge of the environment or the reward function. For instance, if new base station(s) are to be deployed in a particular area where no historic data is available that might lead the RAN planners to optimal location for BS deployment, a RL algorithm by continuous interaction with the environment may learn an optimal policy and assist in taking effective decision. A few of the use cases explored are (a) radio coverage prediction (b) smart interference management (c) mobility management and (d) power optimisation.

The findings from utilising DT for RAN are progressive and could significantly assist in the development of more robust and reliable solutions with less operational and maintenance costs. We now focus to develop UE traffic patterns using real network data in urban, sub-urban and rural environment using the real data measurements from the network. The intent is to then input these developed UE traffic patterns into the developed twin and analyse the network performance. The motivation to do this activity is (a) to map simulation environment as close as possible to the real world (b) minimise the assumptions that are considered when utilising the mathematical mobility patterns which are considered to either simplify the complexity or reduce simulation time.

3.3 Variational Autoencoder Assisted Neural Network Likelihood RSRP Prediction Model

3.3.1 Introduction

The advent of the fifth-generation (5G) network provides a unique potential for UEs through three application themes namely enhanced mobile broadband (eMBB), ultra-reliable low-latency communications (URLLC) and massive machine type communications (mMTC). These network advancements bring challenges to the complex heterogeneous radio access networks (RAN) and ultra-dense networks (UDN) such as BS deployment, radio coverage and capacity planning. Mobile network providers rely on a series of key performance indicators (KPIs) to understand, analyse, and assess network performance for coverage and capacity. One of the KPIs for coverage analysis is signal strength - reference signal received power (RSRP). To measure RSRP values as well as other network metrics drive tests are usually performed. It requires significant human efforts, explicit hardware and substantial capital expenditure (CAPEX). Moreover, the measurements recorded via this method are limited. It only reflects network performance for a short period for a particular location that lacks comprehensive spatial-temporal data collection and assessment. To overcome the challenges for RSRP measurement using the drive test, the 3rd Generation Partnership Project (3GPP) in Release 9 introduced the Minimisation of Drive test (MDT) methodology. Here, radio measurements are collected using an individual’s mobile device that is logged into the network. Each UE feedback its network experience to the associated BS.

Generally, there are two approaches for predicting RSRP. The first is via the 3GPP standard-based empirical model. Based on field measurements from different terrain and scenarios, such as urban, rural, suburban, macro or microcells, it summarizes the propagation rules based on a large number of test values. Typical empirical-based path loss models as documented in 3GPP TR 38.901 [3.1]. Such modelling depicts the channel properties in a general and coarse way, which may not be accurate enough for specific environments. The second method is the data-driven approach. Instead of modelling the propagation model, the correlation between environmental features in an area and the corresponding RSRP values are directly estimated. This approach needs a large amount of historical RSRP data that could be obtained through MDT. However, data collected by MDT has the following issues (a) the signal strength between UE devices at the same location and time could differ more than ±6dB [3.2]. (b) Inaccurate location information for indoor UEs. (c) Some locations only contain limited data points due to imbalanced UE distribution. Estimating and predicting RSRP values from limited and inaccurate measurements leads to ineffective Quality of Service (QoS) analysis. Recently, efforts towards employing Artificial intelligence (AI) or machine learning (ML) techniques in RAN are progressing swiftly. One example is the introduction of the Radio Intelligence controller (RIC) module in the radio network architecture of Open RAN (O-RAN) [3.3]. ML-enabled algorithms provide feasible and accurate solutions to use cases such as intelligent coverage analysis, smart handover, load balancing, etc. In this work, we propose to use an ML model for RSRP prediction, which assists in assessing network coverage in the focused area.

This simulation work presents a generative model for accurate RSRP prediction based on a well-designed neural network (NN) architecture. We not only utilise the historical real data of RSRP but also consider the geographical statistics information. The correlation between geographical information and RSRP distributions is then mapped through data compression. Regarding the extraction of environmental features, we construct images that can reflect the transmission environment from BS to UEs, using them as auxiliary training features of the RSRP prediction model. The BS-UE association modelling is accomplished by a modified digital twin (DT), while the process of feature extraction is completed by a variational autoencoder (VAE), with a convolutional neural network (CNN) as a backbone. Once the model is trained, the encoder of VAE serves as the environment feature extractor in this work. The low-dimensional latent variables will be used as the environmental features to assist the RSRP prediction. In regard to the RSRP prediction model, a multi-layer perception (MLP) trained in a supervised learning manner is applied. Due to changes in the transmission environment, the RSRP value recorded at each location is time-varying.

From a statistical point of view, the RSRP values recorded at this location conform to a normal distribution. To estimate the normal distribution, the MLP model is designed as a likelihood model. It takes the output of the aforementioned encoder and BS-recorded features as inputs and outputs the mean and variance. The main contributions of this simulation work are summarised below:

- We propose a likelihood NN model for the RSRP prediction considering the distribution of real RSRP values.
- We propose to use a VAE to learn the environmental auxiliary features from the geographical map generated by a DT. This VAE can be trained offline separately, which does not affect the training efficiency of the likelihood model. The features are used to assist the training of the NN model.
- To the best of our knowledge, for the first time, a joint training prototype employing a two-tier neural network for radio coverage prediction is put forward, which benefit from both a digitized simulation model and real data.
- We validated the proposed model using real network data which illustrates superiority over the empirical model.

3.3.2 Background

3.3.2.1 Empirical-based model

Empirical-based models are a set of models summarised from a large amount of measured data in different scenarios. The Log Distance Path Loss (LDPL) propagation model is the most representative one which has been adopted widely [3.4]. LDPL treats the power at location $l\ j$ as a log-normal random variable, depicts the relationship between the received power and BS to UE distance $\| \ lBS - l\ j \ \|2$, which can be represented by:

$$P(l_j) = P_0 - 10n_j \log_{10}(\| \ l_{BS} - l_j \ \|_2 / d_0) + w_j \quad (1)$$

where $d0$ is the close-in reference distance, which is determined from measurements close to the transmitter, nj and wj are adjustable coefficients determined by the propagation environment [3.5]. After delicate tuning and testing of these parameters, there are several standardised empirical models for propagation modelling that are largely used by academics as well as in industry such as the COST 231 Hata model [3.6], Okumura model [3.7], Walfisch-Ikegami model [3.8], WINNER II Propagation model, etc. However, in the presented work, classical ML techniques and a linear regression model have been employed on the real network data for network radio propagation assessment.

3.3.2.2 Data-driven approach

The data-driven approach aims to use the historical data of RSRP to analyse the relationship between RSRP and environmental changes over time and space, to achieve fine grained, site-specific modelling. The intensive development of ML provides a powerful engine for this type of approach. For example, a random forest (RFs) based predictor considering a rich set of features that includes location, time, cell ID, device hardware and other features has been proposed in [3.4]. The paper demonstrates the benefits of using fewer measurements and achieving higher accuracy in real-world data sets. Ref [3.9] utilises a Regional Analysis to Infer KPI (RAIK) framework to establish a relationship between geographical data and user data using crowdsourced measurements. A radio wave propagation prediction based on backpropagation (BP) NN and a simplified path loss model is proposed in [3.10]. Ref [3.11] proposed a two-step algorithm for RSRP map generation by regression clustering.

However, the challenge for the data-driven RSRP prediction is that there are only a limited number of training features available. In the current work latitude, longitude, altitude, transmission frequency, timestamp, and cell ID are a few of the parameters that were used. The available MDT data does not fully reflect the channel variation. In the NNbased approaches, this limitation will usually result in the underfitting of a NN model. Hence, how to generate more auxiliary features to assist the training process of the NN model and improve the model performance is a significant challenge. Except for the radio features recorded by the BS, some researchers propose extraction and utilisation of environmental features. It is known that geographical statistics influence the signal quality, so the characteristics of the transmission environment, such as the UE distribution and maps, the obstructions in the transmission path, the height of buildings, etc., are integrated with the BS-recorded features.

The representative of this method is ray-tracing, which is a popular approximation using geometrical optics and knife-edge diffraction theory [3.12]. An ML-based 3D propagation ray-tracing model for the cellular network is studied by [3.13]. Thrane et al. proposed a channel model using deep learning (DL) and a simple path loss model aided satellite image, in which the path loss modelling was also finished by a raytracing model [3.14]. Zhang et al. introduced a CNN-based NN to reduce the computational complexity in the ray-tracing model [3.12]. Despite these papers claiming the advantages of less ray-tracing time and the potential in improving path loss prediction, it is far from a complete solution because the proposed NN ray-tracing models are trained in a supervised way, which inevitably need to execute ray-tracing module and collect dataset in advance.

3.3.2.3 Digital Twin

“A digital twin is a digital representation of a physical item or assembly using integrated simulations and service data” as defined in [3.15]. A DT provides high-fidelity representations of all components of the current live mobile network, including service and UE behavioural characteristics [3.3]. With the maturity of image processing technology based on NNs, pure image-based environmental feature extraction schemes are gaining attention. In [3.16], the geographical features are processed by the open street map (OSM) and expert knowledge is used jointly to learn a prediction model. Yi et al. put forward an environmental feature exploring method by using CNN on the maps of altitude, maps of building height and maps of CI [3.17]. But there are some issues with these methods. For example, the latent features extracted by CNN are sparse, which makes it difficult to identify its effects in training; the CNN-RSRP training is done end-to-end, which makes the model hard to apply to other scenarios. In this study, we utilise a DT, named DRIVE, that was designed by Ioannis et. al. [3.18] to digitalise the BS to UE transmission links in the given scenario. DRIVE is a flexible, modular, and city-scale framework aimed at the vehicular and network simulator. It contains three major functionalities:

- It is designed to parse and simplify the OSM and different types of buildings.
- The SUMO module is integrated to generate the UE mobility traces according to the road situation from OSM.
- It can perform the continuous network simulation according to the UE locations and BS settings.

Although the empirical propagation model is adopted in DRIVE, we utilise its OSM processing and the specific UE generation ability that are valuable in modelling the BS to UE relationship geographically.

It is known that the key factors affecting signal transmission exist in the characteristics of the environment, such as how far the signal transmits, and how many reflections, absorption, and scatterings it encountered during this period, even the building materials and vegetation. But these characteristics are difficult to model accurately. Although ray-tracing methods are aiming to restore the transmission path as much as possible, urban-scale raytracing is too time-consuming and complicated to be realistic. Moreover, as discussed in [3.14], simpler images would improve not only the training of the model and the hyperparameter search but also the final performance of the methodology. Therefore, in this study, we do not seek accurate tracing results. We focus on how to describe the possible impact of the signal transmission path with UE location and environmental information.

3.3.3 The Proposed Two-Tier NN Architecture

We elaborate on the proposed two-tier NN model with emphasis on NN architecture, training data generation and training scheme. As shown in Figure 31, the proposed NN model consists of two cascaded NNs. The first tier is designed as a CNN-based VAE to extract relevant environmental features while the second-tier network is designed as a fully connected network with two heads that outputs the mean and variance of RSRP in each location. The underlying representation of VAE is Z - this parameter will assist the training of the two-tier neural network.

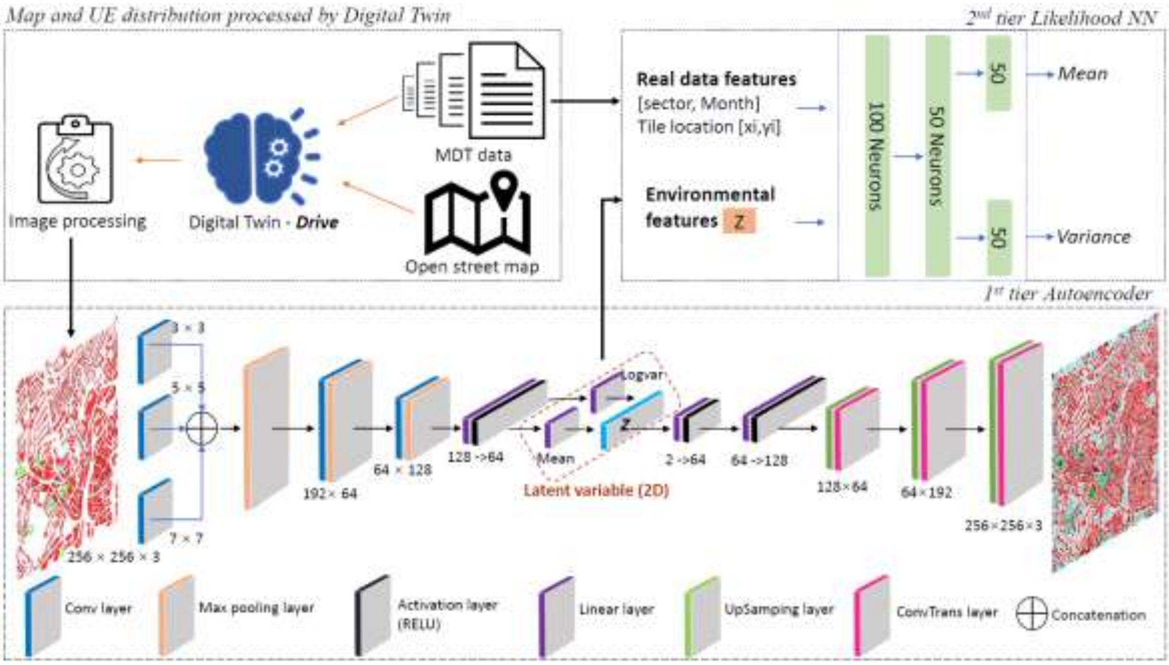


Figure 31 - The proposed two-tier neural network architecture

3.3.3.1 First tier - Variational Autoencoder

1) Data sampling: As shown in Figure 31, the OSM corresponding to the site is imported into the DRIVE simulator. Further, the functions provided by the simulator are utilised to simplify the building and road information to understand the building outline of the involved city. In the map processing, the buildings are accounted for as 2D simple polygons as the real data set described in IV doesn't contain accurate altitude information of UEs. Figure 32 shows an overall association between a UE and a BS. The left side of Figure 32**Error! Reference source not found.** shows the processed map, in which red polygons represent typical buildings, and green polygons represent the foliage. BS is represented by a black circle, and UE is marked by a blue triangle. The right side of this figure shows zoomed areas of BS and UE. The connection between the BS and the UE is highlighted by a light blue line. 10000 such top-view geographical images with resolution 256×256×3 are collected to train the VAE, as described below.

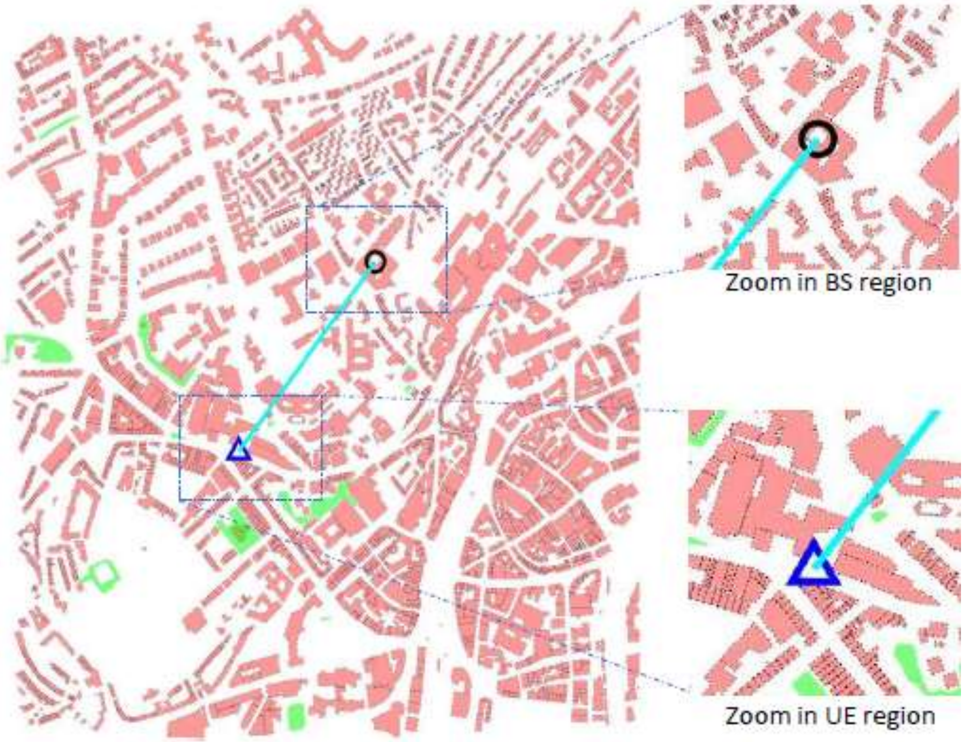


Figure 32 - BS-UE association image generated from DRIVE simulator

2) VAE architecture: VAE is a framework to learn deep latent-variable models and corresponding posterior inference models using stochastic gradient descent [3.19]. It consists of two sections, encoder and decoder, as shown in the lower half of Figure 31. The encoder, also called the inference model, learns the posterior on the low-dimensional latent space over the input data samples. The decoder is a generative model that learns the joint distribution of the latent variables and input data. In this study, the architecture of the encoder is in the form of VGG [3.20]. At the first convolutional layer of the encoder, a set of convolutional filters with sizes of 3 × 3, 5 × 5, 7 × 7 are applied to extract features from different dimensions, respectively, followed by a max-pooling layer. There are two convolutional layers with max-pooling after that. One flattened layer and one fully connected layer with 64 neurons are followed. Next, the output tensor is fed into a two heads output layer with 2 neurons individually. The reparametrized variable Z will be the input of the decoder section. The decoder owns inverse architecture compared with the encoder. A dropout with ratio 0.25 was inserted between the flatten layer to enhance the robustness of training. The loss function is defined in Eq. (2).

$$\mathcal{L}_{VAE} = -\sum_i \left[\mathbb{E}_{z \sim q_\theta(z|x_i)} \left[\log p_\phi(x_i|z) \right] + \mathbb{D}_{KL}(q_\theta(z|x_i) || p(z)) \right] \quad (2)$$

where θ and ϕ are the trainable parameters of the encoder NN and the decoder NN, respectively. $q\theta$ $(z|xi)$ is the posterior inference from input sample i, $p\phi$ $(xi\ |z)$ the generative model given the latent distribution [3.21]. The first term in equation (2) is the expected data log-likelihood (assuming Gaussian probability density function, maximisation of this term amounts to minimisation of the reconstruction mean squared error), and the second term is the KL divergence between $q\theta$ $(z|xi)$ and $p(z)$ which regularises the latent space. The R,G,B channel of each sample i is normalised to the range [−1, 1] for training.

3.3.3.2 Second tier - Likelihood Model

The likelihood model is designed based on an MLP. The detailed architecture is illustrated in the top right side of Figure 31. The first two layers are with 100 neurons and 50 neurons respectively, and the last layer has two heads with 50 neurons in each head, which output the mean and variance of each bin. The overall training feature of this model has been formalised in Eq. (3). It is a 6-dimensional vector, where x_loc and y_loc represent the (x, y) coordinates. The BS is 3-sectored. A UE belongs to one of the sectors of the BS it is associated with. Month specifies the month that data samples are collected. It is worth noting that the 2-dimensional auxiliary feature generated by encoder section Z are also fed into the likelihood model.

$$\vec{\mathcal{F}}_{\text{proposal}} = (\text{x\_loc}, \text{y\_loc}, \text{sector}, \text{month}, \mathcal{Z})^{d=6} \tag{3}$$

The loss function for training is the Gaussian negative log likelihood loss, which is defined as:

$$\mathcal{L}_{Likelihood} = \sum_i \frac{1}{2}\left(\left(\frac{(\mu_i - y_i)^2}{\sigma_i}\right) + \ln {\sigma_i}^2\right) \tag{4}$$

where μi and σi are the numerical outputs of the likelihood NN model; yi is the corresponding label of a sample i. Feature normalisation is also adopted for the likelihood model.

3.3.3.3 Training and Validation

Experiments are performed using the Intel 2 E5-2640v4 CPU, 2 RTX 2080Ti GPU. The data pre-processing is performed by the CPU whilst the training stage relies on the GPU. The training is based on PyTorch. The training and validation set are divided according to the 80% and 20% of the total both for the VAE and likelihood model. The batch size of VAE is 50 and for the likelihood model is 3000. Both models use Adam as an optimiser (default learning rate).

3.3.4 Real World Datasets and Model Evaluation

3.3.4.1 Data pre-processing

The real-world dataset is provided by BT Labs, recorded the monthly data of about 16,000 bins served by one BS. Each bin covers a square of $10m \times 10m$. A bin may also be referred to as a tile in this work. Each sample of data includes the central coordinate position of the tile and multiple recorded RSRP samples. We chose two datasets with significant seasonality, namely January and August, to evaluate our proposed model. The details of the datasets can be found in the first 3 columns of Table 1. Specifically, the number of samples recorded in each sector and months. The datasets do not contain any data involving user-specific information. Due to the uncertainty of the transmission channel, outliers are removed using Hampel's filter. Here, the median of the whole dataset was first calculated, and then the absolute deviation of each data sample from the median was obtained. Also, the median of the deviations was evaluated. We consider any data point greater than the absolute deviation against the value of 4.5×median of the deviations as an outlier. This Hampel's filter is applied on the latitude and then on longitude. The UE distributions of August after removing outliers can be seen in Figure 33.

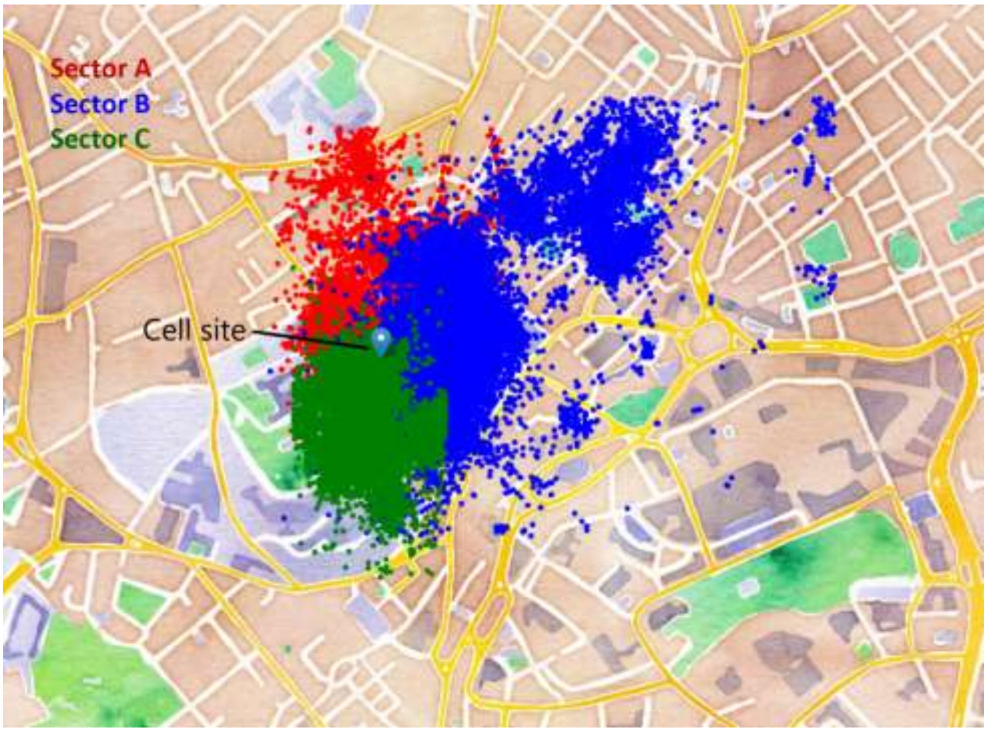


Figure 33 - The UE distribution after removing outliers. The red, green and blue dots indicate UE that are associated with respective sector.

3.3.4.2 Evaluation results

In this section, the evaluation results of the proposed two tier NN model using two real datasets are provided. We evaluate the models in terms of the average RSRP prediction error through a 20-fold cross-validation scheme, and early stopping is adopted in the training with a stop patience of 8. The VAE was trained in an offline way. The VAE models with the same parameters were used to assist different likelihood models training under different months. Linear regression and simple MLP techniques are used as the baseline for compassion. The MLP model doesn't contain the environmental extractor and mimics the same architecture in Section III-B. The training feature of this model is defined in equation (5). To keep the consistency of evaluation, the training and validation set divide is the same as in the proposed two-tier NN model, and both models will be reinitialised for each fold. Three sectors involved in the datasets are validated independently.

$$\vec{\mathcal{F}}_{\text{baseline}} = (\text{x\_loc}, \text{y\_loc}, \text{sector}, \text{month})^{d=4} \quad (5)$$

The evaluation criteria is the mean absolute error (MAE) between ground-truth mean value ai and predicted mean value $\hat{}ai$. The MAE is calculated as

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|a_i - \hat{a}_i|. \quad (6)$$

Data information			Validation results (in dBm)		
Month	Sector	Samples	Empirical	MLP	Proposed Model
Jan.	A	21236	7.29	6.478	**5.840**
	B	53208	7.99	7.323	**6.243**
	C	15172	10.71	6.758	**6.636**
Aug.	A	10699	8.06	7.104	**5.941**
	B	24361	10.08	9.726	**8.623**
	C	20228	8.78	7.790	**7.012**

Table 1 - Data Information and Model Validation Results For Different Data Subset

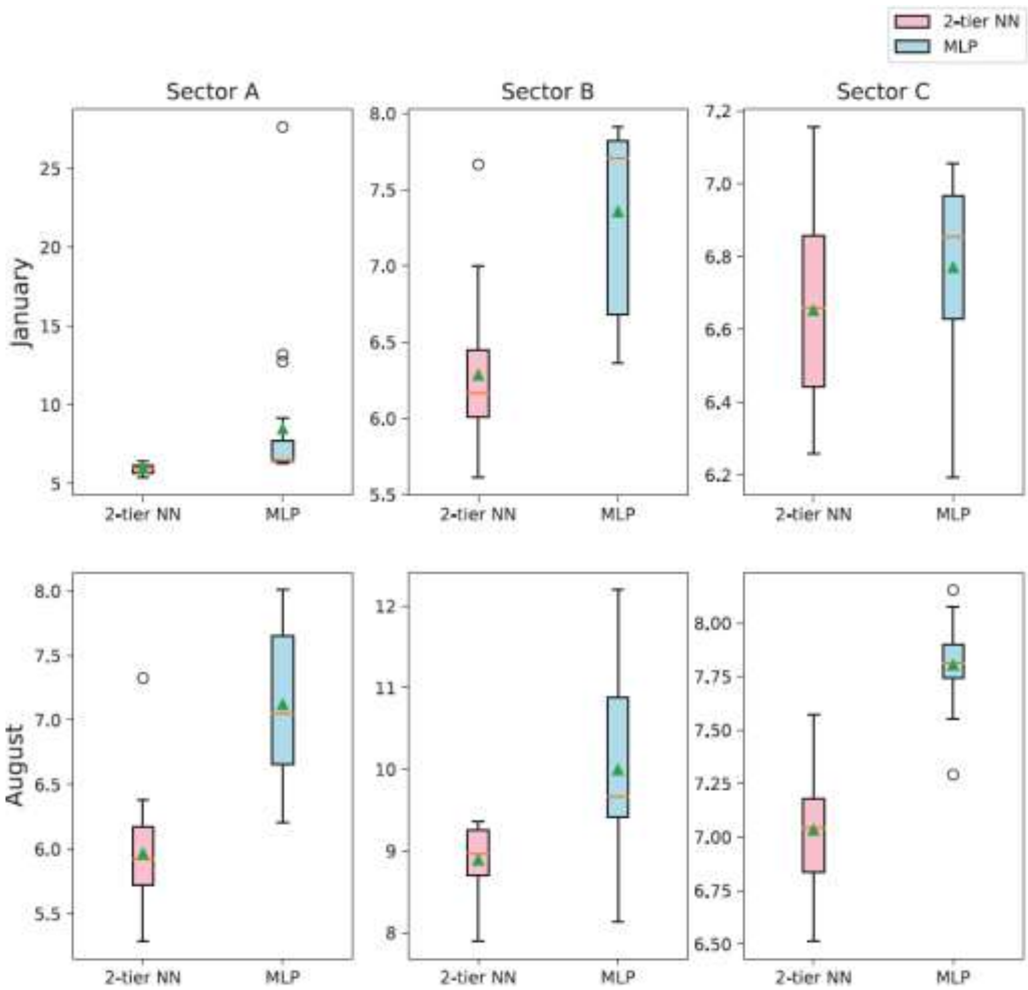


Figure 34 - The boxplots (red for the 2-tier NN and blue for the typical MLP) of cross-validated MAE for RSRP prediction based on different sectors and months.

Table 1 presents MAE results of the empirical model, MLP model and proposed two-tier NN. Compared with the empirical model our proposed model can improve the prediction accuracy by about 20%, and the largest increase accrues on the subset January sector C, where the MAE is reduced from 10.71 dBm to 6.758dBm, about 38%. Meanwhile, compare with the simple MLP model, the prediction accuracy of our proposed model has an improvement by nearly 10%, and the largest improvement lies in the August sector A, around 16.4%.

Figure 34 demonstrates more detailed boxplot results, which summarize the distribution characteristics of the MAE on the test set in 20 fold cross-validation for both MLP and the proposed model. In general, our proposed model can be trained more stable (with fewer outliers) and have a smaller and more concentrated error distribution.

3.3.5 Discussion

This study presents a two-tier RSRP prediction model based on OSM processing and demonstrates gains across different real-world datasets. The training of VAE as an environmental information extractor can be separated from the subsequent network, which reflects good model reusability. Since the actual datasets do not contain the altitude information of the UE, so we do not regard the height as an independent feature during map processing and labelling. But the current model can easily implement the above extensions. In addition, the VAE latent vector is equivalent to regularising the training of the subsequent likelihood model, so the length of the latent vector needs to consider the number of real data features, and an excessively long latent vector will suppress the expression of real data features. The likelihood model involved in this work has the simultaneous output of mean and variance to optimize the loss function shown in Eq. (4). Since in the current dataset, not all tiles have unified multiple samples recorded and obey the Gaussian distribution. So, the output of variance is meaningless for some tiles. Therefore, its value can only be used as a reference for partial tiles.

3.3.6 Conclusions

In this work, a novel two-tier NN architecture is proposed to realise the accurate RSRP prediction. The VAE-based environmental feature extractor constitutes the first-tier network which is used to distil the critical information from BS-UE association top-view geographical images, where the image generation is finished in a modified DT (DRIVE) by using OSM of the given area. Meanwhile, the second tier is designed as a likelihood model which takes the outputs of the above extractor and real data features for training. The numerical results evaluated on real-world datasets show the gains of the proposed model in terms of prediction accuracy. The overall accuracy improvement is more than 20% and around 10% compared with the empirical and a simple MLP model respectively, and it can reach 38% and 16.4% improvement in the best validation case.

3.3.7 Outcome of the study

This simulation work is accepted by IEEE 33nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC) conference to be published as cited below,

* Li, P., Wang, X., Piechocki, R., Kapoor, S., Doufexi, A. and Parekh, A., 2022. “Variational Autoencoder Assisted Neural Network Likelihood RSRP Prediction Mode”l. In 2022 IEEE 33nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), 2022

3.4 Section 3 - References

[3.1] ETSI, “TR 138 901-V14.3.0-5G; Study on Channel Model for Frequencies From 0.5 to 100 GHz, 3GPP documentTR 38.901 version 14.3.0 Release 14,” May 2017. [Online]. Available: https://www.etsi.org/deliver/etsi_tr/138900_138999/138901/14.00.00_60/tr_138901v140000p.pdf

[3.2] “Guideline for scanner-based drive tests,” February 2009. [Online]. Available: https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/dl_application/application_notes/1sp56/1SP56_10e.pdf

[3.3] P. Li, J. Thomas, X. Wang, A. Khalil, A. Ahmad, R. Inacio, S. Kapoor, A. Parekh, A. Doufexi, A. Shojaeifard et al., “Rlops: Development life-cycle of reinforcement learning aided open ran,” arXiv preprint arXiv:2111.06978, 2021.

[3.4] E. Alimpertis, A. Markopoulou, C. Butts, and K. Psounis, “City-wide signal strength maps: Prediction with random forests,” in The World Wide Web Conference, 2019, pp. 2536–2542.

[3.5] T. S. Rappaport et al., Wireless communications: principles and practice. prentice hall PTR New Jersey, 1996, vol. 2.

[3.6] R. V. Akhpashev and A. V. Andreev, “Cost 231 hata adaptation model for urban conditions in lte networks,” in 17th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM), 2016, pp. 64–66.

[3.7] M. Hata, “Empirical formula for propagation loss in land mobile radio services,” IEEE Transactions on Vehicular Technology, vol. 29, no. 3, pp. 317–325, 1980.

[3.8] T. Sarkar, Z. Ji, K. Kim, A. Medouri, and M. Salazar-Palma, “A survey of various propagation models for mobile communication,” IEEE Antennas and Propagation Magazine, vol. 45, no. 3, pp. 51–82, 2003.

[3.9] R. Enami, D. Rajan, and J. Camp, “Raik: Regional analysis with geodata and crowdsourcing to infer key performance indicators,” in IEEE Wireless Communications and Networking Conference (WCNC), 2018, pp. 1–6.

[3.10] Y. Wang, M. Y. Liang, J. Hu, T. Song et al., “A new method for radio wave propagation prediction based on bp-neural network and path loss model,” in 12th International Conference on Knowledge and Smart Technology (KST). IEEE, 2020, pp. 41–46.

[3.11] F. Sohrabi and E. Kuehn, “Construction of the rsrp map using sparse mdt measurements by regression clustering,” in IEEE international conference on communications (ICC). IEEE, 2017, pp. 1–6.

[3.12] X. Zhang, X. Shu, B. Zhang, J. Ren, L. Zhou, and X. Chen, “Cellular network radio propagation modeling with deep convolutional neural networks,” in Proceedings of the 26th ACM SIGKDD International Conference on knowledge discovery & data mining, 2020, pp. 2378–2386.

[3.13] U. Masood, H. Farooq, and A. Imran, “A machine learning based 3d propagation model for intelligent future cellular networks,” in IEEE Global Communications Conference. IEEE, 2019, pp. 1–6.

[3.14] J. Thrane, D. Zibar, and H. L. Christiansen, “Model-aided deep learning method for path loss prediction in mobile communication systems at 2.6 GHz,” IEEE Access, vol. 8, pp. 7925–7936, 2020.

[3.15] J. A. Erkoyuncu, P. Butala, R. Roy et al., “Digital twins: Understanding the added value of integrated models for through-life engineering services,” Procedia Manufacturing, vol. 16, pp. 139–146, 2018.

[3.16] J. Thrane, B. Sliwa, C. Wietfeld, and H. L. Christiansen, “Deep learning-based signal strength prediction using geographical images and expert knowledge,” in IEEE Global Communications Conference.IEEE, 2020, pp. 1–6.

[3.17] Z. Yi, L. Zhiwen, H. Rong, W. Ji, X. Wenwu, and L. Shouyin, “Feature extraction in reference signal received power prediction based on convolution neural networks,” IEEE Communications Letters, vol. 25, no. 6, pp. 1751–1755, 2021.

[3.18] I. Mavromatis, R. J. Piechocki, M. Sooriyabandara, and A. Parekh, “Drive: A digital network oracle for cooperative intelligent transportation systems,” in IEEE Symposium on Computers and Communications (ISCC). IEEE, 2020, pp. 1–7.

[3.19] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” arXiv preprint arXiv:1312.6114, 2013.

[3.20] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.

[3.21] S. Odaibo, “Tutorial: Deriving the standard variational autoencoder (vae) loss function,” arXiv preprint arXiv:1907.08956, 2019.

4 Hierarchical and multiagent Reinforcement Learning

4.1 Machine Learning Overview

Machine learning (ML) is a field of artificial intelligence concerned with the development of algorithms which converge to an optimal solution and improve the system performance without human intervention [4.1] [4.2]. The ML paradigm is broadly classified into three different categories: supervised; unsupervised; and reinforcement learning. In the presented work, the reinforcement learning technique is employed to understand the influence of learning through continuous interactions in a dynamic environment on network performance.

4.1.1 Introduction to Reinforcement Learning

Reinforcement learning (RL) is a branch of artificial intelligence, a class of machine learning, that employs a reward and punishment policy to enable an agent to learn a solution to a decision problem by interacting with its environment purely through trial-and-error such that the overall reward value is maximized. Unlike other learning techniques, RL focuses on a goal-directed learning, therefore, depending on the consequences of the learnt action a reward is awarded to the learning agent in case of successful attempts else it is punished [4.2]. The key merit of RL is its ability to learn a solution without any prior knowledge of the environment or the reward function. However, one of the challenges in RL is the trade-off between exploration and exploitation. A learning agent aims to maximize the reward by effectively employing an action that has proven promising in the past. But, to discover such an action, the learning agent must try each available action. Therefore, the task of a learning agent is to explore all the available actions to learn and subsequently exploit the most efficient action in the future. Figure 35 shows a basic diagram of the RL.

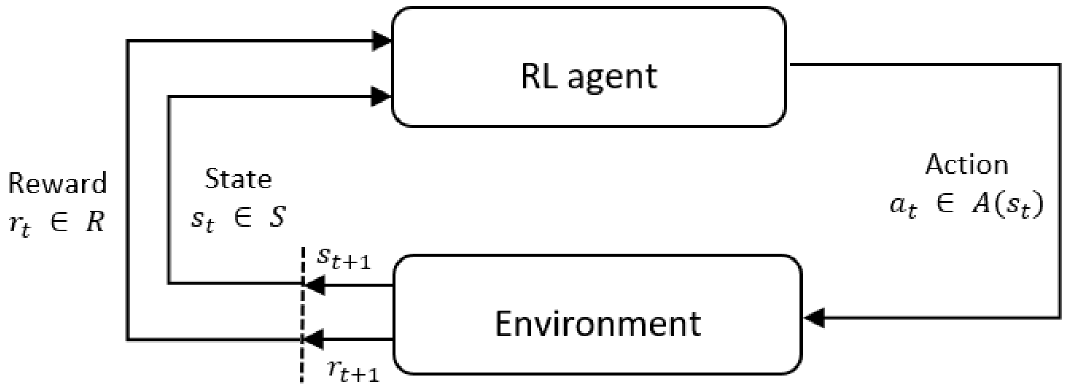


Figure 35 - Fundamental block diagram of reinforcement learning

The most widely used reinforcement learning techniques in artificial intelligence domain as well as in wireless networks are Q-learning and state-action-reward-state-action (SARSA).

Q-learning (QL) proposed by Watkin is one of the most popular RL techniques in widespread use in wireless and artificial intelligence domain. Here, a centralized array known as the Q-table is maintained. The values in the Q-table are called the Q-values and are initialized to unity, allowing the agent to start to learn with an equal choice among all available actions. The agent uses the learning policy to learn an action, whereas the update rule is employed to update the Q-value associated with each action. The Q-table, therefore, presents an analysis of the choice of behaviour of all the individual agents, while the Q-value represents the expected cumulative reward the agent receives by learning an action. The learning agent in Q-learning uses a learning policy, such the *ε-greedy policy*, to learn an action with maximum value. The approach states that an exploratory random action is picked with probability *ε* otherwise a good policy action (greedy) is selected with probability *1- ε*.
The greedy action is selected using

$$\text{action} = \arg\max\left(Q\left(s, a\right)\right)$$

Subsequently, a learning agent recursively updates the Q-value of each learnt action using the following update equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q'(s',a') - Q(s,a)]$$

where, *Q (s, a)* corresponds to the Q-value of the current state-action pair, α is the learning rate, γ is the discount factor, *r* is the reward received at each instance of time, and *Q'(s', a')* is the Q-value of the previous state-action pair. The learning rate parameter, $\alpha \in [0, 1]$ controls the convergence rate. The discount factor $\gamma \in [0, 1]$ controls the importance of future rewards with respect to immediate rewards. *r* represents the reward value that is awarded for the learnt action, and *Q'(s', a')* is the maximum Q-value among the available actions in the next state (s').

4.2 Federated Learning for Traffic Steering in Macro Cell networks

4.2.1 Introduction

The Traffic steering is a hard problem which becomes more difficult the larger the number of participants. Thus, heuristic methods often fail to fulfil such high QoS demand. Therefore, machine learning-based algorithms are a more promising way to solve this problem. Machine learning (ML) based resource allocation for wireless networks have been studied for over a decade [4.3]. Yet new problems are still emerging, with new demands and new radio access technologies (RAT) such as effectively managing 5G new radio (NR) features. ML-based algorithms for resource allocation problems often use reinforcement learning (RL)-based methods [4.4]. These methods use a simulated environment to generate data to train RL agents by using these data. While there are number of RL formulations, we used a deep reinforcement learning (DRL) method for decision making in simulated environment.

RL agents that have stochastic environments require an adaptation phase to achieve optimal rewards. In order to increase the convergence speed of model adaptation to the environment, a collaborative learning algorithm, namely federated meta-learning (FML) [4.5] is proposed to orchestrate the network in the O-RAN environment.

There are several reasons to choose the FL framework for the O-RAN ecosystem. One of the main reasons is to generalise across the distribution of environments for RL agents [4.6]. Cellular networks contain highly dynamic and unique environments. Even well-trained RL agents may fail to adapt to the environment after deployment. If RIC management cannot deal with a quickly changing environment, it can cause significant QoE issues for users. Another reason is some areas may have different priorities than others in aspects of service types; for example, some applications may demand higher throughput, and some may need lower latency while communicating. We defined several QoS metrics such as throughput and latency to generalize the problem within the FML framework.

Our motivation for using meta-learning approach for DRL algorithm is to enhance the adaptation process of RL agents.

The reason for focusing on the adaptation process is that wireless communication in RAN is highly dynamic. Moreover, service demands are application-dependent (as mentioned be- fore) and, obtaining optimal solution faster plays a crucial role in intelligent resource management applications. Hence, we aim to train a DQN model that enables RL agent adapt to a new task i.e., latency, throughput, or caching rate, can be quick.

We propose the form of FML which uses the reptile algorithm [4.7] for meta-learning.

The major contributions of the work are as follows:

- A RAT allocation environment which enables RL agents to train their DQN models for steering traffic between RATs to provide service to vehicles.
- Various QoS performance metrics are measured in the environment. These QoS metrics are defined as unique tasks for the meta learning algorithm.

- A federated meta-learning framework is designed for higher convergence speeds to unseen tasks and environments. We distributed learning algorithms in the frame- work and analysed the results.
- We evaluated how the rule-based approach and learning- based approaches are performing in our RAT allocation environment. Results show that the proposed FML algorithm performs the best among other approaches.
- To the best of our knowledge, there is no paper that simulates a distributed setup for traffic steering which is supported by O-RAN architecture.

4.2.2 AI-based Traffic Steering

Federated learning (FL) [4.8] paradigm aims to perform training directly on edge devices with their own local data and employs a communication efficient model management scheme. While collected data is kept local, communication between the global server and edge devices contains only model gradients [4.8]. Therefore, FL is a communication and computation-efficient way of distributed learning algorithms. The server merges these updates to form a better more generic model for all environments. Model updates at local RL agents represent information about local environments. Thus, the global model collects representative information about the deployed environment without getting any state data. This feature prevents both constant data flow from deployed units and keeps private data local such as user equipment's (UE) locations, routines, data usage, etc. After aggregating model updates at the server, the server forms a global model as a generic model for all agents.

Federated reinforcement learning (FRL) enables RL agents to generalise environment by using collaborative scheme. Liu et al. [4.6] used FL framework to extend RL agents' experience so that they can effectively use prior knowledge. Their objective is to improve adaptation speed to a new environment by using this knowledge. Our proposed method uses various environments for the same motivation as well. However, in addition to FL framework, we also employ meta-learning to enable RL agents to adapt faster to new QoS tasks as well as environments.

The FML algorithm is used to increase converging speeds to unseen tasks [4.9]. This feature enables RL agents or any other DNN-based learning algorithms to adapt to new tasks faster. Yue et al. [4.5] used FML approach to maximize the theoretical lower bound of global loss reduction in each round to accelerate the convergence. Unlike our approach, they added user selection mechanism according to contributions of local models.

Besides RL-based methods, there is also recurrent neural network (RNN) based FML methods [4.10]. Zhang et al. used different cities as meta-tasks to train an RNN model to predict network traffic. While they used each city as task, we used each QoS metric in the network as a meta-task in this simulation. In both way, tasks are not completely independent. Network traffic for cities usually represents a seasonal behaviour that changes accord the time of the day. Therefore, traffic demand changes at similar times but in different volumes. In our use case, we used different QoS metrics as tasks and they are indirectly dependent as well e.g., higher throughput will lead to transmit data in less time and it will decrease the latency.

There are many ML-based resource allocation papers in the literature [4.11]. However, there are only a few studies that use ML to provide a solution to traffic steering use case since it is relatively new problem. Adamczyk et al. [4.12] used an artificial neural network trained with the SARSA algorithm for traffic control in RANs. For traffic steering use case, they allocated resources among the users according to base station load. They defined different user profiles according to their data rate demands. In our study, we define these demands as meta-tasks. Thus, after the deployment of an RL agent, it adapts to a new demand profile more rapidly.

In this use case, we consider a connected urban system model with multiple users moving along roads. A multi-Radio Access Technology (multi-RAT) Base Station (BS) is set in this area, providing network coverage to users. Each user has download service requests to be satisfied by the BS, such as road condition information downloading, or other internet services like web browsing or video streaming. Note that each request has its own lifetime. These requests are made by the users and stored at the BS. A scheduler is located at the BS, serving downloading requests of all users in an efficient manner.

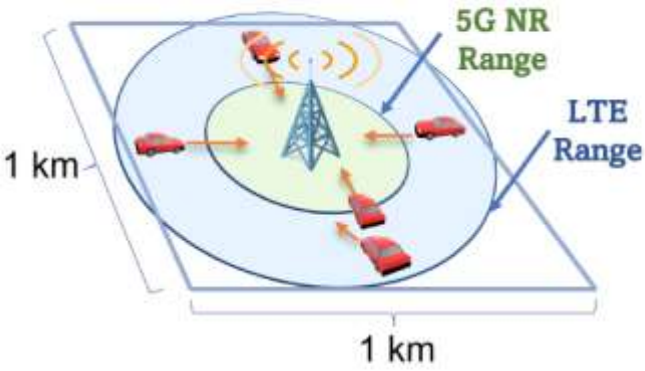


Figure 36 - Traffic steering simulation environment map.

4.2.2.1 Urban User Model

Consider an urban area shown in Figure 36, with a 1 km by 1 km map. The multi-RAT BS is located in the middle to provide better coverage. Users can be approaching or leaving the BS. Note that there can be a T-junction, crossroad, roundabout, etc. in the centre. In this simulation, the number of vehicles is always 5. When one vehicle leaves this area, another vehicle will be automatically generated to maintain a constant vehicle number.

4.2.2.2 I2V Connectivity Model

We assume that the BS provides connectivity across R RATs. In the case of I2V connectivity, for each RAT, we define the downlink data rate $r_{i,j}$ achieved by the BS to vehicle i over the RAT j as follows:

$$r_{i,j} = \sigma_j \log_2 (1 + \mathrm{SINR}_{i,j}) \quad (7)$$

Where σ_j is the bandwidth of the RAT j and SINR_i is the Signal-to-Noise and Interference Ratio (SINR) associated with the downlink transmissions originating to vehicle i over RAT j . In particular, we define $\mathrm{SINR}_{i,j}$ as follows:

$$\mathrm{SINR}_{i,j} = \frac{G_j P_j h_j \ell^{(j)}(d_i)}{W_j + I_j} \quad (8)$$

Where:

- G_j signifies the overall antenna gain
- P_j is the transmission power for transmitting over RAT j
- $\ell^{(j)}(d_i)$ expresses the path-loss at a distance d_i (between the BS and vehicle i) and it is defined as $C_j d_i$^$(-\alpha_j)$ – C_j and α_j are constants associated with RAT j
- h_j is a random variable modelling the fast fading component and it depends on the RAT in use.
- W_j represents the white thermal noise power. It can be seen as a constant that depends on the RAT in use.
- I_j is the interference power. Here we assume $I_j = 0$, thus $\mathrm{SINR}_{i,j} = \mathrm{SNR}_{i,j}$.

Considering the i-th vehicle v_i , we say that through every request, it requests a 'job' J_i , which consists T_i data frames, namely, $J_i = \{v_{i,1}, \ldots, v_{i,T_i}\}$) from the BS. Each data frame has an identical size, while each job is associated with a lifetime, that is, a downloading deadline. If the job has not been downloaded before its deadline, it will be discarded, thus the corresponding request is not satisfied. During transmission, if any data frame is not successfully downloaded due to any possible reasons, it would be regenerated and transmitted again.

4.2.2.3 System Goal and Tasks

The goal of this system is to design a scheduler to dynamically meet vehicle downloading requests by using multiple RATs in an efficient manner.

1) Caching rate: The caching rate is the ratio of successfully transmitted bytes/packets over all requests. We define caching rate as follows:

$$CR = \frac{\text{Completed jobs}}{\text{Total requests}} \tag{9}$$

2) Latency: Latency is calculated as follows:

$$\Delta t = \frac{t_c}{t_d}, \tag{10}$$

where tc is the completion time and td is the total deadline time. Since there are two different job sizes, creating proportional latency values is fairer than calculating the remaining time in seconds.

3) Throughput: Throughput metric is calculated as follows:

$$T = \frac{T_c + T_l}{t}, \tag{11}$$

where Tc, Tl are successfully transmitted bytes in completed jobs and lost jobs respectively. t is the time duration in the simulation step.

4) Proportional Fairness:

Fairness is a comprehensive term. Fairness in network can be based on different metrics such as latency, throughput, availability etc. To simplify calculations in simulation we used proportional fairness in terms of throughput distribution among users [4.13]. It is calculated as follows,

$$F(x) = \sum_{v} \log(x_v), \tag{12}$$

where xv is the flow assigned to the demand of vehicle v.

The data scheduler aims to find a policy, deciding which data frame to be sent through which RAT at each time step, providing good data downloading service upon request. We assume that each vehicle is equipped with a Global Positioning System (GPS) service and the vehicle location information are accessible by BS real-time.

There are two RATs available at the BS. One is the 4G (LTE), and the other is 5G NR. The communication range of LTE covers the whole simulation area, while 5G NR only covers part of the area. Simulation parameters can be found in Table 2.

This problem can be modelled as a Markov Decision Process (MDP), with the finite set of states' S, the finite set of actions A, the state transition probability P and the reward R.

Parameter		Value
Number of vehicles		5
Vehicle speed		8 m/s
Max communication range	LTE	922 m
	5G NR	200 m
Buffer size at the BS		5
Job size	Type A	1 MB
	Type B	10 MB
Job deadline	Type A	0.1 s
	Type B	1 s
Time interval between actions		1 ms

Table 2 - Environment parameters

States: At time t, the state St can be represented as

$$S^t = [S^t_U, S^t_{BS}]. \tag{13}$$

Here $S^t{}_U$ is the user(vehicle) states and $S^t\ BS$ is the BS status at time t. For simplicity, we do not specify the time t unless necessary in this work. SU can be written as SU = [(x1, y1, vx,1, vy,1), ..., (xn, yn, vx,n, vy,n)], where xn, yn, vx,n, vy,n are the geographical position and velocity of user i, respectively. n is the number of users.

For the BS status, it contains the job buffer, status of RATs, and the link status of the 'BS - vehicle' connections, represented as sb, sR and sc, respectively.

$$S_{BS} = [s_b, s_R, s_c] \tag{14}$$

With length of l the buffer status can be written as

$$s_b = [(b_1, t_1, D_1), (b_2, t_2, D_2), ..., (b_l, t_l, D_l)] \tag{15}$$

where (pi , ui , ti) are the number of packets left for current job i, the vehicle which requested job i, and the time left for job i before it would be discarded, respectively. To show the RATs status, we use binary values to describe their availability:

$$s_R = [a, b], \text{s.t.} a \in [0, 1], b \in [0, 1] \tag{16}$$

If there is no packet being downloaded through the first RAT, LTE, then a = 1, meaning that the first RAT is currently available, and vice versa. b represents the status of the second RAT, 5G NR.

As for the 'BS — vehicle' connection status, we show the potential data rate each vehicle could get through two RATs at its current position. sc can be written as:

$$s_c = [dr_{(1,1)}, dr_{(2,1)}, dr_{(1,2)}, dr_{(2,2)}, ..., dr_{(1,n)}, dr_{(2,n)}] \tag{17}$$

Here dr (m,i) means the data rate vehicle which i could obtain through RAT m.

Actions: The action space for the BS is defined as A = ({0, 0}, {0, 1}, ..., {T, j }, ∅), where {T, j } means to download a packet of the Tth job from the buffer to the required vehicle through RAT j , with ∅ meaning no action. Note that if more than one RAT is available, then the BS could choose the maximum of two actions at one time step.

Reward: The goal is to schedule the data downloading process so that the BS could satisfy vehicle requests in an efficient manner.

1) (R1) RAT usage reward: for every unused RAT at each time step, the agent receives a reward of -1.

2) (R2) Lost job reward: for every lost job, the agent receives a reward of -100.

3) (R3) Successful job reward: for every successfully finished job, the agent receives a reward of +10.

4) (R4) Latency: for every successfully finished job, the agent receives a reward of +10(1 − Δt).

5) (R5) Throughput: for all jobs, total bytes transmitted are calculated and the agent receives a reward of +0.1T.

6) (R6) Proportional fairness: for all vehicles in the environment, the agent receives a reward of +$F(x)$.

4.2.2.4 Heuristic action selection

The RAT allocation simulation environment is a unique environment. Therefore, to compare results with the proposed approach, we designed a heuristic action selection algorithm as a baseline. The heuristic algorithm utilizes the same state information as an RL agent to decide on its actions, and then will try to provide an optimal solution according to the state. This heuristic is given as

Algorithm 1: Heuristic action selection

```
Require: The setting of different indoor scenarios.
Require: s
Destinations ← s(a : b)
UE_pos ← s(c : d)
RAT availability ← [5G, LTE]
if No RAT available then
    Action ← Do nothing
    Break
end
for UEs in Buffer do
    if UE_x in 5GNR coverage then
        Action ← [UE_x, 5G]
        Break
        else
            Action ← [UE_x, LTE]
        end
    end
end
Return: Action
```

This heuristic is designed to choose 5G when UEx is one of the destinations UEs in the buffer. Since the 5G NR service has higher data rates, it can complete jobs faster and reduce the loss of jobs. However, in our simulation setup, we defined a 5G NR data rate as lower than LTE when a UE gets closer to the border of the 5G NR coverage area. We expected our RL agents to learn this pattern and manage resources according.

4.2.3 Federated Meta-Learning For O-Ran Traffic Steering

In this simulation, we use FML algorithm [4.8] to provide both a hierarchical and fast-adapting framework. FL low communication overheads permits deployment at end devices. We consider deployment at either RICs or E2 nodes [4.14] as a resource management unit in the hierarchy. When the FML algorithm is deployed in the field, it will require data with the least latency. Hence, the action decisions of the RL agents will not expire for the collected state information from the environment. We assume state information of simulation is collected close to real-time. Therefore, any latency caused by transmitting data from RU to RIC is ignored in this work. The proposed FML algorithm for traffic steering in O-RAN is given in Algorithm 2.

Algorithm 2: Federated meta-learning for TS simulation

Require: N, K, E, I, $\mathrm{Num}_{\mathrm{vehicles}}$, $\mathrm{Size}_{\mathrm{Buffer}}$, α, γ, β
Initialize: K environments, Vehicles, Base station, Buffer objects
for n *in* N **do**
  for k *in* K **do**
    $\theta_k \leftarrow \theta_{\mathrm{Global}}$
    $\mathrm{Buffer}_k \leftarrow Randomdemands$
    $i \leftarrow 0$
    for e *in* E **do**
      $s([V_{\mathrm{pos}}, V_{\mathrm{vel}}, \mathrm{Buffer}, A_{RAT}]) \leftarrow observeenv_{n,k}(t)$
      $a \leftarrow \epsilon_{\mathrm{greedy}}, \theta_{n,k}(s)$
      $R \leftarrow step(env_{n,k}(t), a, T_i)$
      Store transition (s',a',R,s,a) in buffer D
      Sample random minibatch of transitions from D
      if *Episode terminates* **then**
        $y \leftarrow R$
      end
      else
        $y \leftarrow [R + \gamma \max_{a'} Q(s', a'; \theta_{n,k})]$
      end
      Perform gradient descent step on $(y - Q(s, a; \theta_{n,k}))^2$
      $t \leftarrow t + \Delta t$
      $i \leftarrow i + 1$
      if $i \geq I$ **then**
        $\theta'_{n,k,\mathrm{meta}} \leftarrow \theta_{n,k} - \beta \frac{1}{I} \sum_{i=0}^{I} (\theta_{n,k} - \theta_{n,k,i})$
        $\theta_{n,k} \leftarrow \theta'_{n,k,\mathrm{meta}}$
        $i \leftarrow 0$
      end
    end
    Upload $\theta_{n,k}$
  end
  $\theta_{\mathrm{Global}} \leftarrow \frac{1}{K} \sum_{k=0}^{K} \theta_{n,k}$
end

In this algorithm, N is the number of FL aggregation cycles, and K is the number of parallel environments. Since each environment is managed by a single RL agent K also equals the number of RL agents. E is the number of training cycles for each RL agent, and I is the number of training tasks. These tasks are used to train models in meta-learning algorithm. $\mathrm{Num}_{\mathrm{vehicles}}$ is the number of vehicles in each environment, and Size Buffer is the buffer capacity for each environment. After registering these data, the simulation initializes K unique environments for each RL agent. These environments have different vehicle starting points in the simulation map as shown in Figure 36. Before beginning the training all buffers are filled with Type A and B jobs randomly as given in Table I. Each RL agent trains its DQN model in their own environment. DQN algorithm aims to find optimal policy to obtain optimal return according to state and action. DQN algorithm estimates the action-value function by using the Bellman equation as in equation (18) [4.2].

$$Q^{\pi}(s, a) = \mathbb{E}_{s' \sim \mathcal{S}}\left[R + \gamma \max_{a'} Q^{*}(s', a') \middle| s, a\right]. \quad (18)$$

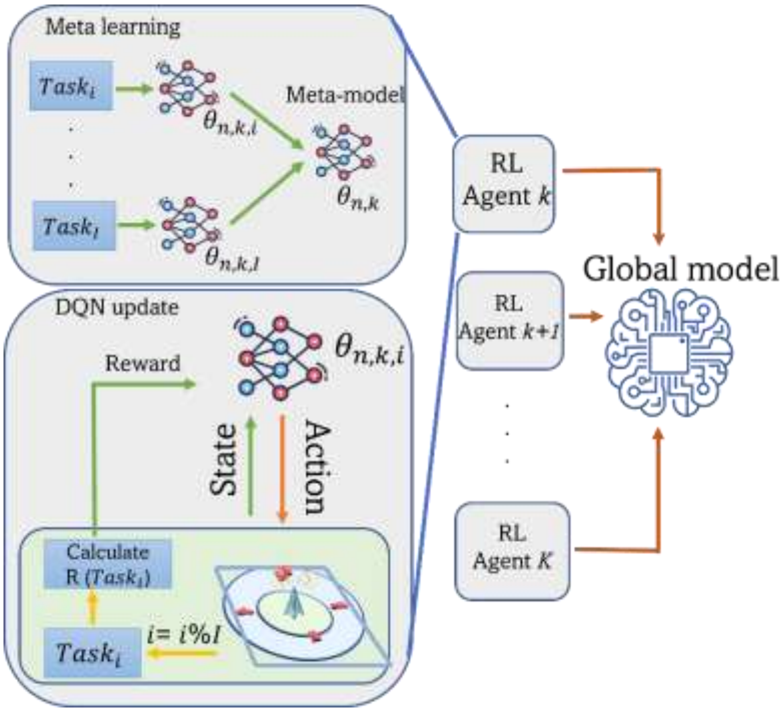


Figure 37 - Federated meta-learning framework set-up for traffic steering.

Here the RL agent tries to find the best action a' for corresponding state s' to find optimal policy. To prevent unstable updates, gradients are limited by a discount factor γ. After E cycles they upload their model parameters to the server. Then server aggregates these model updates with the FedAvg algorithm [4.18] to form a global model as given in equation (19). After aggregation for K agents is completed, the server broadcasts global model θ_{Global} back to the RL agents. The RL agents use this pre-trained global model in their environment in the next FL aggregation cycle.

$$\theta_{\text{Global}} = \frac{1}{K}\sum_{k=0}^{K}\theta_{n,k} \tag{19}$$

4.2.3.1 Meta-Learning Tasks

The goal of meta-learning is to train a model which can rapidly adapt to a new task using only a few training steps. To achieve this, RL agents train the model during a meta-learning phase on a set of tasks, such that the trained model can quickly adapt to new tasks using only a small number of adaptation steps [4.15]. Since RL agents are deployed in a simulation environment where UE trajectories are likely to be unique, RL agents will try to adapt to this unseen environment. Moreover, as mentioned before besides the UE trajectories, the demands of UEs can differ in each environment. Hence, we used four different tasks to train RL agents in the meta-learning phase and observe them adapt to the fifth one. After every four rounds, RL agents update their DQN models according to equation (20),

$$\theta'_{n,k,\text{meta}} = \theta_{n,k} - \beta\frac{1}{I}\sum_{i=0}^{I}(\theta_{n,k} - \theta_{n,k,i}) \tag{20}$$

Here β is the meta-step size, which scales model updates, and I is the number of tasks to train in a meta-learning manner. Scaling updates prevents DQN model from fully converging to a specific task. Instead of a specific task, it is expected that the DQN model converges to a point where it can adapt to an unseen task as quickly as possible. Six different reward functions are described in meta-tasks section for meta-learning methods. Five unique tasks are defined by using these reward functions; the tasks are listed as follows:

1) Task 1 is the most comprehensive reward that an RL agent can train in this environment, which is calculated as $R1 + R2 + R3 + R4 + R5 + R6$

2) Task 2 is reward based on proportional fairness, calculated as $R1 + R6$

3) Task 3 is latency-prioritized reward, calculated as, $R1+R4$

4) Task 4 is throughput-based reward, calculated as $R1 + R5$

5) Task 5 is reward based on caching rate, calculated as $R1 + R2 + R3$

Here R is the reward function described previously. In the Reptile algorithm and the FML method, RL agents use the first four tasks for training and all methods try to adapt the 5th task in the adaptation phase, while a single DRL agent uses only Task 1 for training and tries to adapt Task 5.

4.2.3.2 O-RAN and Traffic Steering

There are several resource types in the O-RAN structure and the demands of UEs can change the scheme of resource management. However, the action space for RL agents grows exponentially with number of dimensions of resources to be managed. Since most of the RL algorithms depend on explore-and-exploit methods, using multiple RL agents collaboratively is likely to enhance exploration, and help RL agents converge to better rewards faster. Hence, this is one of the major reasons for proposing the FML framework for traffic steering in O-RAN. Simulation results show even a single resource type such as the RAT allocation problem can be handled better with collaborative learning of multiple RL agents.

4.2.4 Simulation Results and Discussions

A. Simulation parameters

In traffic steering simulations, we observed several parameters in the simulation to see how FL performs under different conditions. In most of the simulation runs, FML performed significantly better than other methods, but on some occasions heuristic and FML performance was almost equal. We ran a simulation with 20 different parameter combinations, and an average performance comparison is achieved with the parameters as, the number of FL aggregating cycles is $N = 10$, the number of training tasks is $I = 4$, number of created environments (and RL agents) in FL network is $K = 5$, number of episodes before aggregation is $E = 100$ and time interval between environment steps is $\Delta t = 1$ms. Note that, to have a fair comparison, single DRL and Reptile-based DRL agents are trained equally as much as RL agents in the FML framework. In this case, it is $N * E$.

B. Caching performance results

In the simulation, we tracked lost packets and lost bytes in each lost packet to calculate a caching rate performance indicator. Each method has been run for 10 validation runs in an unseen environment (unique environments in terms of UE trajectory) in the training phase. After 10 runs for each parameter combination, we averaged the caching results to get the final score for each approach which is given in Table 3.

Method	Packets	Bytes
Single RL	81%	90%
Heuristic	78%	88%
Reptile	86%	92%
Proposed FML	**89%**	**95%**

Table 3 - Caching-rate performance comparison

As shown in Table 3, the proposed FML approach achieves the highest caching rate performance amongst the compared methods in terms of packets and bytes. We calculated these results as the ratio of successfully transmitted bytes/packets over total requests.

C. Adaptation performance

Simulation results show that FML method improves adaptation performance from the very first training episodes. As a quantitative comparison, RL-based algorithms try to reach heuristic method performance as soon as possible. According to Figure 38, the FML algorithm achieves heuristic equivalent performance the quickest among all methods. Zero-shot adaptation performances of different methods are compared in Table 4, where HEP is heuristic equivalent performance.

Method	0-Shot (mean)	0-Shot (std)	Episode of HEP
Single RL	51%	13%	13
Reptile	60%	12%	9
Proposed FML	**72%**	**10%**	**3**

Table 4 - Adaptation performance comparison

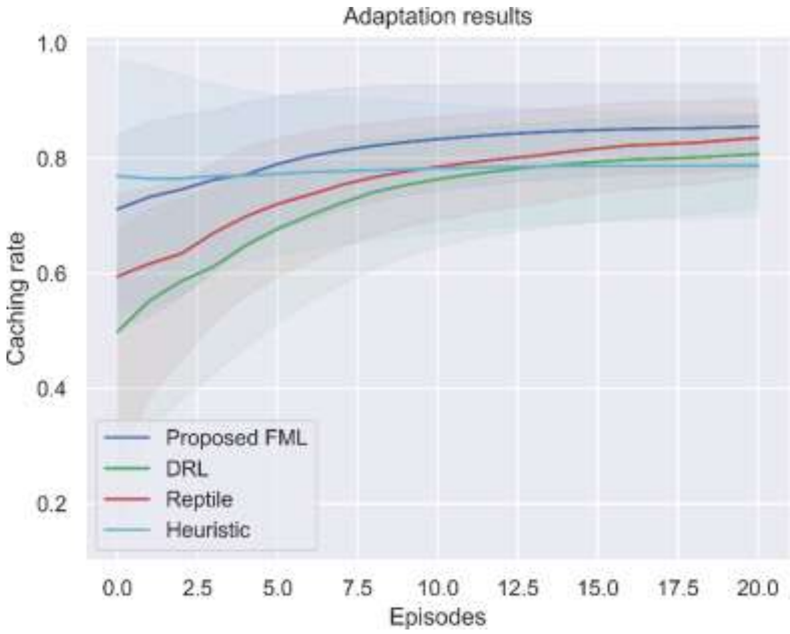


Figure 38 - Adaptation performance comparison to unseen task and environment.

4.2.5 Discussion

As shown in Table 3, FML has demonstrated to be a better solution for designed traffic steering simulation. Even though we observed similar training performances for a single RL agent and FML agent, FML performed better in an unseen environment. The FL framework takes advantage of collecting information from various environments, and so it becomes easier to adapt to a new environment. There are cases where a single learner performs better than the global model because of the issue of collected not independent and identically distributed (non-iid) data. Nevertheless, there are other solutions to prevent performance deterioration at the global model [4.16]. In future work, we will add new resource types to this environment. Traffic steering is a comprehensive use case, since the 5G NR standard allows service providers to collect various communication-related data from UEs, it is more likely to have AI/ML-based solutions for such problems [4.11].

4.2.6 Conclusion

In this work, we have focused on the generalization of different tasks and environments by using the FML framework. As a use case, we used traffic steering in O-RAN architecture. We designed a traffic steering environment to investigate how DRL-based learning algorithms perform in such an environment. While unique environments are created for every RL agent in a stochastic way, RL agents try to manage RIC and allocate RAT services among UEs in the environment. We have analysed the convergence speed of the DRL algorithm that uses a single task and single environment. Another DRL algorithm that uses multiple tasks in the Reptile algorithm and single environment and proposed FML framework that uses multiple tasks and multiple environments to train RL agents. Simulation results confirm the effectiveness of our proposed algorithm. Future work can investigate how the FML framework deals with managing multiple radio resources on RICs.

4.2.7 Outcome of the study

This simulation work is accepted by IEEE 96th Vehicular Technology Conference (VTC2022-Fall) to be published as cited below,

* Erdol, H., Wang, X., Li, P., Thomas, J.D., Piechocki, R.J., Oikonomou, G., Inacio, R., Ahmad, A., Briggs, K. and Kapoor, S., 2022. “Federated Meta-Learning for Traffic Steering in O-RAN”. In 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall), 2022.

4.3 Indoor Small-Cell Power Optimisation

4.3.1 Indoor pathloss models

The AIMM simulator indoor pathloss module represents walls, floors, and ceilings by planar panels made up of triangular sections. Each panel can have a specified signal absorption (wall loss), typically in the range 10 to 15 dB. Once these panels are instantiated, every pathloss calculation in the simulator will involve computing the number of panels intersected by a line joining the cell location to the UE location. This wall loss is added to either a free-space pathloss computed from the distance, or alternatively the 3GPP InH indoor propagation model can be used.[1] This is provided as an AIMM Sim function.

We emphasize that this is not a ray-tracing code; experience shows that ray-tracing would be much too slow for the ML application envisaged, and would probably add little accuracy.

4.3.2 Indoor mobility models

Another design issue concerns appropriate UE mobility models in these scenarios. Though it is not realistic, it is probably nevertheless acceptable in the simpler simulations to have user walking right through walls. A model we used frequently to train ML models was the one we called “wave”. Here the users start with a uniform distribution over the building, but gradually all move to one end of the building, and then back to uniform. This cycle repeats indefinitely. The intention is to provide the ML agent with experience on extremes of user distributions, from completely uniform to highly non-uniform.

For cases where this is not acceptable, an experimental “billiard” model was implemented. In this model, users are given an initial position and velocity, and then make soft bounces off walls and partitions. The softness of the bounce is a controllable parameter.

As examples, Figure 39 shows a typical three-room building with a billiard UE path (defined below), and Figure 40 shows a larger open-plan office, with the UE path bouncing off the exterior walls but not the internal partitions. These diagrams are automatically created by plotting functions in the simulator code. Cell locations are the large red dots.

[1] Specification # 36.873 (3gpp.org)

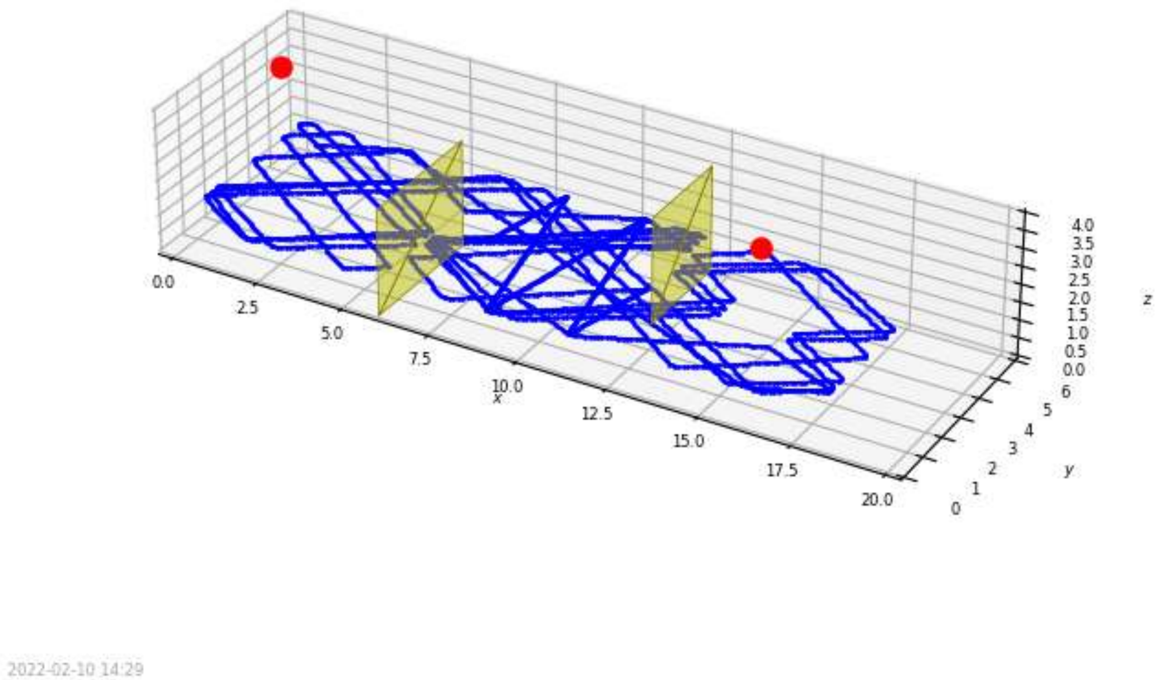


Figure 39 - Example three-room building.

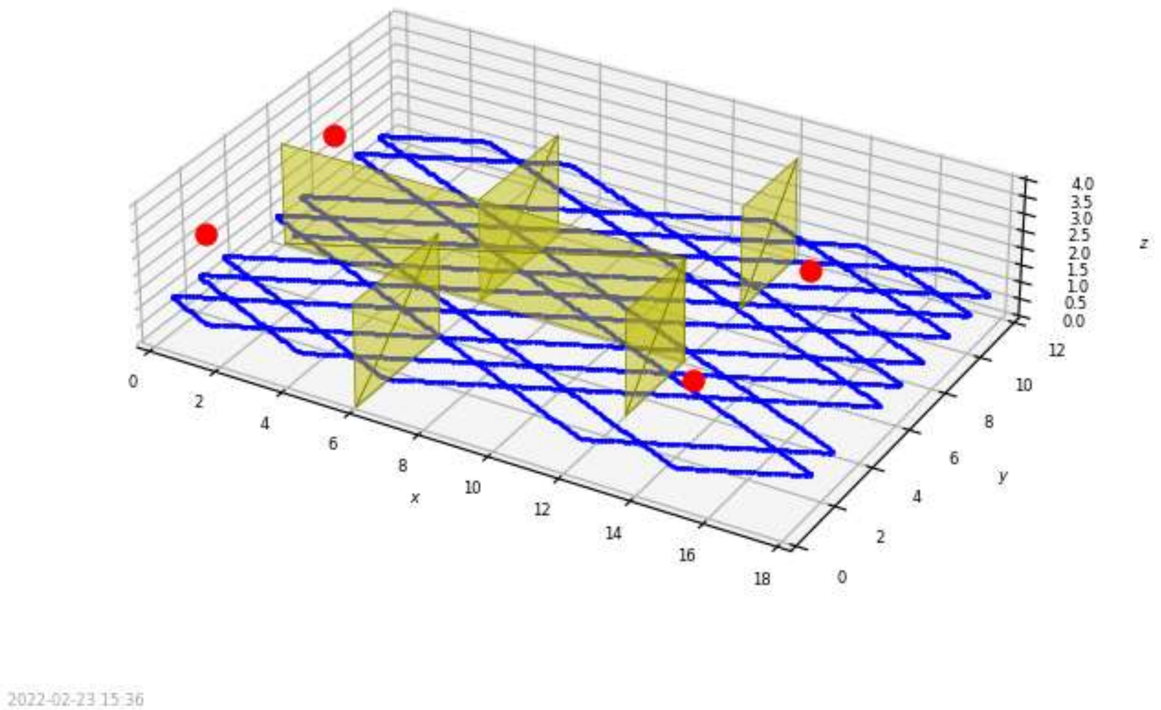


Figure 40 - Example open-plan office with partitions.

4.3.3 The indoor small-cell power-control use-case

The aim is to have ML agent learn to control the transmit power of two or more indoor small cells, in such a way as to react to changes in UE location. The hope is that by suitably setting the powers, the impact of interference on throughput will be minimized.

A very important question concerns the choice of objective function, or (in ML applications), the reward function. It might be thought that average (or total) downlink throughput across all UEs would be a suitable objective function to be maximized. However, experience show that this can be bad because a high average can be achieved by giving all (or most) resources to a single UEs which happens to be close to a cell. This is an unfair allocation.

A better possibility is to try to maximize the minimum throughput, so that the worst-served UE gets the best possible service. But this can also be a poor objective, as a large proportion of system resources may be given to a UE which is in such a bad location that it will never get a good service. Therefore, we have used a compromise: we compute the distribution of throughput across all UEs, and then estimate a lower quantile, such as the 10% or 25% quantile. This is then the statistic which we try to maximize. It does of course mean that any UE below the chosen quantile will have no guaranteed level of service, and it also means that the average throughput is not controlled directly and has no guarantee. The essence of the problem is that we are trying to control a whole distribution with a small number of parameters (perhaps only one). From that point of view, there is no perfect solution.

4.3.4 Results

Our main aim is to determine whether reinforcement learning (RL) performs better than a simple heuristic (SH). What is called here a “simple heuristic” is in fact a very simple, but exact, algorithm, which simply tries all power settings and picks the one which maximizes the quantile mentioned above. It is, in effect, an exhaustive search.

The scenario has four cells in six rooms, as shown in Figure 40. The users (UEs) move according to a wave model, meaning that they start with a uniform distribution, and gradually concentrate at the left-hand wall. They then return to the original uniform distribution and repeat this pattern indefinitely. Two complete cycles can be seen in the plots.

The four plots below compare RL against SH for indoor propagation, with line-of-sight (LOS, no walls), and non-line-of-sight (NLOS, with walls). In all cases the most important statistic is the second light blue curve in the uppermost plot. This is the 25% quantile of throughput, and we want to maximize this. The cases, with a summary of performance, are:

1. SH, NLOS. The quantile averages at about 1.5Mb/s.
2. SH, LOS. The quantile averages at about 0.7Mb/s. It is lower than the previous case, as the absence of walls increases interference.
3. RL, NLOS. The quantile averages at about 1.5Mb/s.
4. RL, LOS. The quantile averages at about 0.7Mb/s. It is lower than the previous case, as the absence of walls increases interference.

The conclusions are very simple - reinforcement learning performs very similarly as the simple heuristic. Since the simple heuristic in fact performs optimally, then so does RL. But RL has much higher complexity, both implementation and computation. We conclude that RL has nothing to offer for this use case.

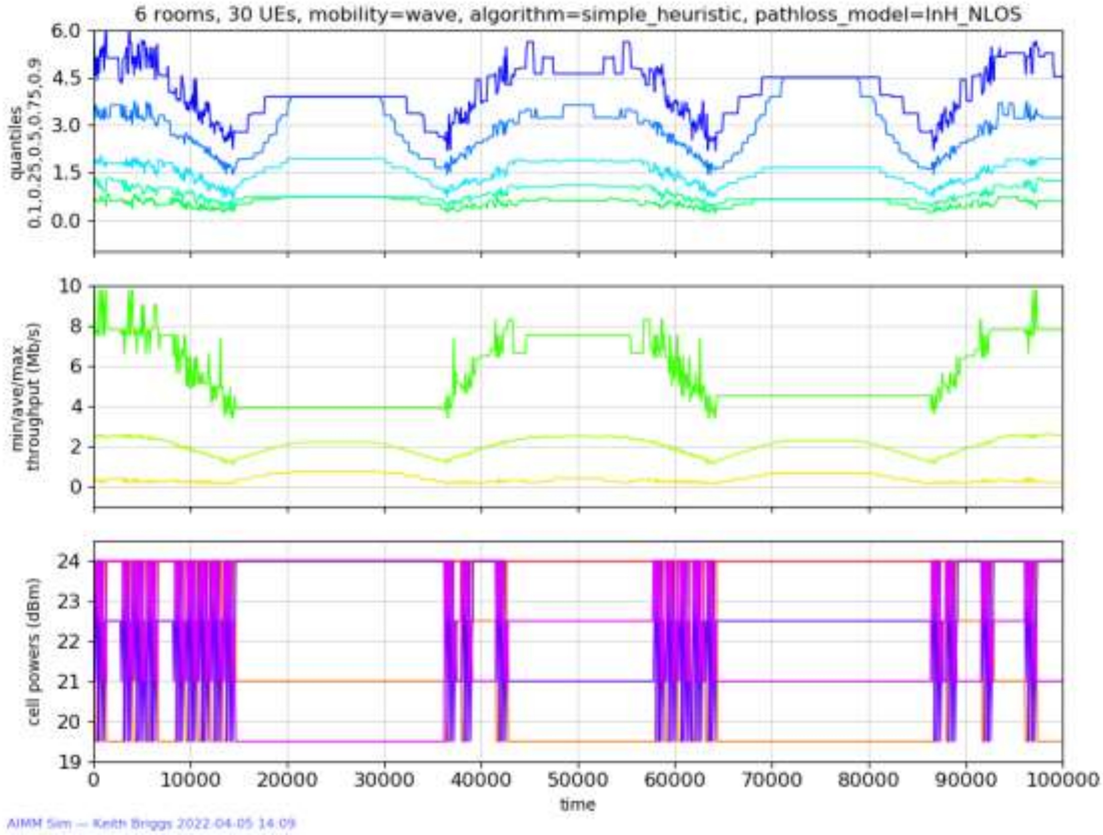


Figure 41 - Simple heuristic, non-line-of-sight.

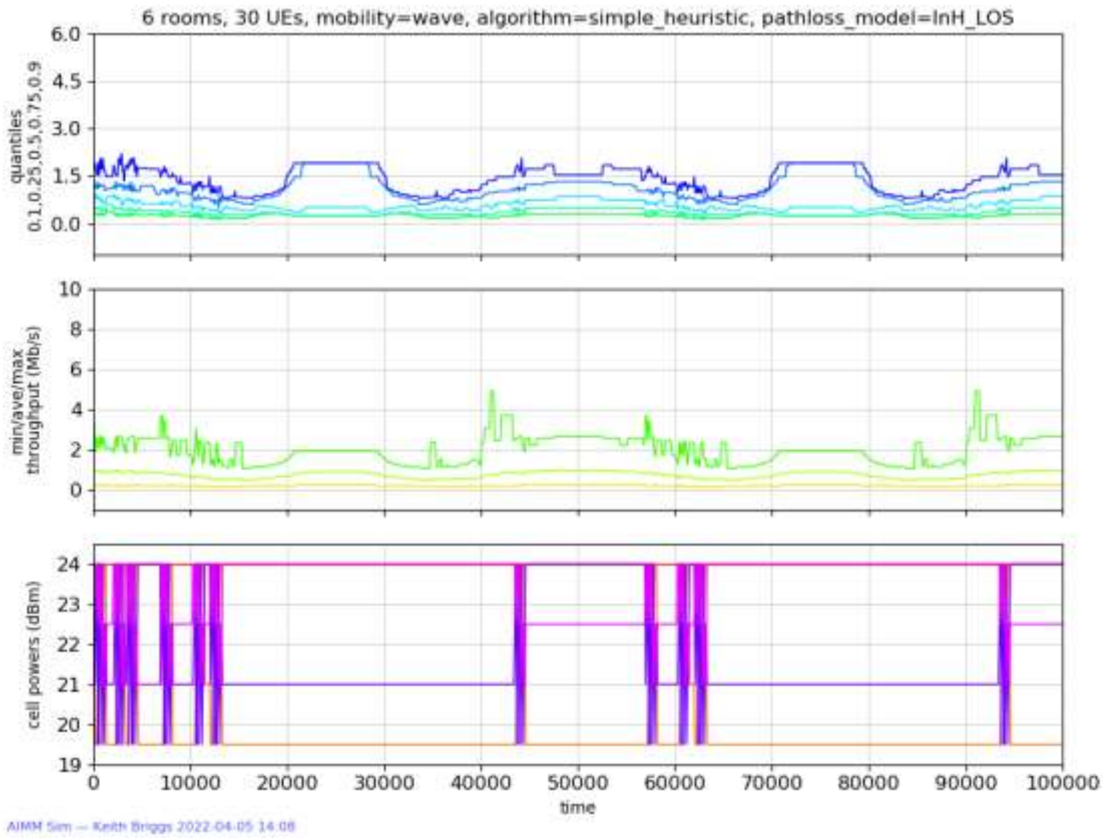


Figure 42 - Simple heuristic, line-of-sight

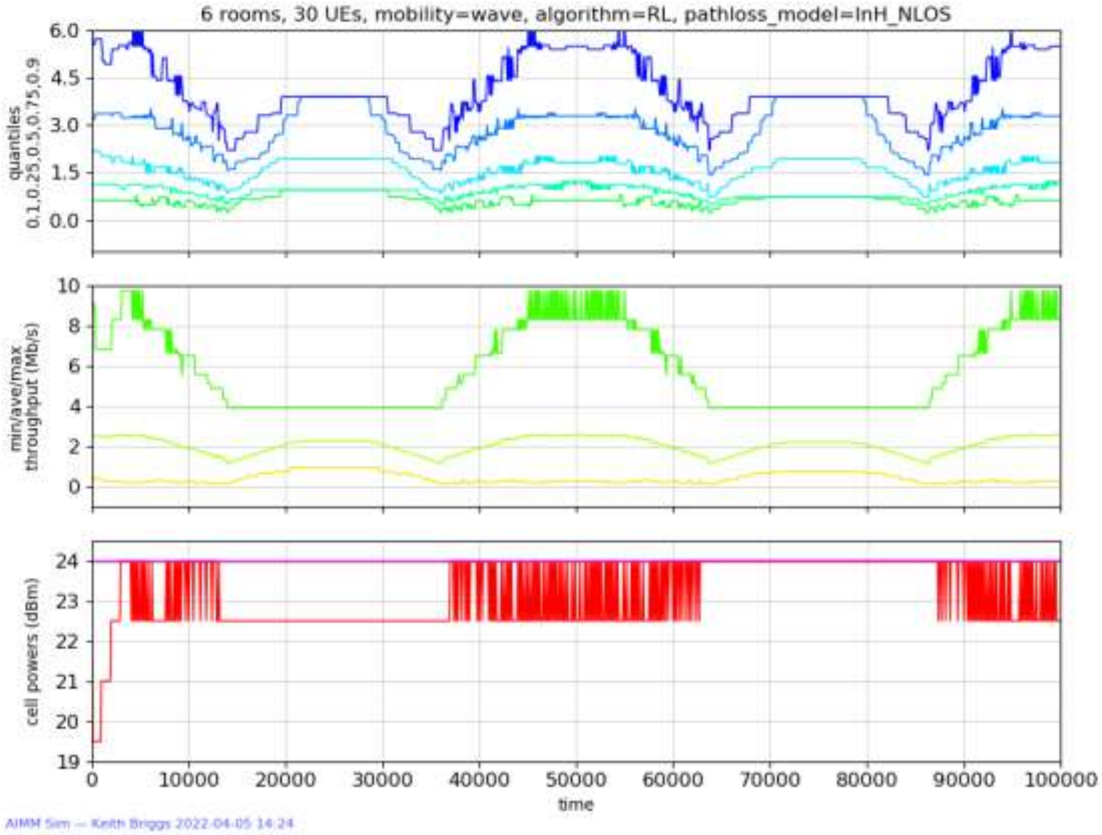


Figure 43 - Reinforcement learning, non-line-of-sight.

Figure 44 - Reinforcement learning, line-of-sight

4.4 Transmit Power Control for Indoor Small Cells: A Method Based on Federated Reinforcement Learning

4.4.1 Introduction

With the continuous development of 5G technologies, 5G-related services inevitably have begun to enter indoor environments, with signal coverage provided by microcells or femtocells. The deployment location, power setting, resource allocation, and antenna gain of such small cells will greatly affect the quality of service (QoS) for UEs. Therefore, there is rich research targeting the optimization of small-cell-related settings. Recently, machine learning (ML)-based, especially reinforcement learning (RL)-based algorithms appear attractive in this domain because of their proven success in solving complex optimisation problems. For instance, the interference control in a heterogeneous network utilising Q-learning was discussed in [4.17]. Similar Q-learning-based power control for indoor voice over LTE radio bearer was proposed by [4.18]. Recently, Mismar et al. put forward a deep Q network (DQN)-based method for joint beamforming, power control, and interference coordination [4.19]. In [4.20], multi-agent RL (MARL) is adopted to realise self-organising and power control in heterogeneous networks. In comparison, the MARL method is adopted in [4.21] to tackle interference mitigation for indoor coverage for 5G (and beyond) systems. In [4.22], the authors put forward an RL framework for uplink power control. A federated DQN approach for user access control is proposed in [4.23] under the context of O-RAN.

However, it is noticeable that existing approaches are discussed without differentiating scenarios and are mainly for outdoor macro cells. The underlying assumption is that such environments share the common signal transmission properties and UE patterns, so that the trained RL models can be applied to other scenarios. However, such assumptions do not hold for indoor scenarios. As the UE moving patterns largely depend on (or are limited by) the layout of the room, the optimal model in one room can be drastically different from others, i.e., the model is room-dependent, which is difficult to serve in other rooms. To increase the model's generalisation ability, a training process considering multiple indoor environments is needed. Also, from the view of the indoor network provider, it is necessary to have a general model that can be deployed into a new scenario with zero or minimal amount of learning. Meanwhile, the training process ought to be controllable and not consume too much bandwidth.

The two considerations motivated us to develop a federated reinforcement learning (FRL) framework in this work. The adoption of FRL involves the updates of RAN at the hardware and cloud system and correspondingly, the data collection and model deployment pipeline. Fundamentally, the O-RAN architecture enables the feasibility of executing the ML/RL model through radio intelligence controllers (RICs) [4.24]. For each room, an independent RL agent is needed, while all rooms together cerate the federated learning (FL) learning paradigm. It is worth pointing out that the definition of "state" in the RL model used here relies only on off-shelf cell information, like CQI. FRL is promising because it involves neural network parameters communication rather than real user data, which removes privacy concerns for indoor

UE information. Meanwhile, the global model trained by FRL can adapt to a new environment more rapidly. The FRL framework is a step towards intelligent RAN. The contributions of this simulation work are summarised below:

- For indoor cell transmit power control, this is the first work that considers the variation of RL model training in different room layouts.
- We put forward an FRL framework to solve the generalisation, distribution, and adaption problems of the model under the context of O-RAN.
- Extensive simulations are performed to demonstrate the gains on throughput and generalisation ability of the proposed method.
- The simulation process strictly follows the hierarchical orchestration structure of O-RAN, where the RICs are established on top of the simulation environment. It provides a simulator design paradigm compatible with ML and RL. The document of code for the entire simulator is available at: https://aimm.celticnext.eu/simulator/.

4.4.2 Background

4.4.2.1 O-RAN

O-RAN is a new emerging architecture for the radio access network. It is attracting much attention due to two proprieties: openness, and intelligence. Openness means that it adopts a standard and well-defined hardware interfaces and software services, so the equipment or interfaces involved in O-RAN are not vendor-specific. More importantly, it embraces artificial intelligence (AI) in its basic standard formulation. Two types of RIC are designed in O-RAN to realise intelligent control of the entire network: non-real-time RIC and near-real-time RIC. AI or ML models can be deployed into RICs in the form of microservice applications, i.e., xApps and rApps.

4.4.2.2 RL and DQN

RL is a class of learning paradigms in ML. The agent focuses on the actions of interacting with the environment to achieve the largest accumulative rewards. DQN is a relatively mature and a widely used algorithm of RL. It has been proposed for controlling complex video games only using images [4.25]. The idea of DQN lies in the use of deep neural networks (NN) f_θ , to estimate the state (s)-action (a) value (Q value), that it f_θ = $Q(s, a)$. Taking the action a in the given space, the optimal policy can be constructed as:

$$\pi^*(s) = \arg\max Q^*(s, a). \tag{21}$$

$Q*$ (s, a) obeys the Bellman optimality equation [4.2]:

$$Q^\pi(s, a) = \mathrm{E}_{s' \sim S}\left[r + \gamma \max_{a'} Q^*(s', a') \middle| s, a\right]. \tag{22}$$

To learn the Q value at iteration i, the following loss is minimised with respect to θ:

$$L_i(\theta_i) = \mathrm{E}_{s,a \sim \rho(\cdot)}\left[(y_i - Q(s, a; \theta_i))^2\right], \tag{23}$$

Where:

$$y_{i,Q} = \mathrm{E}_{s' \sim S}\left[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1} | s, a)\right]. \tag{24}$$

Meanwhile, an experience replay mechanism and a target network are introduced in DQN to stabilise the training process.

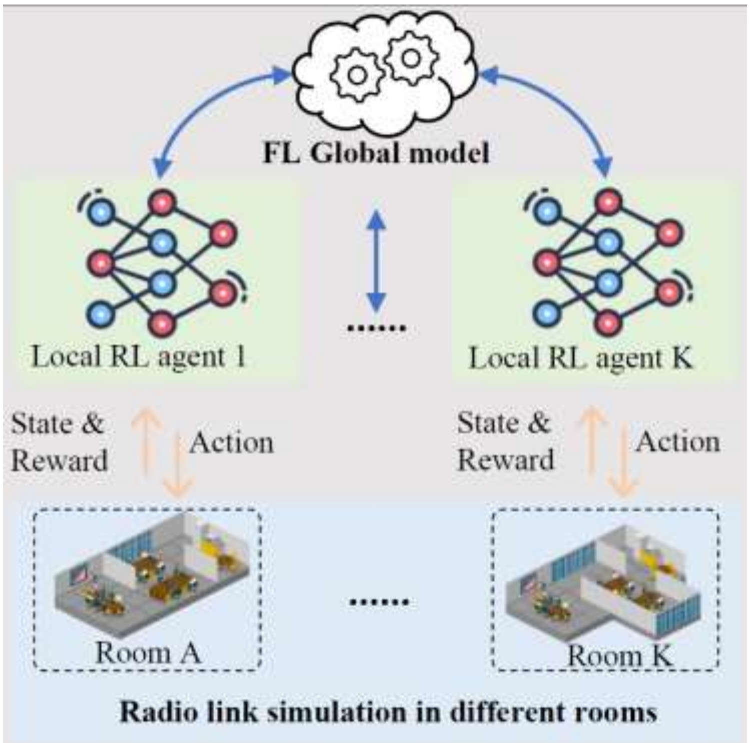


Figure 45 - System diagram of FRL framework

4.4.2.3 Federated Learning

FL is an ML setting in which multiple clients collaboratively train a model under the orchestration of a central server, while keeping the training data decentralized [4.26]. Due to concerns of privacy and communication efficiency, the training paradigm is that local models need to upload the model parameters to the global model (in the central server), and the global model returns the model parameters after parameters aggregation. In this work, we apply FL to the RL paradigm.

4.4.2.4 Simulator Design

In this simulation work, the simulator not only plays the role of radio link simulations, but also undertakes the work of office layout and UE trajectory generation. It is also the venue of RL agent and FL instantiation and training. Meanwhile, the trained model aims to be transferred to the O-RAN. Hence, the interfaces between RL agents and radio simulations should be specified. The definition of the simulator's functionalities obeys the hierarchical architecture of O-RAN strictly. It adopts a process-based discrete-event simulation framework, so different processes like RICs, Logs, and mobility management entities (MMEs) execute in parallel without interfering with each other.

4.4.3 Problem Formulation

This work studies a scheme for transmission power control of small cells in distributed indoor environments. Each indoor environment is viewed as being controlled by an independent local RL agent, while multiple RL agents are orchestrated by FL. FL distributes the model from the central server to the local agent, and aggregates local models to a new global model periodically. The system diagram is shown in Figure 45.

SNR(dB)	CQI Index	Modulation	Code Rate(× 1024)
$-\infty$	0	Out of Range	—
-6.9360	1	QPSK	78
-5.1470	2	QPSK	120
-3.1800	3	QPSK	193
-1.2530	4	QPSK	308
0.7610	5	QPSK	449
2.6990	6	QPSK	602
4.6940	7	16-QAM	378
6.5250	8	16-QAM	490
8.5730	9	16-QAM	616
10.3660	10	64-QAM	466
12.2890	11	64-QAM	567
14.1730	12	64-QAM	666
15.8880	13	64-QAM	772
17.8140	14	64-QAM	873
19.8290	15	64-QAM	948

Table 5- CQI Table [4.27]

4.4.3.1 Cells Transmit Power Control in a Single Room

It is assumed that there are M cells and N UEs in a single room and no subband or physical resource block allocation is considered. The downlink data rate Cm,n from the cell m to UE n can be modelled as follows:

$$C_{m,n} = B_m \log_2 \left(1+\mathrm{SINR}_{m,n}\right), \tag{25}$$

where Bm is the bandwidth of the cell m and SINRm,n is the Signal-to-Noise plus Interference Ratio (SINR), which is determined for the transmission from cells to UEs. The SINRm,n is defined as follows:

$$\mathrm{SINR}_{m,n} = \frac{G_m G_n P_m \ell^{(m)}(d_n)}{W_m + \sum_{i=0,i\neq m}^{M} I_{i,m}} \tag{26}$$

Where:

- Gm, Gm are the transmission and receiver antenna gains.
- Pm signifies the transmission power of cell m.
- $\ell(m)$ (dn) expresses the path-loss at a distance dn (between cell m and UE n).
- Wm represents the thermal noise power.
- Ii,m is the interference power received.

In the process of code implementation, the SINRm,n will be first converted to the corresponding Channel Quality Indicator (CQI) value, then the final data-rate is calculated according to the relationships demonstrated in Table 5.

Optimisation Objective: For a local RL agent, the optimisation objective is to maximize the overall throughput of the entire room. The objective function is written as:

$$\max \sum_{m \in M} \sum_{n \in N} C_{m,n}, \quad \text{s.t. } P_m \in P_{\text{POT}}, \tag{27}$$

where P_{POT} is the set of possible transmission power levels.

Algorithm 1: Federated DQN for power adjustment of indoor cells.

Require: The setting of different indoor scenarios.
Initialize K clients with network Q_k and $\hat{Q}_k$, and Global model Q_{Global}.
Initialize the experience replay memory D.
Initialize the agent to interact with the environment E_k.
while *not Done* **do**
  for $k = 1, K$ **do**
    Update model Q_k by Q_{Global}
    for $t = 1, T$ **do**
      Reset the environment
      Set the initial state $s = s_0$
      With probability ϵ select a random action a_t
      Otherwise $a_t = \arg\max_a Q_k(s_t, a; \theta, W)$
      Execute action a_t in environment k and observe reward r_t and new state s_{t+1}
      Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
      Sample random minibatch of transitions $s_j, a_j, r_j, s_{j+1})$ from D
      Perform a gradient descent step on $(y_i - Q_k(\phi_j, a_j; \theta, W))^2$
      For every C steps, set $\hat{Q}_k = Q_k$
    end
    Upload model Q_k to Q_{Global}
    Wipe D
  end
  For every E cycles, aggregrate the global model Q_{Global} using equation 10.
end

4.4.3.2 Markov Decision Process

We formulate the problem in (27) as a finite Markov decision process (MDP). An MDP is defined by the tuple (S,A, P, R, γ), where the set of environment states is represented by S; A is the action space of agent; P is the transition probability from state $s \in$ S to state $s' \in$ S for any given action $a \in$ A, and R is the reward function. γ is the discount factor.

Steps and Episodes: In one room, an episodic task is defined. Each episode contains 100 sequential steps, while UEs move to new locations at each step. The possible locations of UEs are initially pre-generated by a “billiard” model [4.28], when initialising the room layout. In this model, users bounce off walls. In the sequential decision-making problem, the RL agent looks for the optimal combination of transmission powers for all cells at each step and naturally in every episode. At each step, when a new action is performed, handover events for all UEs will be triggered immediately. The handover decision is based on Reference Signal Received Power (RSRP), which means that the UE always attaches to the cell with the highest RSRP.

Furthermore, at time t, for $m \in M$ and $n \in N$, the state, action and reward of the deep RL agent are defined as below:

- **State**: The state st is consist of three parts:
  - 1) the normalised transmission power of current M cells Pm,t ;
  - 2) the number of UEs attached in the each cell NUm,t ;
  - 3) the CQI information of all UEs reported to different cells, that is $CQIm,n,t$.

So, the overall state is:

$$s_t = (P_{m,t}, NU_{m,t}, CQI_{m,n,t}) \tag{28}$$

- **Action**: The action at at time t is to select a transmission power for each cell. The action space is discrete.
- **Reward**: The training criterion is the throughput of all UEs. In this simulation, under a joint consideration between the maximum throughput and the QoS guarantee for UEs, we take the 0.25 lower quantile of the distribution of throughput across all UEs, denoted Q1, as the optimisation objective. So the reward r (st , at) of executing action at at state st is defined as the quantile improvement of the entire network after this action, and any subsequent handover, have taken effect.

$$r(s_t, a_t) = \mathbf{Q_1}(a_t) - \mathbf{Q_1}(a_{t-1}) \tag{29}$$

It should be noticed that the reward design for this optimisation problem is flexible and goal-related. It depends on the focus of optimisation. For instance, we formalise the reward to equation (29) because we are concerned about the lowest QoS guarantee for all UEs. However, if more attention is paid to the balance of workload of cells, the reward can easily be redefined.

4.4.3.3 The FRL Algorithm

FRL is a promising and efficient method of RL to create a distributed paradigm and so preserve data privacy. In our case, FRL consists of multiple independent RL agents serving multiple rooms. Each local agent is acting as the cell's transmit power controller for one room based on the global DQN model, which is aggregated by an FL algorithm, as shown in Figure 45. We adopt Fedavg as the default algorithm for global model aggregation [4.29]. For local agent $k \in K$ with model parameters θk, the aggregation operation is expressed by equation (30):

$$\theta_{\text{Global}} = \frac{1}{K} \sum_{k=0}^{K} \theta_k . \tag{30}$$

Local agents upload their model parameters to the central server every E cycles. Then the global model will be broadcast back to all agents after aggregation, and the global model serves as the pre-trained model for each agent after broadcasting. The RL models will be installed in the RICs of ORAN through the form of xApps or rApps, to perform local training and parameter uploading, while the global model can be deployed in the central server of network operators. The overall FRL training scheme is illustrated in Algorithm 1.

4.4.4 Simulation

4.4.4.1 Room Layouts and User Mobility Mode

To evaluate the performance of the proposed FRL scheme, we defined five typical room layouts. As shown in Figure 46, all rooms are of 4m height. Rooms A and B are narrow rectangular layouts with a size of 18×6m. Room C is L-shaped and room D is T-shaped, while room E is L-shaped in another direction. They are all of dimension 18×12m. The yellow rectangles represent the interior wall panels of the room; the red points are the indoor cells deployed. The blue traces are the potential UEs locations generated by the Billiard model [4.28]. Since we are dealing with heterogeneous UE distributions, which are directly caused by different room layouts, the billiard model can reflect the room layout information as much as possible, which is helpful in evaluating the RL performance. For the same reason, we maintain the positions of edge users that seem to penetrate the room's interior walls. The user trajectories are sampled from these locations according to the number of steps in each episode. To increase the generalisation ability of the RL agent, at each step, we add random position offsets in initial x, y locations respectively, which are sampled from Gaussian distributions with mean μ = 0 and variance σ = 0.5. The height of all UEs is fixed at 1m.

4.4.4.2 Radio Simulation Setting

The cells modelled in this study follow the 5G gNodeB architecture. According to engineering experience, we reasonably assume that M = 2 cells and N = 30 UEs are typical for one large room.

The five rooms are initialised for FRL to reduce the simulation complexity. For parameters related to Eqs. (25–26), Bm is 20 MHz; Gm and Gn are 0 dBi; Wm is constant. The initial transmission power is 24 dBm. The indoor path-loss model shown in equation (31) is used for ℓ, which comes from 3GPP TR 36.873 version 12.7.0 Release 12.

$$\begin{aligned} \ell_{\mathrm{Los}} &= 22\log_{10} d_{3D} + 28.0 + 20\log_{10} f_c \\ \ell_{\mathrm{NLos}} &= 36.7\log_{10} d_{3D} + 22.7 + 26\log_{10} f_c - 0.3(h_{\mathrm{uT}} - 1.5), \end{aligned} \tag{31}$$

where fc is 3.5 GHz, and huT = 1m. It is to be noted that 3GPP updates the indoor propagation models in different releases, but the variation between such models is minor and has negligible influence on our RL training.

For every cell, we assume PPOT. = [19.5, 21.0, 22.5, 24.0] dBm, i.e., there are four power levels for each cell, thus the total number of power levels, for 2 cells, is 16 possible combinations. After excluding those combinations with the same interference proportion, the action-space size of the RL agent is 11.

4.4.4.3 FRL Setting

DQN and FedAvge are adopted as algorithms for the proposed FRL framework. Qk and $Q\hat{}k$ are deep neural networks (NN) with fully connected layers; hyperparameter details can be found in Table 6. The hyperparameters in this table are reasonable empirical values, based on our experience of running many simulations with varying values. The RL agents of the first four rooms (A–D) are used for the federated global model training, and room E is used for the validation of the FRLmodel.

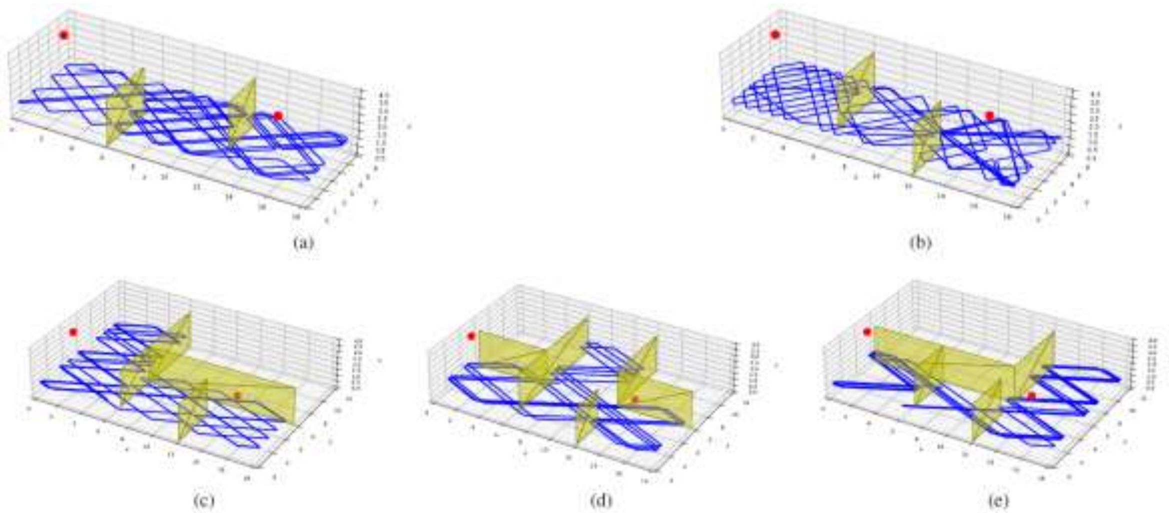


Figure 46 – (a)–(e) show different 3D layouts of rooms (room A–E). The yellow panels are the interior walls. The blue lines are possible UE locations generated by the billiard algorithm. Red dots indicate the locations of cells

4.4.4.4 Baseline: exhaustive search

To provide a reliable baseline for evaluation of the FRL method, we exhaustively search through all allowed power levels and then select the power setting which achieves the highest throughput. This is guaranteed to correctly maximize equation (27) and is feasible in our test scenarios because of the small problem size.

4.4.5 Results

4.4.5.1 Training of the Single RL Agent and FRL

The reward during training RL agents in room A-D are illustrated in Figure 46,where each agent is trained independently five times, to evaluate the amount of variation in the training process. Each training phase lasts around 2000 episodes. It can be seen that, the single RL agent works well for the corresponding scenario. Although the convergence time varies, a stable reward gain can always be observed. It is noticeable that the reward varies in each room; this is because of the heterogeneity of the UE distribution across different rooms.

As discussed in the above section, we train an FRL global model using rooms A–D. The global model aggregation happens every E = 380 cycles. The training curves of local clients (A–D) are demonstrated in Figure 47. It can be observed that the reward drops every 380 cycles; this is where the aggregations happen. The whole FRL training process ends with the convergence of each client.

In the single RL validation stage, the trained model is frozen and deployed in the same environment as in the training stage. We calculate the cumulative throughput of the entire network based on the 0.25 quantile and average data-rate of all episodes. The results of the random power allocation, RL model and exhaustive search method are shown in Table 7. The RL algorithms outperform both the random allocation and exhaustive search method in any environment, and the trained global model of FRL shows a similar performance compared with signal FL. Moreover, it is noticeable that the trained RL model shows great advantages in terms of inference time. When the UE locations change, the DQN only needs to make one forward inference to get the optimal transmit power setting, which is a capability that the greedy algorithm can't match.

Name	Value
FL algorithm	FedAvg
Number of Clients K	5
Aggregation cycles E	380
RL algorithm	DQN
Exploration rate ϵ	0.9
Batch size	128
Maximum timesteps in each episode	100
Target network update interval	100
Reward discount factorγ	0.98
Optimizer	Adam
Learning rate	0.001
Layer type	fully connected layer
Number of neurons of each layer	[200,100,50]
Activate function (not for output layer)	Relu
Activate function for output layer	Linear

Table 6 - Hyperparameters of Federated Reinforcement Learning

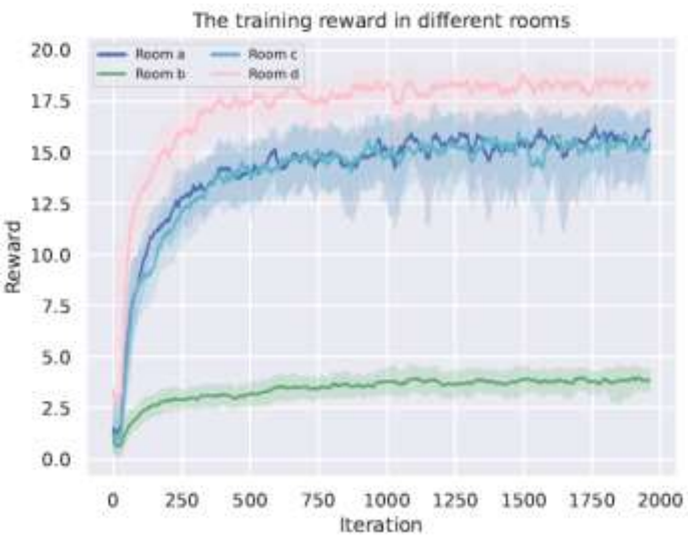


Figure 47 - RL training reward in different rooms

Algorithm	Criterion	Room A	Room B	Room C	Room D
Random	Q_1	93	66	62	46
	Avg.	122	109	117	65
Exhaustive	Q_1	103	156	109	112
	Avg.	179	207	216	184
Single RL	Q_1	114	163	118	113
	Avg.	221	223	218	225
FRL	Q_1	115	164	113	112
	Avg.	219	223	219	225

Table 7 - The Cumulative Throughput Compare Based on 0.25 Quantile ($Q1$) and Average for All UEs (In Mbps)

4.4.5.2 Adaption Test of FRL Global Model

To validate the generalisation and adaptation ability of the FRL approach, the model trained in rooms A–D is tested in a new environment (room E). Two single RL agents are trained. One is trained from scratch; another one is trained from a FRL model pre-trained in room A–D. The comparison can be found in Figure 49. It is obvious that the adaptation of pre-trained FedAvg global shows significant advantages in training speed and final performance. The RL agent trained from the FedAvg model converges faster than all others. This reveals that the knowledge learned in the global model can guide the model training in a new environment.

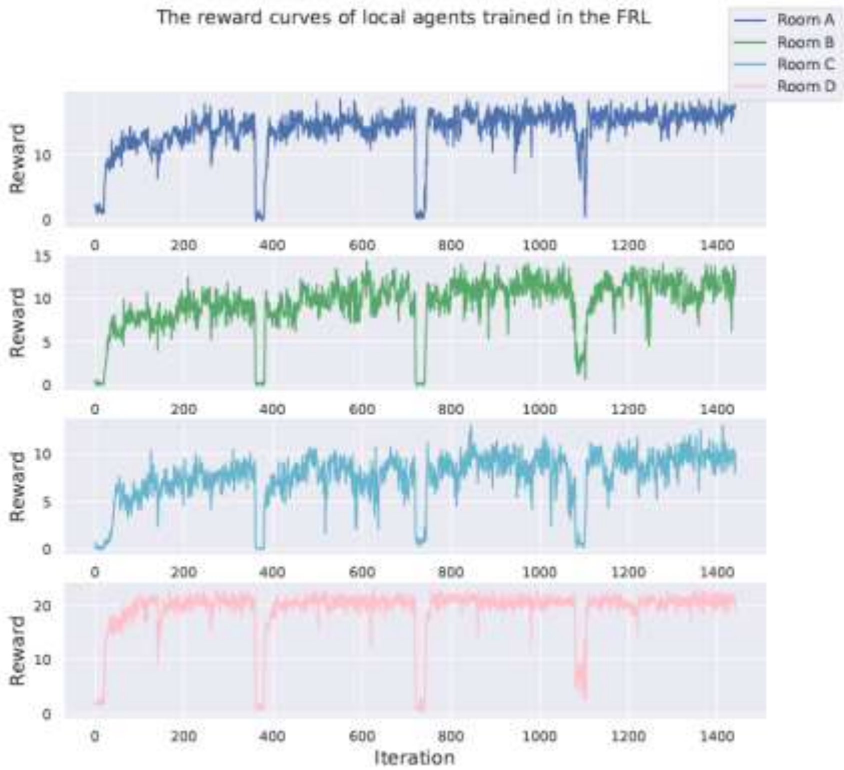


Figure 48 - The reward of agent trained in the FRL

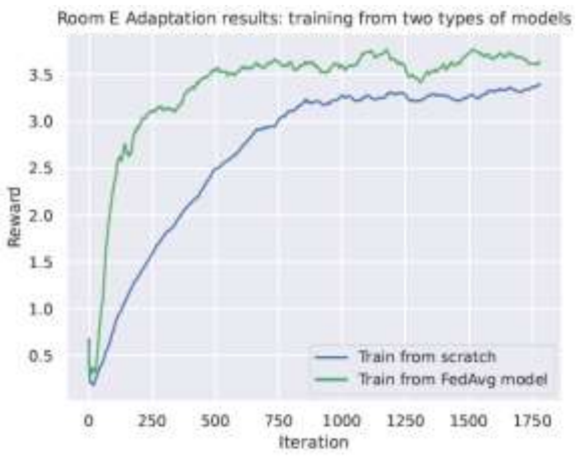


Figure 49 - Room C's RL model training from different pre-trained models

4.4.6 Discussion

We utilised FRL to solve indoor small cell transmission power control problem. The FRL framework ensures the privacy and security of UEs and can provide a template for model distribution, which fits the xApps model framework. Network operators may move towards intelligent networks with our proposed methods. For the RL-based controller, there are still some problems waiting to be explored. One of the problems is that when we increase the number of cells or add other optimization options, the action space will grow exponentially, which can lead to a large increase in training costs. Although some schemes such as action-space encoding and actor-critic structure can partially solve this problem, the effect is not satisfactory. On the other hand, we assume the path-loss models in the different indoor environments are the same, but, due to the multipath effect of indoor environments, such empirical models are not reliable, which results in the simulation-versus-reality issue needing to be addressed [4.30]. In the future, we will consider the joint optimisation of transmit power, physical resource block, loading balance etc.; all these optimisations will be unified in our proposed FRL approach.

4.4.7 Conclusions

This study discusses the issue of indoor cell transmit power control in the context of O-RAN, emphasizing the room dependent properties and lack of generalisation ability of a single RL model. Based on this, we propose an FRL framework. The client is in a single indoor environment and learns the best policy by RL. All clients will periodically upload model parameters and integrate them in the global model. The global model will act as the base model for learning in new environments. The simulation results demonstrate the feasibility and advantages of the proposed method, both in throughput and the learning efficiency.

4.4.8 Outcome of the study

This simulation work is accepted by IEEE 96th Vehicular Technology Conference (VTC2022-Fall) to be published as cited below,

* Li, P., Erdol, H., Briggs, K., Wang, X., Piechocki, R.J., Ahmad, A., Inacio, R., Kapoor, S., Doufexi, A. and Parekh, A., 2022. "Transmit Power Control for Indoor Small Cells: A Method Based on Federated Reinforcement Learning". In 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall), 2022.

4.5 Smart Interference Management

Interference continues to be a key limiting factor in cellular radio access network (RAN) deployments. Effective, data-driven, self-adapting radio resource management (RRM) solutions are essential for tackling interference, and thus achieving the desired performance levels particularly at the cell edge. In future network architecture, RAN intelligent controller (RIC) running with near real-time applications, called xApps, is considered as a potential component to enable RRM. In this work, integer linear programming and reinforcement learning (RL) enabled sub-band masking xApp for effective radio resource management is proposed as a solution for smart interference management.

Here, we have developed a simplified model for testing algorithms. The model is a standard type used in this kind of work; it is mathematically a geometric random graph (GRG), and consists of a set of nodes representing cells, and edges between nodes which are geographically close together, representing potential interference. UEs are at cell centre as well as cell boundary in macro-cell based homogeneous networks. Here, the signal strength from serving cell tends to be very weak for cell-edge users as signals from neighbouring cell act as interferers. Therefore, the development of effective radio resource allocation procedures is essential to mitigate interference thus improve average throughput of the edge users whilst delivering a guaranteed QoS in next generation wireless networks. The developed algorithms were assessed on a range of networks graphs with number of base stations ranging from 10 to as a maximum of 100. Our standard test problem used GRGs of sizes 10 to 30, but always of mean degree 4, and the target was to allocate 4 subbands out of 13, with the number of clashes minimized. The mathematical as well as RL approaches are exploited on the same environment. The developed xApp is scalable in both storage and computation.

4.5.1 Interference Management using Mathematical Model

We considered a downlink multi-cell OFDM network where each node i has variables $x[i,:]$ taking values 0 or 1. The second array axis is the number of available subbands. $x[i,j] = 1$ means that node i is allocated subband j. Constraints such as $sum\_j\ x[i,j] = n$ ensure that each node has the required number of subbands. One possible objective function is the total number of subband clashes; a clash is a case of

$$x[i,j] = x[k,j] \tag{32}$$

where nodes i and k are neighbours. Minimizing an objective of this type, subject to the constraints, is *integer linear programming (ILP)*, for which good exact algorithms are available, even though this is an NP-complete problem. Problems with up to 50 nodes were solvable in reasonable time (less than about a minute). In each cell, a BS is in the centre of the cell while UEs are either at cell centre or cell edge. We have used ILP solutions to provide the exact solution as a benchmark for our heuristics, essentially to test how often they reach the true minimum. The heuristics were of two types: (1) greedy, and (2) Q-learning.

The greedy heuristic works as follows:

1. Pick a node at random.
2. If there are any subband clashes with neighbours, do the following: re-allocate the 4 subbands in such a way as to minimize neighbour clashes. This can easily be done by choosing the 4 subbands which are least used by neighbours.
3. Go to 1.

This is thus a distributed heuristic, which makes repeated local optimizations with no knowledge of the value of the global objective function. Nevertheless, it works extremely well, always finding the global optimum (although it doesn't know that it has found it), typically in a few hundred steps, and taking something like 10^{-4} of the CPU time of ILP. Also show in Figure 50.

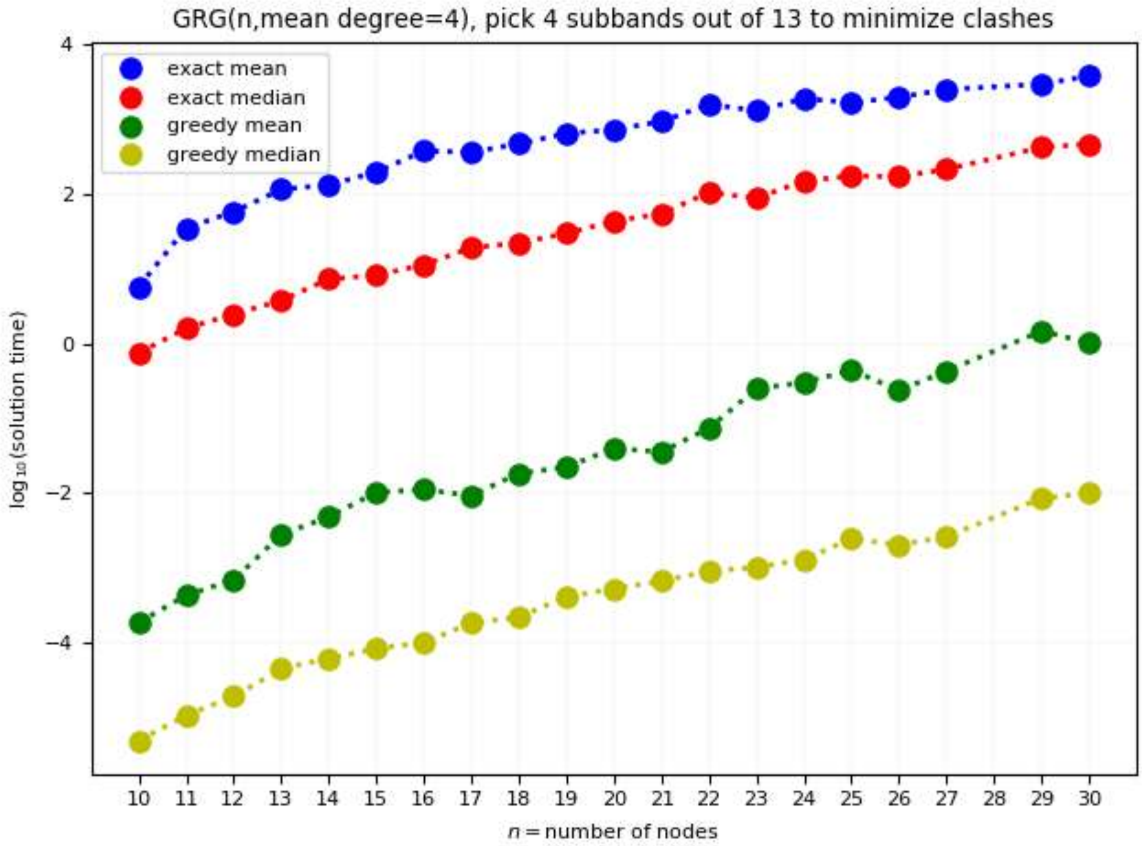


Figure 50 - Time complexity of ILP and Greedy heuristic techniques

As there is no concept of a stopping condition in such a heuristic, in a real system we imagine it running forever. In such a way, even if it never finds the true optimum, it will nevertheless keep improving the objective. This greedy heuristic is thus setting the challenge which AI must beat, and it would not be surprising if AI failed to beat it. If that is the case, we have shown that AI is not a good solution to this problem.

4.5.2 Interference Management using Machine Learning

The problem is posed as a cooperative multiagent control problem using reinforcement learning techniques. One popular RL technique in widespread use is Q-learning (QL). Here, a centralized array or look-up table known as the Q-table is maintained. The values in this table are called Q-values, initialized to zero, allowing the agent to start to learn with an equal choice among all available actions. The Q-table, therefore, presents an analysis of the choice of behaviour of all the individual agents, whereas the Q-value represents the expected cumulative reward the agent receives by learning an action.

In the presented scenario, each base station is a RL agent. During the learning phase, agents employ Є-greedy exploration-exploitation policy for action selection. An exploratory random action is picked with probability Є; otherwise, a set of good actions, that is, the number of sub-bands to be learnt per base station, is selected following a greedy policy with probability 1- Є. A set of greedy actions is selected from among the available actions using the equation 1. Thereafter, one action is randomly selected for the set of optimal actions learnt after greedy selection.

$$\text{action} = [\text{argmax}(\text{Q}(\text{action})), 1] \qquad (33)$$

RL is goal-directed learning; therefore, depending on the consequences of the learnt action a reward is awarded to the learning agent in case of successful attempts, else it is punished. As learning agent follows temporal difference learning approach; therefore, the Q-value of each learnt action is recursively updated using the Q-learning update equation below:

$$Q_{new}(s,a) = Q_{current}(s,a) + \alpha[r(s,a) + \gamma \max Q'(s',a') - Q(s,a)] \qquad (34)$$

The RL enabled algorithm works as follows:

Step I: Distributed Q-learning relying on environment interactions alongside subband usage information from a subset of neighbouring nodes.

1. Perform action selection using exploration and exploitation mechanism.
2. The exploration rate decreases gradually from high exploration phase towards high exploitation. The exploration decay is constant and occurs at the end of each episode.
3. Reward policy: punish if learnt resource is used by any neighbouring cell else reward.
4. Update Q table for each learnt action per cell.
5. Reduce the exploration rate and go to step 1.
6. Termination condition for distributed learning - current exploration rate is less than minimum exploration rate.

Step II: Learn Optimal Policy

7. Policy learnt by each agent is fed into central controller. The central controller assists to remove short term fluctuations to stabilize the control process.
8. The central controller assesses if each cell has learnt unique resource and policy learnt leads towards higher rewards in comparison to the previous policy learnt, thereafter, update Q table and set exploration rate to 1 for next episodes.
9. Termination condition: number of radio resource clash is constant for the last 100 iterations.

The aim is to learn *a* set of subbands per base station (4/13) such that the number of subband clashes in the environment are minimised that indicates to minimized interference. The results illustrated are gathered whilst training the model: (a) total number of subband clashes in the environment - all agents interact with the environment and learns unique solutions dynamically; thus, fluctuations in learnt solution are monitored; (b) optimal policy learnt by the model - the central entity accesses unique solutions learnt by each agent using distributed learning and local knowledge. Figure 51 demonstrates RL enabled RRM.

Figure 52 demonstrates the time complexity of RL technique in the training phase. The simulation was performed on a range of cluster size, i.e., the number of base stations in each environment was different, however, the average number of neighbouring nodes was same. The results help us to understand the convergence time a RL model would take if the environment scaled. It also demonstrates a box plot for convergence time. On each box, median is indicated by the central mark while the bottom and top edges represent the 25^{th} and 75^{th} percentile respectively. The whiskers extend to the most extreme data point not considered as outliers. Outliers are shown by the '+' symbol.

The conclusion could be drawn that there is a gradual increase in convergence time using RL model. On contrary, ILP technique illustrates to be an effective solution for environment where cluster size is less than 25 however, else it demonstrates a steep increase in convergence time and proves to be an expensive solution.

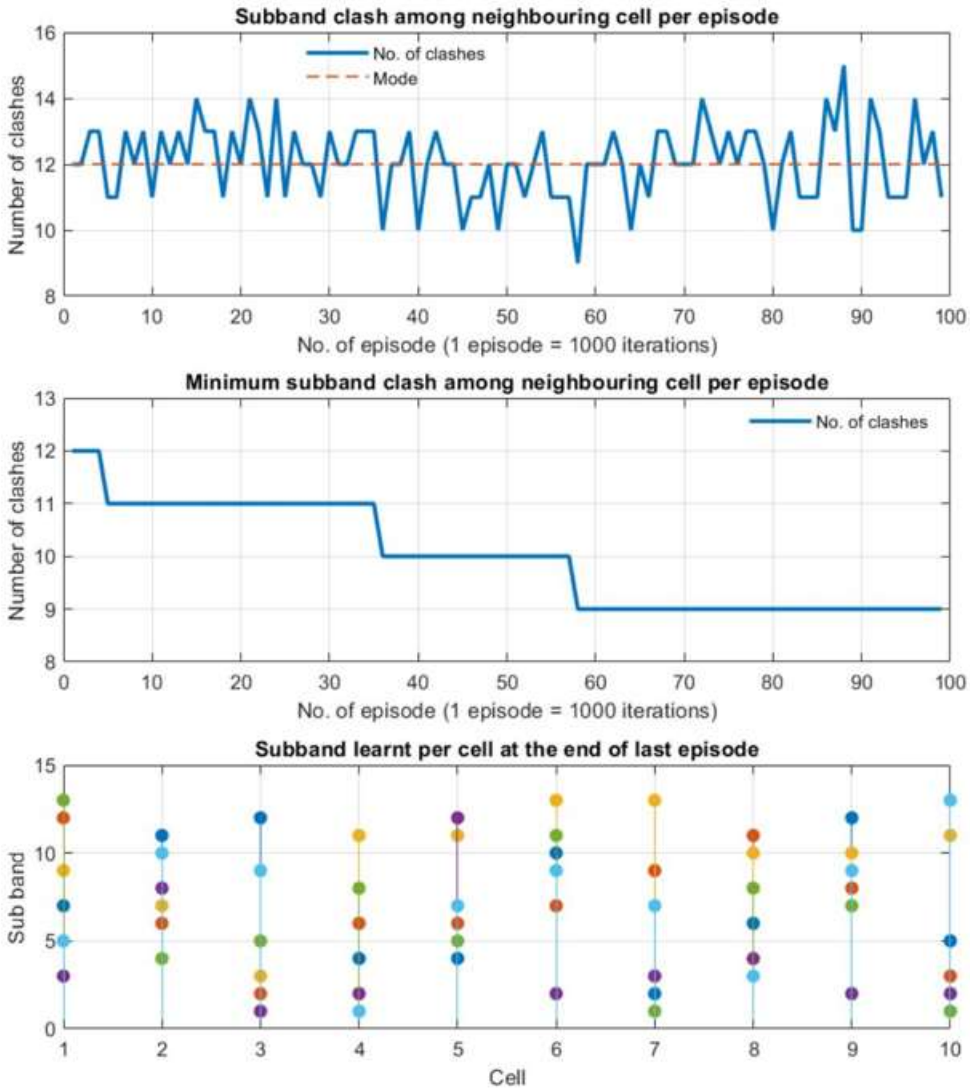


Figure 51 - RL enabled optimal subband selection for interference mitigation in a cluster of 10 cells

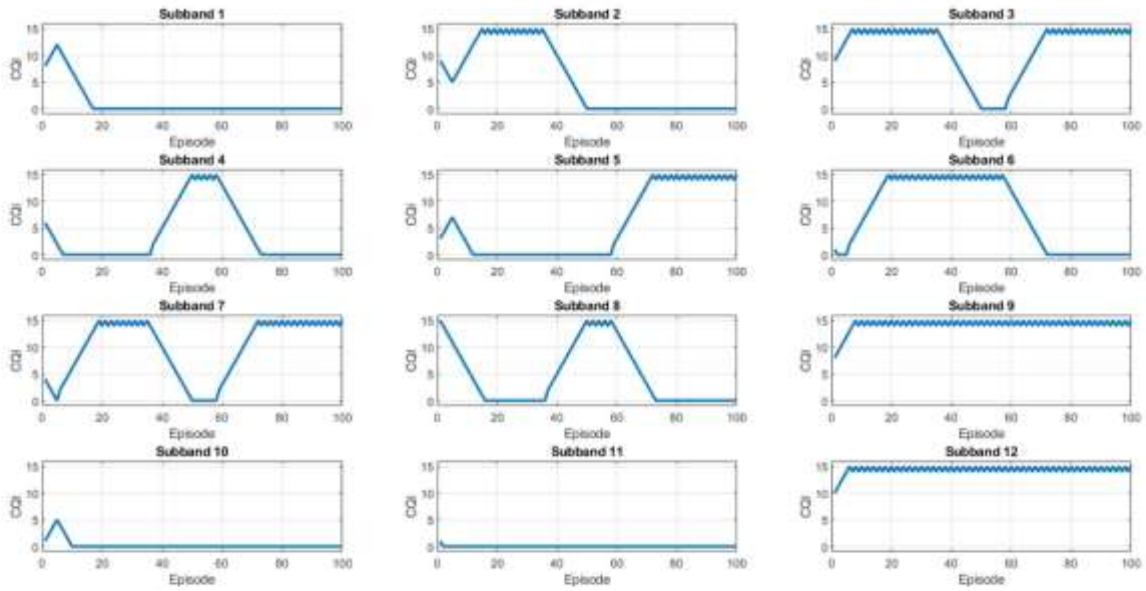
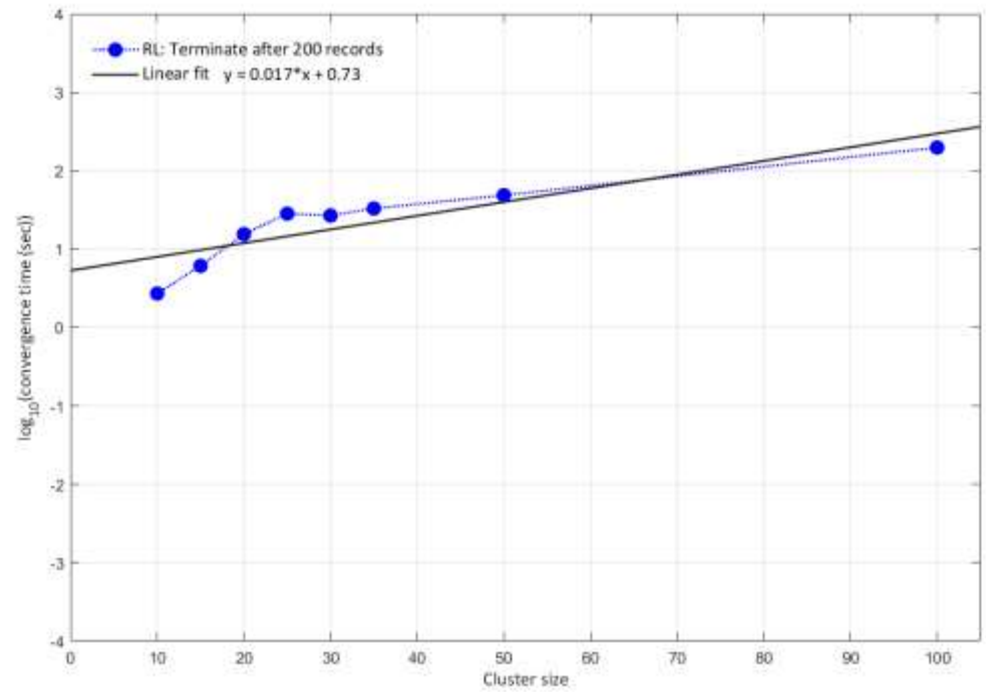


Figure 52 - RL mechanism optimal subband selection on a particular cell for Interference Management



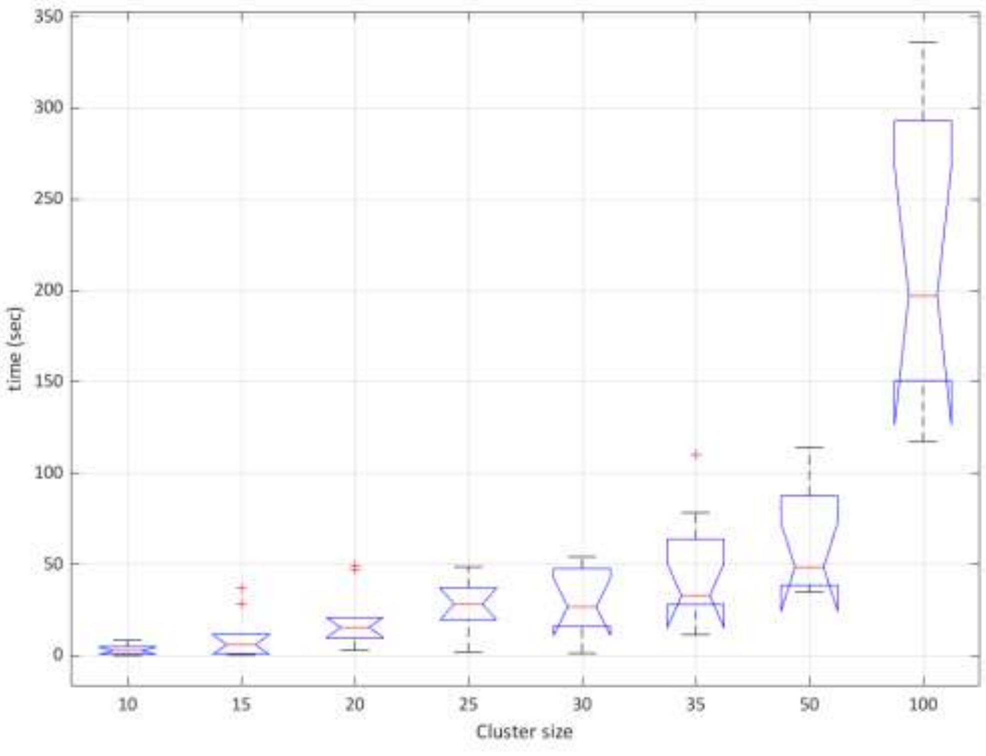


Figure 53 - Time convergence for RL model: (top) the box plot presents the median time and outliers as the network scales; (bottom) time complexity plotted on log scale.

4.5.3 Conclusion

The obtained result assists us to understand reliability, robustness, and scalability of different approaches for link scheduling in wireless environment. ILP provides an exact solution and is quick for networks with a low number of nodes; however, as the network scales, the convergence time also scales. The RL model has ability to update policy along with changes in environment, since it continuously interacts with the environment and updates the policy accordingly. Also, it is an expensive approach if the goal is to achieve quick solutions as a significant number of iterations are required during the exploration phase to learn an optimal policy. However, once an optimal policy is learnt, the agent can make an intelligent action selection, thus maximizing the rewards. Nonetheless, the RL approach is expensive in time complexity for larger networks where the number of nodes is greater than 25. Moreover, improvement in CQI values per subband has been monitored with both approaches.

4.6 Smart Mobility Management

The term “Smart Mobility Management” also known as “Smart Handover” refers to more advanced heuristics that make use of network information such as user and cell location, cell load, physical resource blocks, UE mobility and/or radio signal strength information to perform handover [4.31]. Handover is defined as a process in which the ongoing transmission is transferred from the current base station (BS) to a target BS, depending on cell association policies [4.32].

The number of handovers per transmission is termed as handover frequency. Traditional 4G and 5G handover heuristics uses signal strength information, RSRP reports from UEs, for optimal base station association. In dynamic small cell, vehicular environments, a linear increase in handover frequency is monitored with an increase in the vehicle speed, if the maximum radio signal strength (max-RSRP) user association approach is considered. This may lead to significant increase in handovers as well as switching and signalling load resulting in undesirable network performance. Therefore, considering the dynamic environment or shift in user traffic pattern due to environmental phenomenon there appears a need to develop data-driven, proactive ways for smart handover or mobility management. Algorithms that are dynamic and adaptive to network changes with an objective is to spread the load more evenly across the network while ensuring that user service levels are not compromised. In the following sections, mobility management using mathematical as well as machine learning enabled are discussed.

4.6.1 Mobility Management using Mathematical Model

A mobility model is one of the key determinants in the provision of accurate simulation. The role of mobility model is to mimic the movement behaviour of users, and thus UE location, in the network through the inclusion of critical movement factors such as direction, speed and destination, therefore correct design and selection of a mobility model is essential to evaluate the impact of developed protocols on network characteristics, and to analyse the network behaviour under a proposed protocol. A range of mobility models with different characteristics have been explored in simulation-supported research [4.33]. Amongst all, random walk model with reflection mobility model - a variation of random walk mobility model has been used to demonstrate mobility management. Here, a mobile node is initially placed in a random location in the simulation area, and then moved in a randomly chosen direction between [0,2π] at speed between [speedmin, speedmax]. As soon as mobile node reaches any edge of the simulation area, the node changes its angle of movement to (α + π/2) while the speed remains constant [4.33]. This mobility model imitates real life mobile nodes as they are more likely to reflect their direction of movement when meeting with an obstacle.

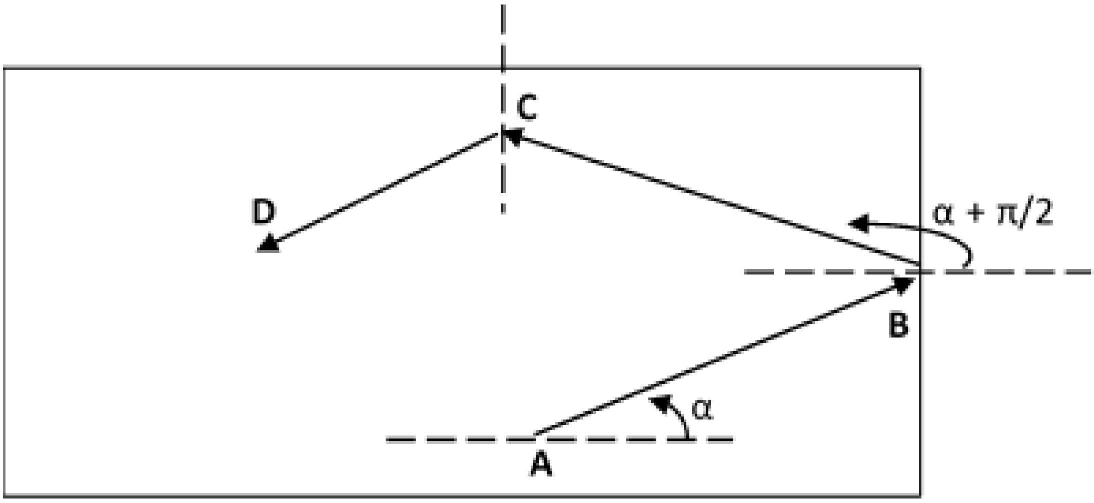


Figure 54 - Random Walk with reflection mobility model [4.33]

Further, mobility management entity which enables a handover based on maximum signal strength technique has been employed. Nonetheless, if the cells are not equidistant, then excessively frequent handovers may result (ping-ponging) [4.32]. The key objective of the work performed is to understand and monitor if application of reinforcement learning approach will lead to a similar or better handover performance compared to pure heuristic. Simulation results demonstrate that handover performance monitored using RL techniques is not only at par with its counterpart but also can lead to an improved minimum throughput.

4.6.2 Mobility Management using Machine Learning

The presented work considers a scenario containing 7 cells and 20 UEs. This has been designed to be large enough to be challenging, but small enough to run fast in the simulator. The cells are in a hexagonal arrangement, and the UEs follow approximately circular paths, designed to cross cell boundaries frequently to speed up the learning process as shown in Figure 55. The UEs move at different speeds. 50% of the UE traverse in opposite direction but follows the same circular trajectory.

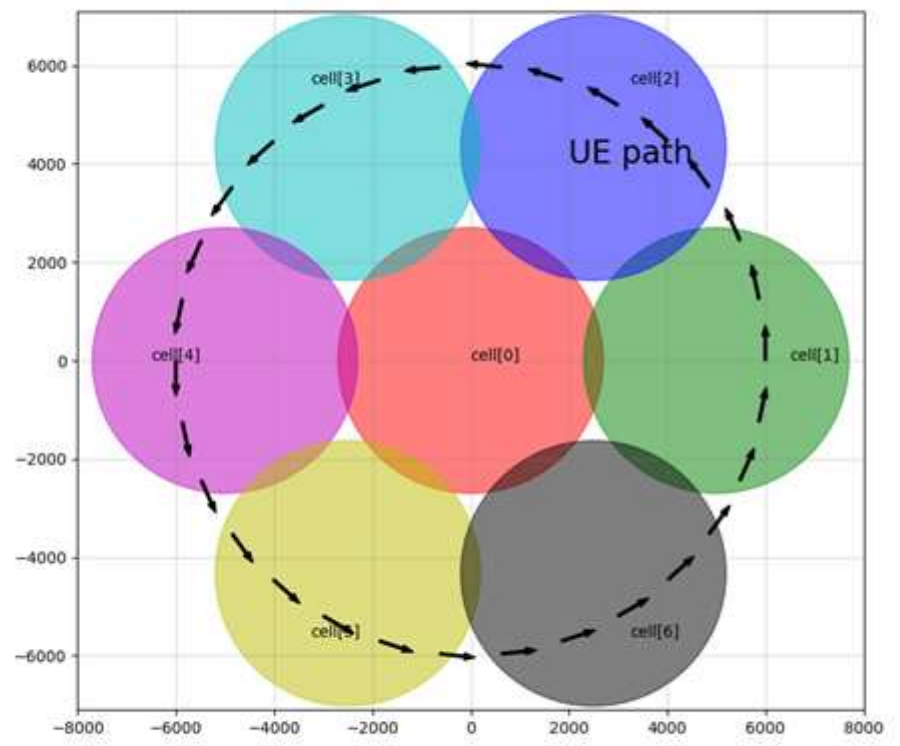


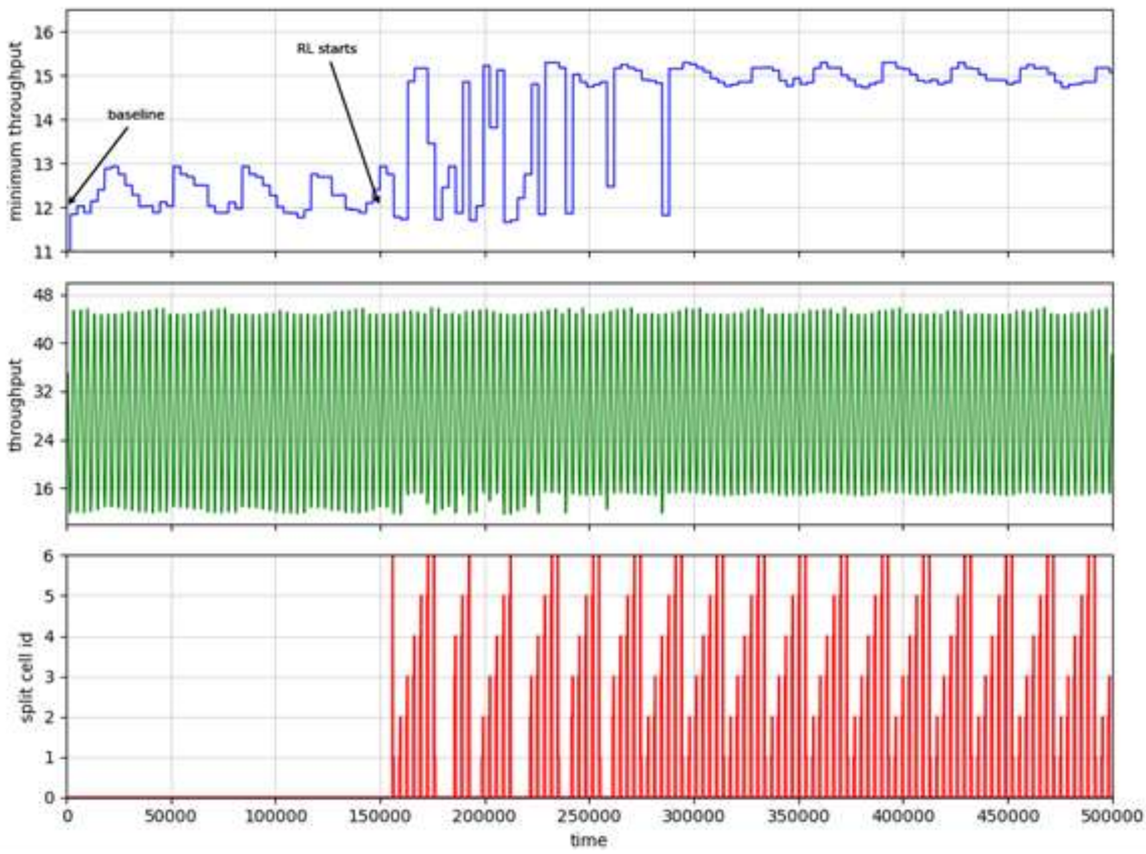
Figure 55 - Smart handover scenario with 7 Cells and 20 Users

Furthermore, Q-learning algorithm runs on the AIMM simulator. To use Q-learning requires pre-defining a set of states and a set of actions. It is always necessary to keep the number of states and the number of actions as small as possible, otherwise the learning process is too slow. We report here on an experimental heuristic which could be described as "Q-learning-assisted", rather than being pure Q-learning. It is built as follows.

An algorithm in the RIC detects cell-edge situations by monitoring RSRP reports sent to cells. A cell-edge situation is triggered by two RSRP reports being within a threshold of each other (typically 6dB), and both being above some specified minimum (typically -120dBm). Upon trigger, a state *(i,j)* is set, meaning that the UE current in cell *i* is on the boundary of cell *j*. The Q-learning agent is now triggered, causing it to consider whether a handover between cells *i* and *j* is to be made. After this decision, the reward is set to the difference in downlink throughput (that is, after handover, minus before handover). This reward is boosted by a factor of 10 if the cell *i* RSRP has been trending down, and cell *j* RSRP has been trending up. This automatically reduces the chance of ping-ponging. The proposed method also automatically performs load-balancing since any handover to a heavily loaded cell will give a low reward.

The Q-learner has the task of determining whether this action is beneficial or not. To get good performance, it is preferable to start the system "hot", and cool it gradually, that is, if the "raw" probability of choosing action *j* when in state *i* is *Qij*, then we use *exp(βQij)*, in which *β* is a parameter (the inverse temperature) which starts at 0 and increases over time. This makes the distribution flat at the start, and equivalent to "pick the biggest" after long times. Thus, after sufficiently long times, the system stops learning and behaviour becomes deterministic. The rate of this cooling process can be tuned by adjusting β.

Typical results are shown in Figure 56. Until time 10000, no handover at all is used, to demonstrate the very poor throughputs. Between times 10000 and 30000, a standard 4G MME algorithm is used. After time 30000, the MME is switched off and Q-learning takes over. We see on the top graph that throughput at least as good as the MME algorithm is achieved after about time 60000. The dark blue is the instantaneous throughput; the light blue is a smoothed value. The green curve shows the number of UEs attached to *cell [1]* (other cells are very similar) and demonstrates that the load never exceeds 5. The red curve shows the serving cell for *UE [0]*; it is only occasionally 0 (the cell in the centre of the hexagonal arrangement), which is as expected as the UE path stays away from *cell [0]*. These are very promising results, and future developments will look at making the heuristic yet smarter by using more explicitly the cell-load information along with the subband allocation technique that was developed and presented in the last deliverable. Attempts have also been made to find promising use case to include soft re-use (power reduction rather than complete switching off a subband).



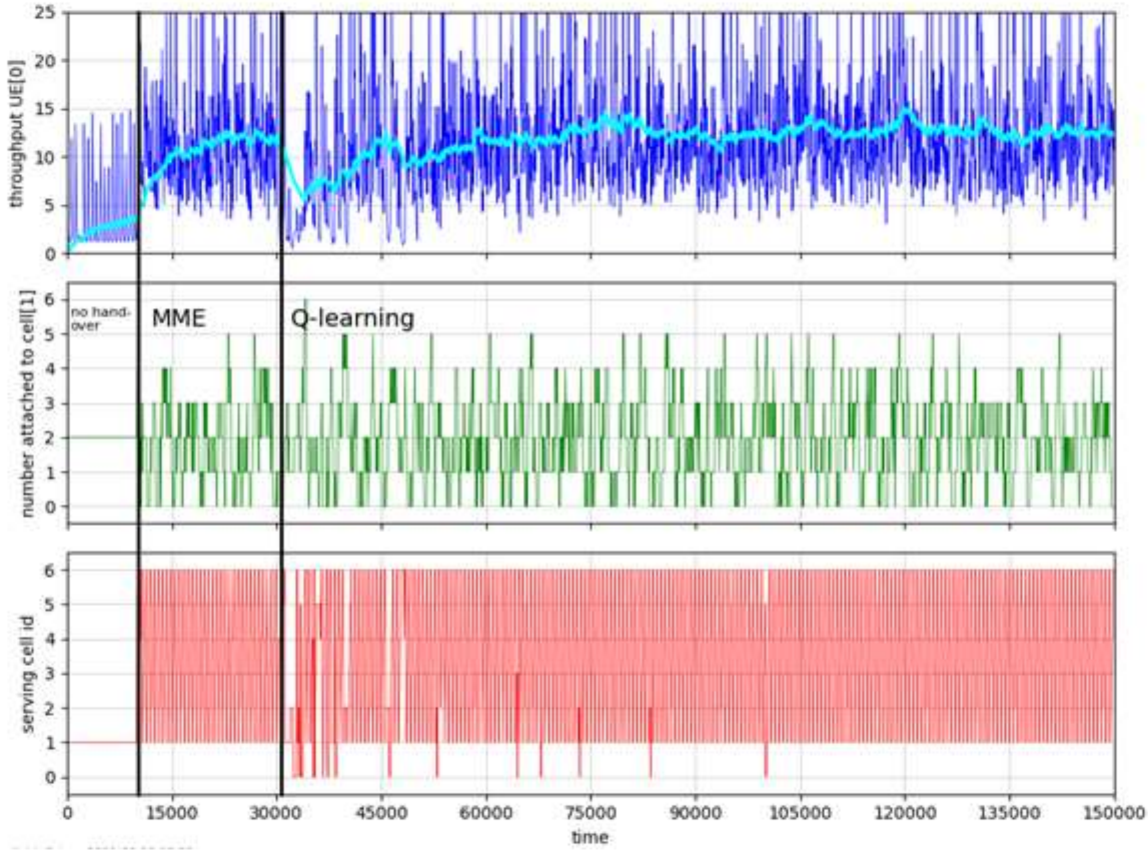


Figure 56 - Results from handover experiments

4.6.3 Conclusion

The deep RL approach has been proven to solve sophisticated problems even in partially observable environments. Since communication channels dynamically change over time, it is nearly impossible to formulate every communication channel between UEs and Radio Units (RU). Therefore, heuristic methods are being used to find near optimal solutions such as deep RL. While deep RL can provide such comprehensive solutions, its computational cost can be high. Especially when it is applied on a larger scale instead of single RU and connected UEs. Accumulated computation cost will be higher than desired. Beside the computational cost, it will be not feasible to train a RL model for each RU or DU individually. This is because each RU has unique environment and may need to be trained for its own environment.

4.7 Beam Selection

4.7.1 Introduction

There are several ways that Massive MIMO networks can address the issues of limited radio resources to meet user demands for service [4.34]. When located within a dense urban environment, for example, the multipath propagation within the environment tend to cause variations in the angles of arrival, signal strength and phase angles across the array. These variations, combined with appropriated precoding and detection schemes, allow for the use of aggressive spatial multiplexing, where multiple communications channels can share frequency and bandwidth resources at the same time. The large number of antenna elements also allow for the directing of energy toward much narrower regions compared with what is possible with smaller arrays. This beamforming is achieved by varying the antenna weights, that is the amplitude and phase angles, across the elements of the array.

The use of ML for the management and optimisation of programme parameters within Massive MIMO algorithms is currently an active area of research, and the interest has only increased thanks to growing use and investigations of disaggregated and Open Radio Access Network (O-RAN) systems, which provide for increased network softwarisation and the deployment of ML applications in near-real time and real time contexts and easier implementation through the use of dedicated ML components within the O-RAN standards for example the near real time RAN Intelligent Controller (RIC) and the non-real-time RIC [4.35].

There is the option to deploy Massive MIMO systems within a variety of contexts that each have different criteria for optimal performance aligning with the related 3GPP standards. This section focusses specifically on the questions of coverage as it relates to the synchronisation process within the 5G standards. This process is discussed in more detail in the following section and is dependent upon how beamforming is configured at the BS.

The application of ML to the optimisation of beamforming parameters within Massive MIMO has been approached in several different ways. For example, Wang et al [4.36] propose a low-complexity, sub-optimal beam allocation algorithm based on fixed beams formed using the Butler method [4.37]. This is a combinatorial optimisation problem, but with the constraint that each beam serves only one user and where the aim is to maximise the sum data rate of the system.

Shafin et al [4.38] take a different approach to applying ML to beam optimisation, by making the beamwidth and angles of the beams the parameters to be optimised. In other words, a set of beam weights are determined, again by formulating an optimisation problem. This paper aims to maximise the Signal to interference plus noise ratio (SINR) at the user by determining the most appropriate beam parameters.

This section describes an approach for beam selection that is based upon optimising the coverage during the synchronisation process for a set of users for whom the spatial distribution is known. The approach involves the adjustment of the beamforming parameters by choosing beams from a defined set according to a defined criterion.

4.7.2 Beamforming in Massive MIMO

Synchronisation with the RAN, whilst important in previous generations of mobile networks, is especially important within 5G, owing to the time-critical nature of many of the technologies within such networks and the increased number of devices operating with these requirements. For example, TDD technology requires both the BS and the UEs to transmit using the same bandwidth resources, thus making the timing critical at both ends of the link to avoid interference [4.39]. Likewise, the use of multiple beams within Massive MIMO technology that are dynamic and not covering the same area, with the same boundaries at the same time, necessitate the use of correct timing at both the UE and the BS end of the link. The synchronisation process is a method within the 5G standards, preceding the transfer of data, that allows the UEs to establish the timing necessary to connect to the network [4.40].

The synchronisation process allows the UE to detect the time at which a radio frame begins and the time at which an OFDM symbol begins, a process that is achieved by the transmission of a Synchronisation Signal Block (SS Block). This forms part of the process of downlink synchronisation. Uplink synchronisation is the process that allows the UE to obtain timing data for when it should send uplink data. This uplink synchronisation forms part of the random-access channel (RACH) process. However, downlink synchronisation remains the focus of this paper because of the relatively cohesive way in which it can be mapped onto the process of beamforming described in the introduction.

The downlink synchronisation process consists of the BS sending several synchronisation signal (SS) bursts, each consisting of a block of data separated by defined timing intervals. The number of SS bursts is currently defined as not being permitted to exceed certain numbers depending on the frequencies being used, with 4 below 3GHz, 8 up to 6GHz and 64 above that. The use of 8 blocks is considered here, as mid-3GHz is typical for 5G BSs; however, it would not be complicated to adapt this to lower numbers of blocks or to different frequency ranges if necessary.

The separation of the SS bursts in 5G are defined in terms of orthogonal frequency-division multiplexing (OFDM) symbols, and each SS Burst is a mapping across four symbols, each of which are mapped across 240 subcarriers, although not all these subcarriers are used for the transfer of information. The first symbol contains the Primary Synchronisation Signal (PSS), the second and fourth contain the Physical Broadcast Channel (PBCH) and the third contains both the PBCH and the Secondary Synchronisation Signal (SSS).

Each of the SS bursts may be transmitted while the antenna array at the BS is configured to form the beam according to a pre-determined specification. The effect of this is that, during the synchronisation process, the beam can be made to sweep across a geographic region. The UEs, as part of the process, report back to the BS the reference signal received power (RSRP) values as an indication of signal quality within the RACH preamble. This allows for a beam to be identified as providing the highest quality link between the BS and the UE. This information, with the process initiated during synchronisation, forms part of the overall beam management framework for 5G communications. Once the beam has been established as best serving a certain set of UEs, the beam refinement process, where beams can be made narrower and more specifically directed, can begin to allow for further data transfer. However, beam refinement is beyond the scope of this research.

This section now focusses on synchronisation and beam selection from the perspective of coverage. In other words, how can the beams associated with the SS Blocks be best configured to provide the best coverage from the perspective of a BS for a given distribution of users and environment. It is possible to ask this question because the characteristics of the beams (in terms of width, azimuth, elevation, and electrical tilt) are not defined and there may therefore be some discretion for the operator to define these, subject to operation considerations.

The method for addressing this problem is discussed further in later sections, however the following assumptions are used as the foundation to this research:

1. The best set of beams for a given BS, operating within an environment and with a defined user distribution, is the one that provides the most even coverage to the users. In other words, each beam serves an equal number of users.
2. A single beam can serve multiple users.
3. There is no maximum number of users that can be served by a BS. This assumption is not realistic but is reasonable for the studies here as only typical numbers and distributions of users are considered.

Use is made of a MATLAB simulation from the 5G toolbox as the basis for obtaining the RSRP values from users. 5G synchronisation allows for beamforming at both ends of the link, however only beamforming at the BS is considered here. The frequency for the simulation is 3.5GHz, and thus, following the standards, eight synchronisation signal blocks are transmitted, each with an associated beam from the beam set. A user distribution is first defined around a central point with users distributed between 100 and 1000 metres. The area for sweeping is limited to within a typical tri-sectored region, not accounting for the edges of the beams and the sidelobes. The full waveform and grid for the synchronisation signal bursts is generated and transmitted through a channel. Initially a Rayleigh fading channel. The RSRP values are then reported back from the UEs, with the highest RSRP value corresponding to the selected beams.

The process of simulating the propagation of the waveform through the channel is computationally slow and overly complex for investigating synchronisation coverage. Therefore, a more simplified model is used that greatly increases the speed of the simulations.

4.7.3 Overview

In general terms, the aims of the studies described in this section are, firstly, to investigate the application of Machine Learning to the process of broadcast beam selection within 5G. The second aim is to investigate how the use of user location data, obtained by processing data available to BT as a Mobile Network Operator, could be used to inform the beam selection process and whether such data could assist in the design or implementation of Machine Learning systems used in such a context.

Much of the research in AI for beam selection in 5G is concerned with either the creation of the beam patterns themselves, or with how different parameters (SINR, for example) can be used to learn which beams should be selected from a set of available beams. However, the investigations described here take a different approach, where sets of different antenna weights (with each weight representing a potential broadcast beam) are compared in terms of their ability to serve a set of users. The reason for taking this approach is because of the interest in using beam selection optimisation to maximise coverage within a cell area, whilst reducing the presence of beams that may not be serving many users in comparison with other beams. It is assumed that an optimal selection of beam weight would be the ones that generate a set of beams that serve an equal number of users each.

It should be noted that, in this section, a beam refers to a pattern that is generated by applying weights to an antenna array, while a ‘set of beam patterns’ refers to a class of beams. The aim of this use case, as interpreted here, is to identify a set of beam patterns for use within a type of environment, not to determine the individual beam patterns themselves.

So far, the work on this use case has used simulations of random users placed around a base station. Several typical beam sweeping patterns have been selected, which are discussed in more detail in the next section, and the 5G synchronisation signal simulated through a channel. As part of the synchronisation process, the RSRP values are obtained from the UEs with, by default, the beam with the maximum value being selected as the one associated with the user. This information is then used to select the beams for the transfer of data, but this step is not currently being considered as part of this use case, which has an emphasis more on the coverage of the broadcast beams. This simulation process is repeated for each set of antenna weights, providing a selected beam for each set.

The results from the simulation are formatted to create training data for supervised learning in the form of feature vectors. The aim of such supervised learning is to attempt to use Machine Learning to determine which set of beam patterns are the best given a certain type of environment. The reason why it will be possible to make this determination about a certain type of environment is that the random user data will be replaced by realistic location data obtained from the network. The location data represents different forms of user distributions (e.g., for urban and rural environments) and can thus be used to make determinations about suitable beam patterns. The purpose of investigating supervised learning methods is to determine the feasibility of simplifying the process of selecting sets of beam patterns, since making the choice through simulation can be computationally expensive and time consuming.

This summary of research begins with an overview of the simulation process. The simulation process, including the running of propagation models, is time consuming in its current form. Therefore, a discussion is presented of some of the compromises that are required to run the simulations in a time effective manner. A method of providing approximate RSRP measurements simply to illustrate the process of applying Machine Learning to beam selection is discussed. This is followed by an explanation of the generated feature vectors and their categorisation using standard Machine Learning methods. An explanation of available location data is then presented along with a basic example of the application of this data. The next section then introduces ideas for how such data may be applied in real-time as the research progresses. The conclusion discusses some of the limitations of the simulation framework presented so far and highlights areas of importance to address within the next stages.

4.7.4 Explanation of Simulation

This section presents an overview of the simulation and its limitations. The MATLAB 5G simulation toolbox combined with the phased array antenna toolbox has been used to generate hypothetical (but realistic) beam pattern sets. Then, for each individual beam within the beam set, an RSRP value is obtained for each user, the combination of which are used to determine the selected beam from within the set. The data of selected beams associated with each set can then be used to determine the selected beam set for a certain configuration of users and an environment. Note that an environment is described partly through user distribution and partly in relation to the propagation environment. The general process of simulation and beam selection is described here, followed by an amended selection process to account for the limitations related to the time it takes to run the simulation on available computer equipment.

A user distribution is first defined in polar coordinates, with each UE represented by a point in a 2D plane. The simulation can run in 3D, but for the moment only two dimensions are considered to simplify the process. The user distribution is initially random, to be update later with more realistic distributions. The simulation parameters are then set with standard physical propagation settings for free space. An 8x8 linear array is selected for the simulation, a typical configuration of Massive MIMO antennas. The elements are modelled as isotropic antennas, again initially to simplify the process. The base station array is placed at the centre of the study area (i.e., at theta and phi equal to zero).

The frequency for the simulation is 3.5GHz, and thus, following the standards, eight synchronisation signal blocks are transmitted, each with an associated beam from the beam set.

The waveform and grid for the transmitted synchronisation burst are then generated. These bursts are associate with beams, obtained from antenna weights associated with beam sweeping vectors. The vectors provide discreet points around a circle such that the energy points at a specific direction over a range of angles, changing for each burst. The azimuth sweep is set for a sweep of 120 degrees, which would be typical for a tri-sectored base station, although some of the beam sets are set to sweep over a much narrower range of angles. The actual beam angles are set according to a dictionary, initially of five sets of angles, the first three of which begin and then narrow the 120-degree sweep. The fourth set includes angles at the extreme ends of the sweep and the fifth in the centre.

A standard Rayleigh channel model based upon a gaussian fading channel is used for the channel model through which the bursts are transmitted. The process for obtaining the RSRP values for each user is then followed. This whole process is repeated for each user and then for each beam set, with the data recorded in each case. For each set, the chosen beam is simply the beam corresponding to the highest RSRP value, with the chosen beam stored in a matrix with the rows representing each user, and five columns representing each beam set. The beam pattern set is then selected by analysing the distribution of beams. In other words, for each set of selected beams, take the beam with the minimum number of users, and subtract this value from the beam with the maximum number of users. The set with the minimum value then represents the set with the most even distribution of users and thus, from the initial assumption about the preferable beam set, this set is selected.

4.7.5 RSRP Curves

The use of simplified propagation models is justified both in addressing the issue of complex 5G channel models and in allowing for a clearer formal description of the problem being investigated, a description that is presented in the next section. However, when implementing such a system in an actual network it is likely to be necessary to use a more sophisticated channel model, since a Rayleigh model may not occur often in environments which may be encountered in a practical network. Many 5G channel models are statistical in nature and the following process results in models that are essentially look-up tables. However, it could be argued that the large number of users being considered, and the level of granularity make a determined average value at each point within the model acceptable. The level of granularity is related to the dimensions of the environment, and in this case all the environments being considered are outdoors and based on macro-cells with beams that are wide enough to cover several UEs at a time. Thus, less precision is required compared with indoor environments or in situations where beams are narrow enough to serve individual UEs within dense user environments.

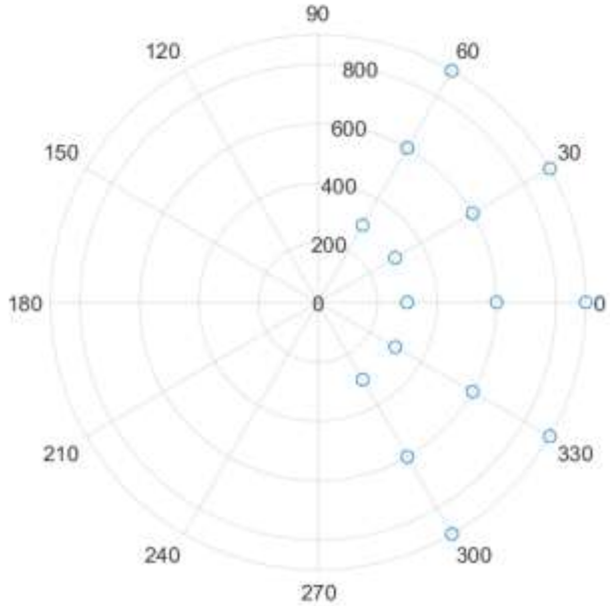


Figure 57 - User distributions around BS at centre of graph. The blue circles represent the location where the RSRP value is determined. The concentric circles represent the distance from the BS in metres.

The aim of this approach is to create an RSRP surface that can represent a typically expected RSRP value for each of the possible beams. The users are placed evenly around a BS as shown in Figure 57. Only a third of the total area is occupied with users to simulate a typical tri-sector BS configuration. The range for these examples is set out to 1km, although another value could have been chosen. Additionally, an area near to the BS is excluded to avoid near-field effects and complications with overlapping beams. An assumption of this model is that there are no users within the exclusion zone. The waveform model for 5G is then run for each of the beams. On each occasion each user reports back its measured RSRP value for the beam. These reported RSRP values at specific points, correspond to points within a two-dimensional RSRP surface (Figure 58). This surface is then extended to cover the entire area of interest from the point of view of the propagation environment. The RSRP surface is also referred to as a ‘propagation grid’ when used in the context of the matrix containing the RSRP values for each beam.

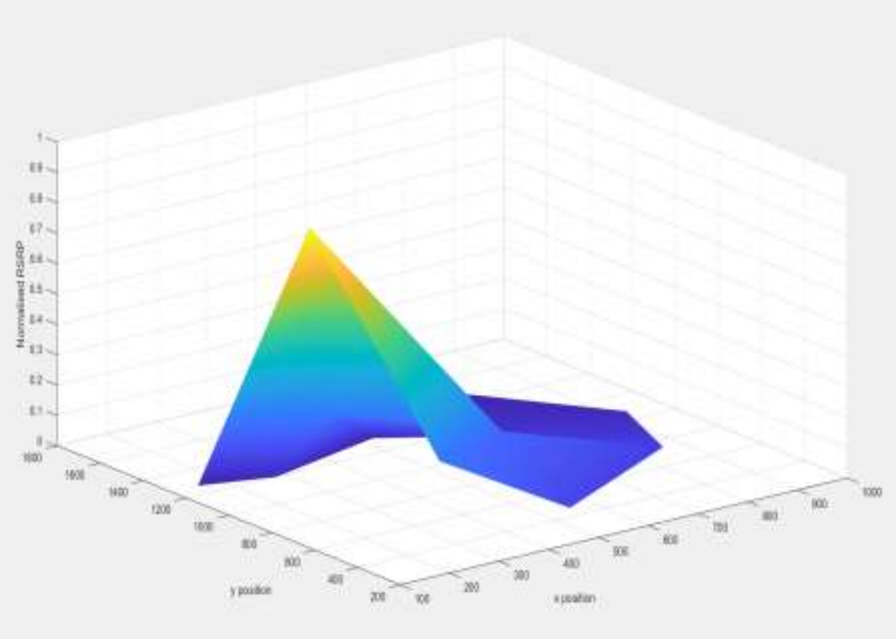


Figure 58 - Example of a generated 'normalised RSRP surface' for one beam from the described methods. Each beam will have its own surface, which can be considered as almost synonymous with the beam.

The main outputs from the simulation are:

- The RSRP measurements for the combination of receive and transmit beams, presented as an MxN matrix (with M and N being the number of BS and UE beams). The maximum value in the matrix corresponds to the chosen beam.
- The selected beam IDs (in the range 1…M and 1…N) of the selected beams.
- The BS array weightings, as a vector with a number of elements corresponding to the number of antenna elements.

4.7.6 Feature Vectors

It is possible, once the beam sets have been selected, to create feature vectors for categorisation. The idea here is to have sets of user positions that vary from one to another, but which represent a typical type of environment, for example a busy shopping location at a certain time of day. There are several possibilities for the format of the vectors, but the format that is the most convenient is a format that contains the number of the chosen beam set and the coordinates of the users within the study area. It may be possible to reduce these vectors further by clustering the users when real data is considered, as discussed later. Formatting the data into feature vectors allows for the testing of supervised learning algorithms to find boundary lines between the vectors, so that it is more immediately apparent which areas correspond to which beam sets.

4.7.7 Use of Real Data

It is possible to run the simulations described using something that resembles more a realistic user distribution and environment in a limited sense without much further modification. The coverage maps, location data and base station information available at BT provides information regarding broad angles of departure from the base station and the general direction for where users would be clustered. Heat map data is also available to refine further the knowledge regarding user clusters. Given this, it is possible to set up a distribution of users by sectorising the plot, with more users generated within the areas of a high concentration of users than elsewhere.

However, in future work, it would be possible to be more precise about user locations by extracting data from the heat maps and using this as the input to the simulation, rather than just obtaining a general impression from the visual data. This would also allow for a greater number of simulations to be run, as it would be easier to automate the process, thus creating a much larger data set for testing.

4.7.8 Real-time Implementations

Within 5G networks, beams are selected, both for synchronisation and for data transfer, by the obtained reports from the UEs during synchronisation and beam refinement. It may be possible to implement Machine Learning to analyze the results of these processes and propose modification of the beam set based on the results. There are potential benefits from this, in that, if it has already been possible to learn the optimal beam set for the environment, it may, at least in principle, be possible to skip some of the stages within beam selection.

4.7.9 Scope

The aim of this section is to discuss some possibilities for approaching the beam optimisation problem as part of this research. It also summarises the data requirements for the project, provides a description of the problem to be solved, and describes some of the resources available and their functionality.

It has been necessary to determine whether a specific method for beam allocation is to be assumed. There are various possibilities when it comes to investigating this area of research such as looking at traffic patterns, applying existing ML methods or attempting to learn an optimal grid of beams.

Much of the research in applying ML to Massive MIMO beamforming is concerned with using ML to learn the required beam based on some parameter (such as SINR or receive power). These methods assume a specific ‘grid of beams’ that cover the cell, with the aim being to pick out the beam selected during the synchronisation process.

A second area of interest is to use ML to determine the best parameters of the beams themselves (beamwidth, tilts, elevation, and azimuth).

These two areas of interest could be linked in the sense that several pre-defined grids could be considered, some with narrow and some with wide beams, for example. Then, the ML method is applied with different grids, the results of which are combined and processed to somehow determine which general beam configuration should be used. It may be, for example, that a configuration consisting of several narrow beams in one direction with wider beams in another direction could be optimal in many urban environments, for example when a shopping in centre is in a certain direction relative to the BS. It would be necessary to have a way to represent numerically how good a certain configuration would be, perhaps related to the number of unused beams within a configuration.

Advantages of using a ML approach include the reduction of time and costs in the case of running physical experiments and the possible reduction of complexity compared with running deterministic simulations.

The aim of this project is to make use of realistic data regarding the distribution of mobile users.

4.7.10 Prior Work

4.7.10.1 Problem Statement 1

Junyuan Wang *et al* [4.41] have formalised a beam selection problem that can be applied to ML in a way that focusses on learning selected beams from a ‘grid of beams’ (i.e., a collection of beams covering an entire cell) based on some chosen parameter.

The assumptions of the problem statement described by the authors are that the users are all equipped with a single antenna and are uniformly distributed within a circular cell area. The array is linear with isotropic elements and half wavelength spacing with a collection of beams formed by the Butler method.

$$A_n(\theta) = \frac{\sin\left(0.5N\pi cos\theta - \beta_n\right)}{Nsin(0.5\pi cos\theta - \frac{1}{N}\beta_n)} \tag{35}$$

Where:

$$\beta_n = \left(-\frac{N+1}{2} + n\right)\pi. \tag{36}$$

Here N is the number of beams, θ is the AoD, and n is the integer index of the beam. The users are distributed around the linear array and served by the beams as shown in Figure 59.

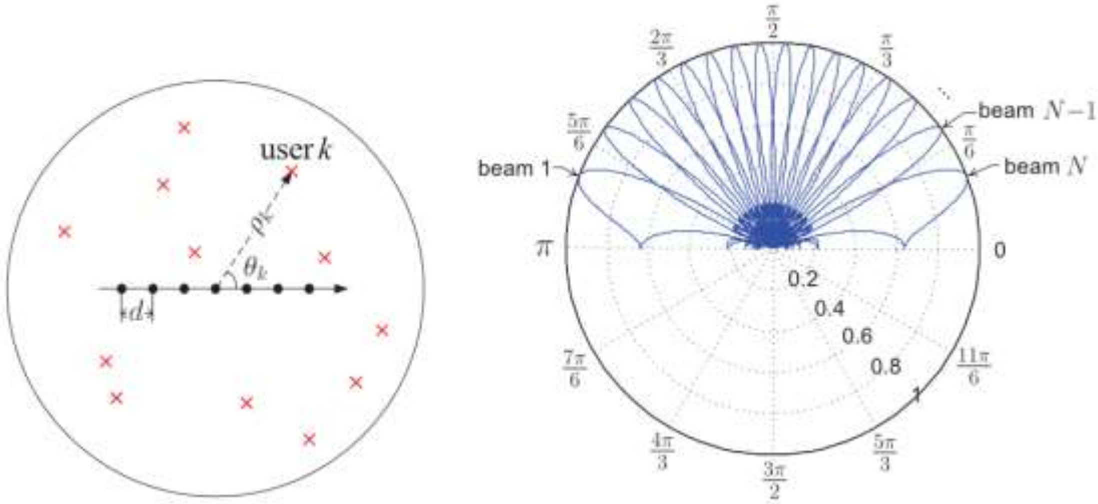


Figure 59 - Distribution of users around the linear array with example beam patterns

The receive power is determined for each user k according to a standard path loss model

$$P_k = \sum_{n=1}^{N} 1_{k,n} p_n D_n(\theta_k) \rho_k^{-\alpha} \tag{37}$$

Here $1_{k,n}$ is an indicator function indicating the chosen beam (i.e., 1 where beam *n* is selected for user *k* and 0 otherwise), p_n is the allocated beam transmit power, $D_n(\theta_k)$ is the relative directivity of beam n, ρ_k is the distance from the cell centre to user *k* and α is the path loss exponent. A further assumption of the model is that each beam can serve at most one user and that the total transmit power is fixed, with the power allocated to each beam being given by

$$p_n = \begin{cases} \frac{P_t}{N_s}, if \sum_{k=1}^{K} 1_{k,n} = 1 \\ 0, if \sum_{k=1}^{K} 1_{k,n} = 0 \end{cases} \tag{38}$$

where N_s is the total number of allocated beams. The assumed achievable data rate is

$$R_k = log_2\left(1 + \frac{P_k}{\sigma_0^2 + I_k}\right) \tag{39}$$

Here σ_0^2 is the variance of the noise and I_k is the inter-beam interference

$$I_k = \sum_{j=1, j\neq k}^{K} \sum_{n=1}^{N} 1_{j,n} p_n D_n(\theta_k) p_k^{-\alpha} \tag{40}$$

The aim is then to find

$$\max_{\{1_{k,n}\}\forall k, \forall n} \sum_{k=1}^{K} R_k \tag{41}$$

This problem is slow to solve directly and forms the basis of some research in ML and MIMO beam selection, as discussed later.

4.7.10.2 Problem Statement 2

The second problem statement approaches the problem differently, by making the azimuth and elevation beam widths (in addition to the electrical tilt setting) the parameters to be learned through ML techniques. This effectively means that the beams themselves and the configuration of the grid of beams covering the cell are learned, not which beams within the grid are activated and for which users.

Shafin *et al.* [4.38] formulate this approach to beam optimisation by considering a problem of determining a weight vector (which is applied at the BS array to define how the beam is steered towards individual users). The authors consider the received signal vector for each user k (where the notation m refers to the sector and M) as

Problem Statement 2:

The second problem statement approaches the problem differently, by making the azimuth and elevation beam widths (in addition to the electrical tilt setting) the parameters to be learned through ML techniques. This effectively means that the beams themselves and the configuration of the grid of beams covering the cell are learned, not which beams within the grid are activated and for which users.

Shafin *et al.* [4.38] formulate this approach to beam optimisation by considering a problem of determining a weight vector (which is applied at the BS array to define how the beam is steered towards individual users). The authors consider the received signal vector for each user k (where the notation m refers to the sector and M) as

$$y_k = \boldsymbol{h}_{m,k}^T \boldsymbol{f}_m s_m + \sum_{\substack{m'=1 \\ m' \neq m}}^{M} \boldsymbol{h}_{m',k}^T \boldsymbol{f}_{m'} s_{m'} + z_{\boldsymbol{k}} \tag{42}$$

Here $\boldsymbol{h}_{m,k}$ is the channel vector between the mth sector and the kth user, s_m is the transmitted signal and $\boldsymbol{f}_m$ is the precoding vector applied to the mth sector. The final term relates to the noise. The aim is to learn the optimal beam pattern for each sector. The interference between sectors must also be considered as part of the selection process.

There is, for each BS with its sectors, a predefined class of *J* possible weighting vectors for the antenna array

$$J: \{\boldsymbol{j}^1, \boldsymbol{j}^2, \dots, \boldsymbol{j}^J\} \tag{43}$$

Here each of the weightings corresponds to a beam pattern with a specified azimuth, elevation, and tilt. The chosen weight corresponds to the sector's precoding vector f. The weight vector is chosen based on the distribution of users.

The choice of weight vector is made based on the following process (noting that the aim of any ML scheme related to this general approach is to learn the weight vectors that result from the described process): firstly, the SINR rate for each user is determined for each of the available beam configurations. A threshold SINR value is selected that will be used as the baseline for determining if a user can be said to be connected to the BS or not. The aim is then to compute the set of beams for each sector that maximise the total number of connected users, i.e.

$$\max_{\boldsymbol{f}_1, \boldsymbol{f}_2, \dots, \boldsymbol{f}_M} \sum_{k=1}^{K} 1_{SINR_k > T} \tag{44}$$

Here T is the threshold SINR level and '1' is the indicator function. In other words, this is the total number of connected users that occurs for some set of beam configurations, with one beam pattern associated to each sector. Finding this set of beams is made more complicated by the fact that there will likely be inter-sector interference, a factor that could affect the number of connected users that is achievable.

It should be noted that the criteria for beam selection is very different compared with the first formulation, because here the aim is to obtain a beam that will serve the largest possible class of users, but in the first formulation each beam served only one user.

This formulation of the problem would seem to be closer to what is described in the Autonomous Industrial Mobile Manipulators (AIMM) use cases' specification; however, it may be that this formulation of the problem could be combined with the first formulation. This could work by determining which grid of beam configurations are available to the networks, simulating the channel with user distributions, and then attempting to learn the beams selected through available parameters, repeating this process for each of the possible grid of beams configurations. It would then be necessary to determine which grid should be chosen according to some parameter. Perhaps, for example, the chosen grid would be the one with the fewest number of unused beams.

Determining the grid in this way could help in providing an optimal grid for different environments, as described in the introduction.

4.7.11 Recent Research

There seem to be fewer research outputs related directly to the topic of using ML for SS Block Broadcast Beam optimisation compared with other areas of MIMO research. Much of the relevant literature is from conference proceedings and often assumes the use of mmWave frequencies. There may be opportunities to investigate some of these methods but in the context of frequencies more relevant to the mobile Radio Access Network, as well as to use real data, such as those provided by the ‘average number of connected users’ metric, to provide more realistic data for the development of ML algorithms.

The emphasis of this work is not to attempt to learn which beams are selected because of the synchronisation process from a specified grid of beams. Rather it will be to identify the best beam patterns to use from a set of defined possibilities. This involves identifying a figure of merit to determine which beam pattern should be used. The initial tasks are as follows:

- Identify a selection of beam patterns. The first options will likely be patterns formed from different numbers of active antenna elements, which will correspond to beams of different widths and different total number of main lobes within the beam patterns. The next options could be to rotate the selected beam patterns in various directions. Depending on the metric chosen to identify the favoured beam pattern, it may be possible to infer when a certain azimuth range can be excluded.
- Obtain and pre-process location data of users from the mobile network. These data will be in two dimensions at first, only in the azimuth plane, which is more relevant in situations without high rise buildings. It will be necessary to consider exactly what form these data will take and whether they will include information about buildings within the data or whether the position of the users is presented in terms of radio distance, rather than physical location.
- Modify code so that it can run the simulations with the output required. Determine whether the RSRP measurements and antenna weights or necessary, or whether only the selected beams are necessary.

Run the simulation and obtain results.

4.7.12 Mathematical Description

The generation of the normalised RSRP surfaces allows for the problem to be expressed more precisely as a combinatorial optimisation problem. This formulation is developed with attention specifically to assumption 1 in the opening section, that is to provide even coverage of broadcast synchronisation beams for a given user distribution and environment.

Within a region covered by a base station (i.e. a gNodeB) m user locations (each defined with (x,y) coordinates). It is assumed that a sector is known that corresponds to the coverage area.

The matrix $\boldsymbol{X}$ is $n \times m$ where n refers to the number of possible beams (each with their computed normalised RSRP curves). The quantity n is determined simply by the beams that are being considered, which could be the number of possible beams given the hardware constraints or those that could possibly be permissible given operation considerations. It may also be defined by the requirements of a specific type of study. For example, if it was necessary to test different types of beam configurations, one might choose a set of beams of one type and a set of beams of a different type, with each set being the same size. The m columns of $\boldsymbol{X}$ refer to the RSRP values at the user locations.

The $r \times m$ matrix $\boldsymbol{Y}$ is made up of r rows from $\boldsymbol{X}$, where r is the maximum number of allowed beams (often eight in our examples and $r < m$), define v_j as the position of the maximum value in column j, that is the row of $\boldsymbol{Y}$ that contains the maximum value.

Define z_d as the number of times that d appears in v where d=1,…,r. Then find the r rows of $\boldsymbol{X}$ forming the matrix $\boldsymbol{Y}$ that minimise the difference between the maximum number contained within z and the minimum number.

$$\max(z) - \min(z) \tag{45}$$

This difference is referred to as the ‘beam distribution.’ The beams that are selected to form the rows in $\boldsymbol{Y}$ provide for the most even coverage amongst the user distribution, as described earlier. Intuitively one is presented with a matrix where the rows correspond to the beams and the columns to the RSRP values of specific users. A vector is then created to show the list of beams that provide the best coverage for each user, and the aim is to find the set of beams for which each beam appears in the list with the same frequency as the others.

4.7.13 Available Data

Data Requirements

Location data for the mobile users are a requirement for this research. These could be entirely empirical in nature or a combination of empirical and a statistical data. Data that are generated entirely based on statistical models may not be appropriate, as part of the originality of this research will be based on the use of realistic traffic and locations. Additional information such as performance data could also be useful, though this may be a secondary concern as the first point of interest is the user distributions themselves. Data such as practical RSRP data could be useful for the verification of the accuracy of the distributions used (if these are not based on precise known locations). For example, it may be possible analyse the range of RSRP values present in the practical network and see if these match the results that are suggested in the simulations, although this may be complicated because of the varying configurations of the beams, which may not match between the practical and simulated environments.

Milton Keynes simulation

The data that are currently available (i.e. data that are simple to access within the research group at present) include the following:

Location data for BSs and the surrounding environment are available (Figure 60). The database also includes coverage maps, which will represent the regions that will form part of the analysis of the beam selection simulations.

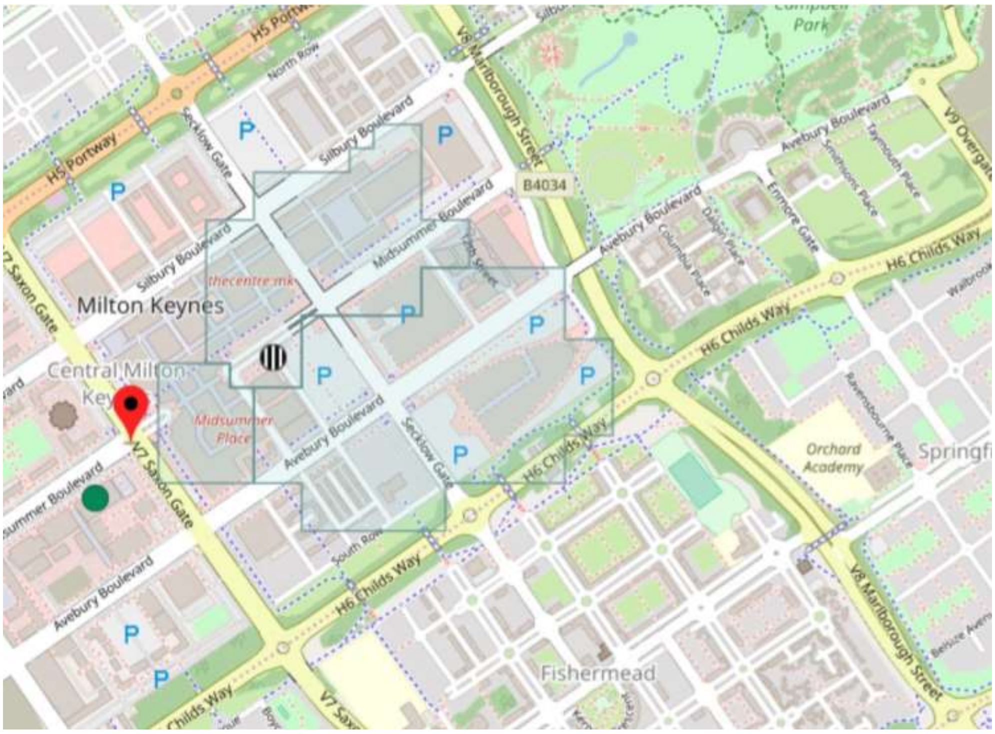


Figure 60 - Milton Keynes BS coverage data. The shaded areas represent the coverage for each sector.

A key type of data that could prove very useful and is readily available is the average number of connected user data. As this provides an average value for the number of mobile devices that are connected at any time, this effectively provides the number of users that need to be included in any simulation.

In addition to these network data, there are some traffic data that has been provided by local councils consisting of origin and destination matrices. This could be useful in refining (and possibly simulating) the movement of traffic, although this project will likely focus firstly on beam configurations that do not change too often, so this could be of limited use at first. It is possible that other public domain data could also be useful, as many councils provide these sorts of data in various formats.

Minimisation of Drive Test (MDT) data are also available from the network and are advantageous in that they contain the exact global positioning system (GPS) coordinates for each user (Figure 61). They also contain altitude data, which could be useful if the proposed framework is extended into three dimensions. They also include both RSRP and reference signal received quality (RSRQ) data, which will be potentially useful for the verification of the simulation.

A major disadvantage of MDT data is that not all UEs provide them, as users must give their permission for these data to be transmitted to the network. Additionally, not all mobile operating systems support the collection of these data. It is likely that the availability of these data will only reduce in the future. Also, no information about the features of the geographic environment are provided within the MDT data, that is there is no information about buildings or direct information about the terrain apart from what can be inferred from the GPS coordinates.

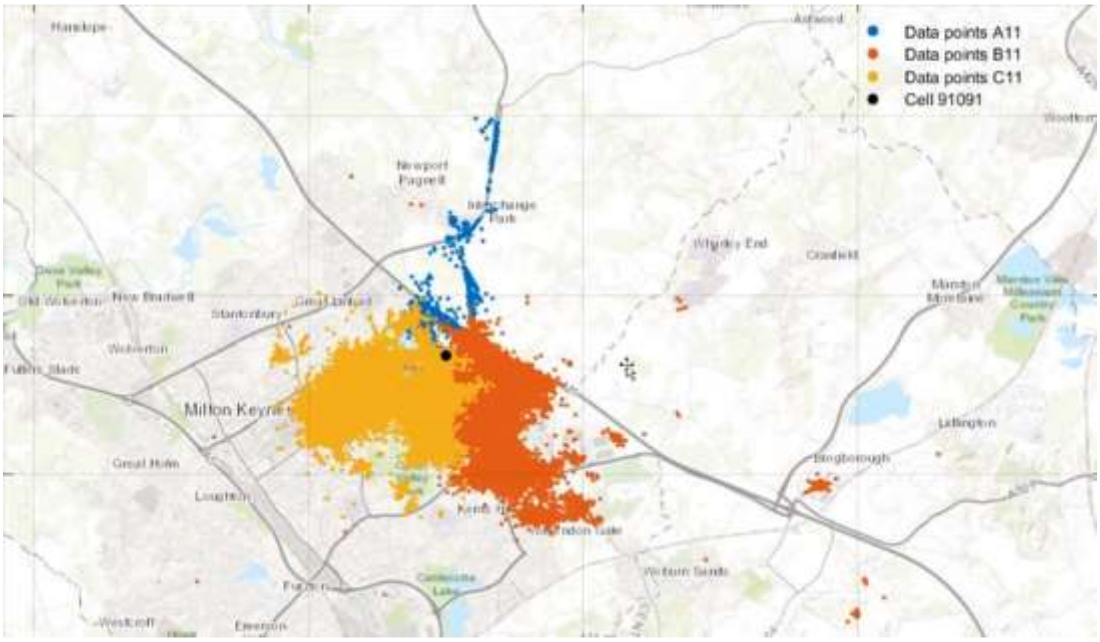


Figure 61- Example location data from MDT (courtesy of Shipra Kapoor)

4.7.14 Approaches to finding a solution

The most obvious approach to solving the even coverage problem described in the previous section is to use an exhaustive search method. This is achieved by computing the RSRP values from the matrices representing the RSRP surfaces for each of the possible beam combinations from a defined set of beams. This approach, however, is potentially time-consuming. For example, finding eight beams from a set of only 15 require 6435 iterations of possible beam combinations. As shown in this section, this many iterations may not be necessary as sometimes there can be many sets of possible beams that yield very similar distributions. This can be seen in the following example.

Here, 15 beams are selected to be investigated. These are not chosen to be related to a specific example, but rather to view the types of results that can be expected. The beams are chosen to sweep a sector of 120 degrees to the west of the BS. The first eight beams are separated so that the main beam sweeps at equal intervals between -60 and 60 degrees. The final seven are the first beams from the set that sweep at equal intervals between -50 and 50 degrees. The simulation is run according to the RSRP surface method described above, obtaining the normalised RSRP values for each of the users present.

An exhaustive search method was used to provide a baseline for comparisons with other methods for finding optimal beam configurations and to analyse the features of the variations in user distributions over beams. The 'distribution over beams' is the difference described in the equation, which is the difference between the number of users that are served by the beam with the highest number of users and by the beam with the lowest number of users.

The results of the exhaustive search are shown in Figure 62, with the horizontal access represented the combination of eight beams from the set of fifteen. The lower the value on the y-axis, the better the combination of beams in terms of even coverage. It can be seen immediately that the range of user distribution over beams is very large, especially considering that the number of users in this example is limited to 100. It can also be observed that, while the user distributions never reach zero, which would represent the most idea scenario for even coverage, the lowest values of around 11 and 12 are revisited several times, suggesting that computing the RSRP values for all combinations may not be necessary.

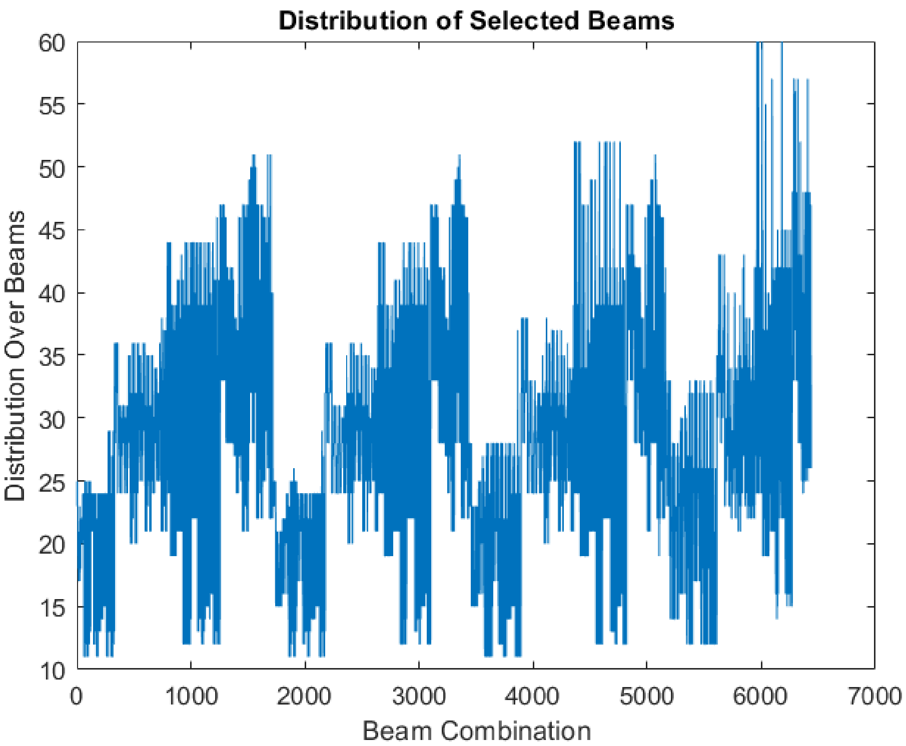


Figure 62 - Variation of the distribution of users over the selected beams, as described in the text, for all possible combinations of a set of 15 beams.

4.7.15 Experimental Results

The problem of finding a minimum value from the beam selections is addressed in this section. The function is discrete, therefore making this a problem in integer linear programming. Unfortunately, this means that many of the standard optimisation algorithms are not appropriate for finding a solution, although simulated annealing (SA), a standard method in optimisation, offers a possibility, as it has often been used for optimisation problems involving discreet data [4.42]. It should be noted that it may not be necessary to find an actual minimum value, as an approximate minimum value may suffice for practical purposes.

SA, first introduced by Kirkpatrick *et al.* [4.43] is similar to other stochastic search methods where a random point is selected, followed by another point which is rejected if it offers a worse solution or accepted if it offers a better solution, with the process continuing until a minimum is reached. SA extends the process by introducing the possibility that a solution will be accepted even if it is worse depending on a temperature coefficient that gradually reduces as the algorithm proceeds.

A variety of parameters were used to determine how quickly a minimum value could be found for a beam distribution using such a method. Firstly, the total range to be considered for the beam combinations was limited to 1000, as it has already been determined that a low value has been reached several times within that range. The maximum step size is set to 100 to keep the step size within the range of the data. Table 8 shows the minimum value of the user distribution as obtained with various parameters.

Maximum Number of Iterations	**Initial Temperature**	**Determined Minimum Value of Distribution**
100	10	11
50	10	11
25	10	13

25	100	11
10	100	11

Table 8 -Minimum value of user distribution as obtained with various parameters

The initial temperature has been seen often to affect the effectiveness of obtaining a reasonable value within the maximum number specified, although this depends on how random numbers appear in the sequence. Figure 63 shows an example of the process of finding the best beam pattern using the SA process described. The blue line represents the sequence of random values at which the distribution of users is measured. This uses the Python random number generator and begins by imitating these pseudorandom numbers, with the variation over time determined by the initial temperature of the algorithm. The other line in the diagram shows the accepted candidate solutions for the beam distribution, which, in this case, obtains the lowest possible distribution.

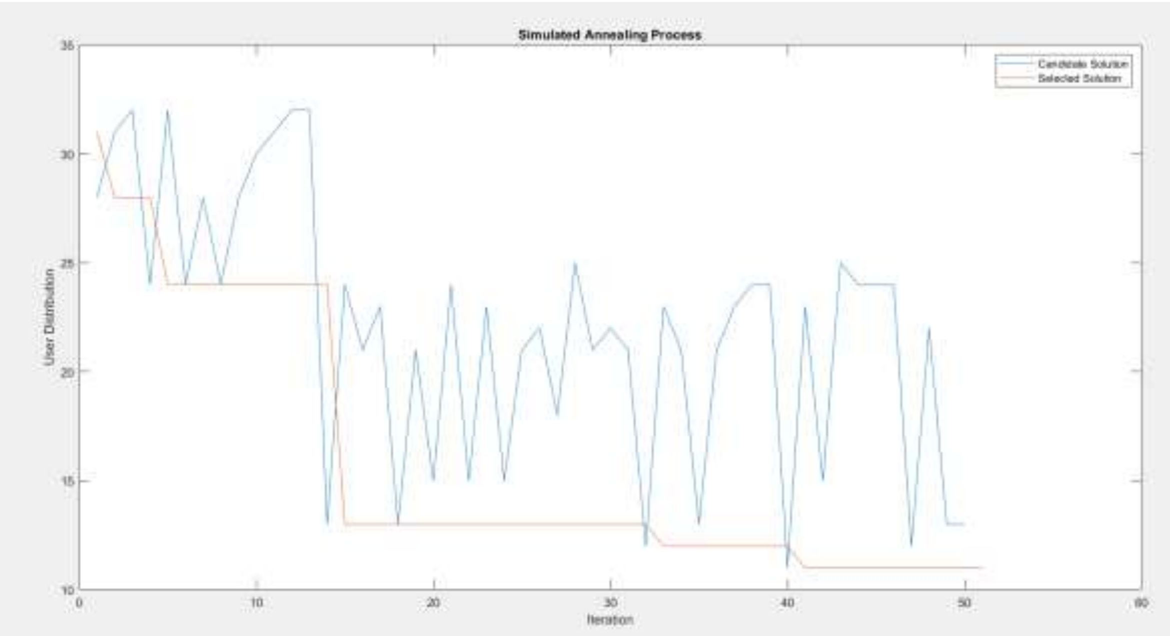


Figure 63 - Representation of SA for 15 beams of equal width with temperature of 10. The blue line represents the candidate solutions, and the red line represents the accepted solutions.

The previous example was for beams of equal beamwidth. The example in this section is for a set of beams containing beams of different widths. The purpose of such an example is to demonstrate how changing such parameters affects the approach to finding optimal beam configurations and how more complex problems can be addressed when designing mobile networks with beam selection.

In the previous section, the beams were all generated based on an isotropic rectangular phased array with 64 elements, each of which were active in generating the beam. For this example, fourteen beams are chosen with the first eight generated according to the array in the previous example and the final eight generated with only 32 of the elements, forming a square around the central point of the 8x8 square of elements. These eight beams are wider than the original eight beams, however they all point the strongest section of the lobe in the same direction, that is equally spaced in one sector of a typical tri sectored BS. This time, there are 3003 possible beam combinations, and the distribution for the users is shown in Figure 64.

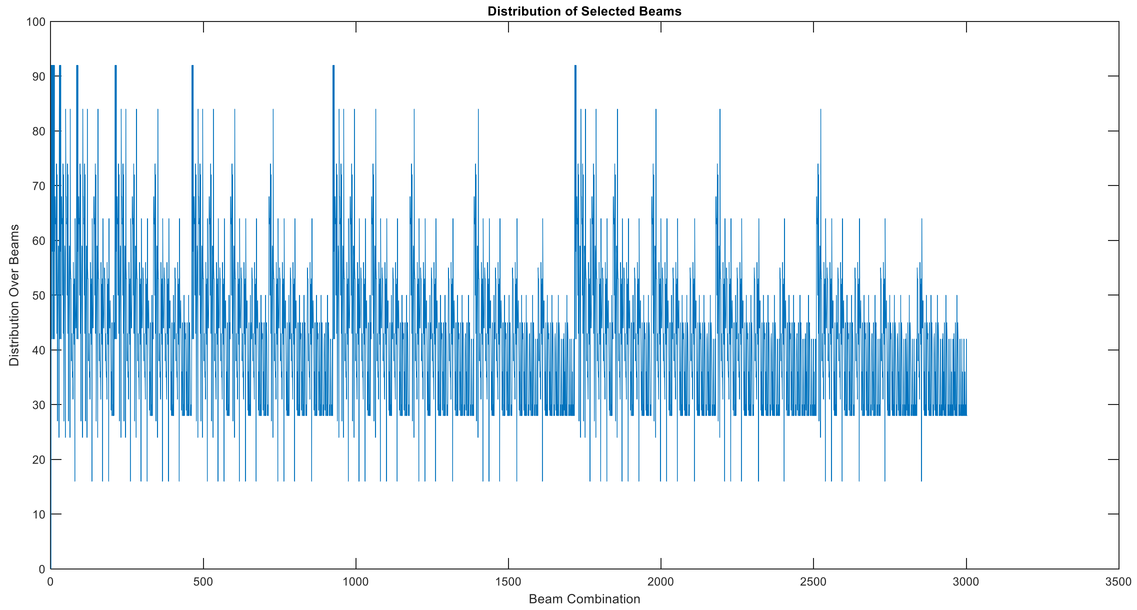


Figure 64 - Distribution of beams over users for 14 beams, eight of a narrower width and eight of a wider width.

Once again, the SA process significantly reduces the number of iterations down to 30 with an initial temperature of 100 (Figure 65).

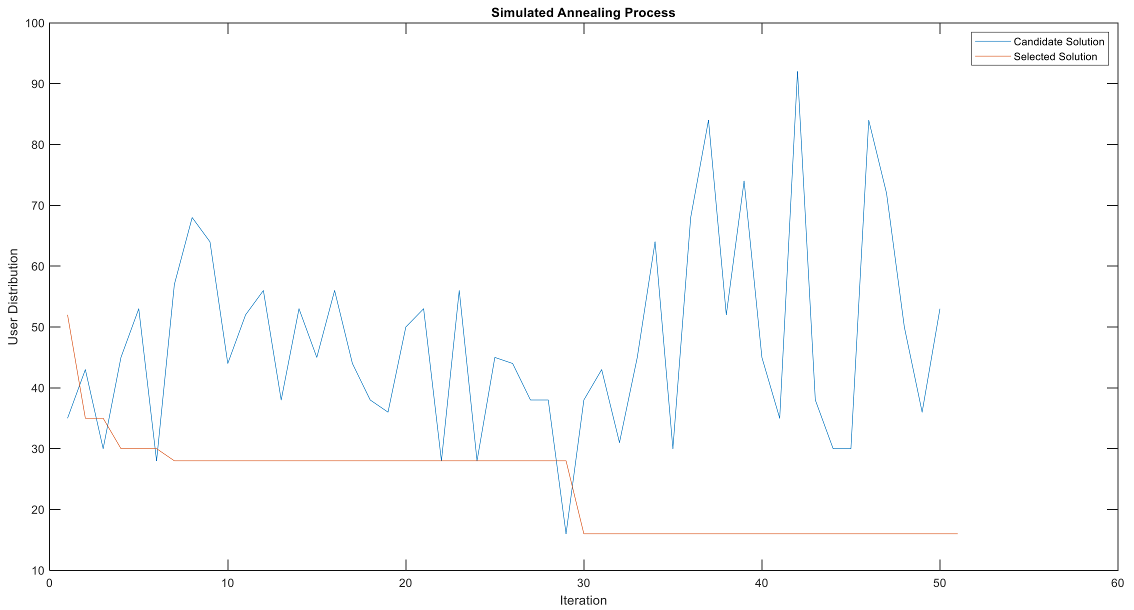


Figure 65 - Representation of SA for 14 beams, with eight of one width and eight of another.

The following provides a basic example of the performance of SA using a distribution where users are more clustered in certain directions. Take, for example, the heat map for a BS in Bromley, designed by Timothy Sanmoogen at BT as shown in Figure 66.

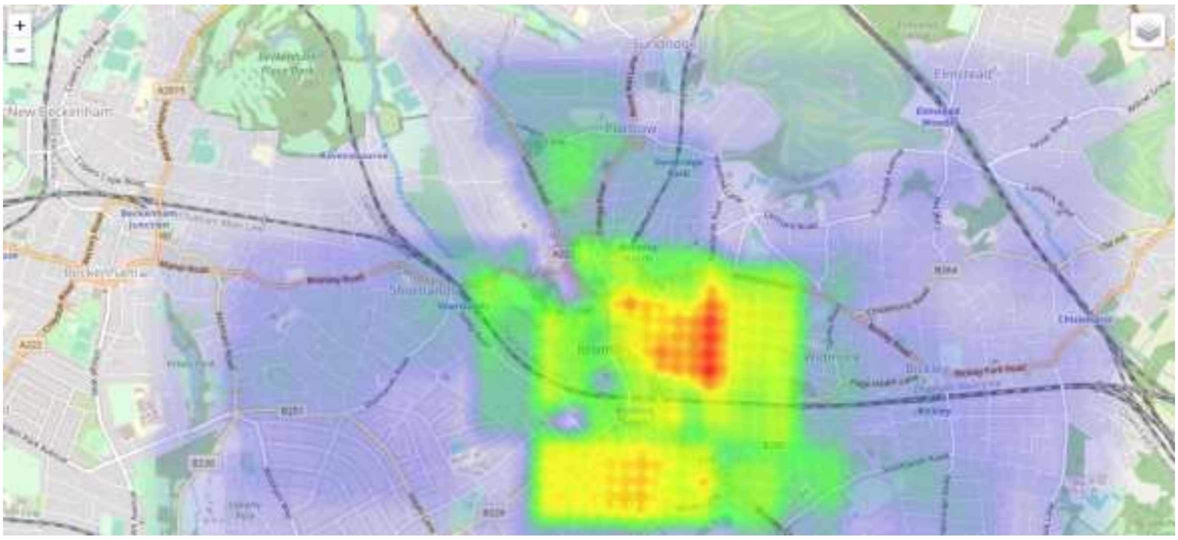


Figure 66 - Heat map for BS in Bromley (courtesy of Timothy Sanmoogen)

There is a BS located just to the west of the darker red area of the map, which represents a greater contribution of users. Again, considering a tri-sectored BS, the density of users would vary with a sweep from -60 to 60 degrees (if 0 degrees represented a direction of due east). This can be approximated through a random selection of 100 users with a larger number chosen to occupy a certain section of the total sweep (Figure 67).

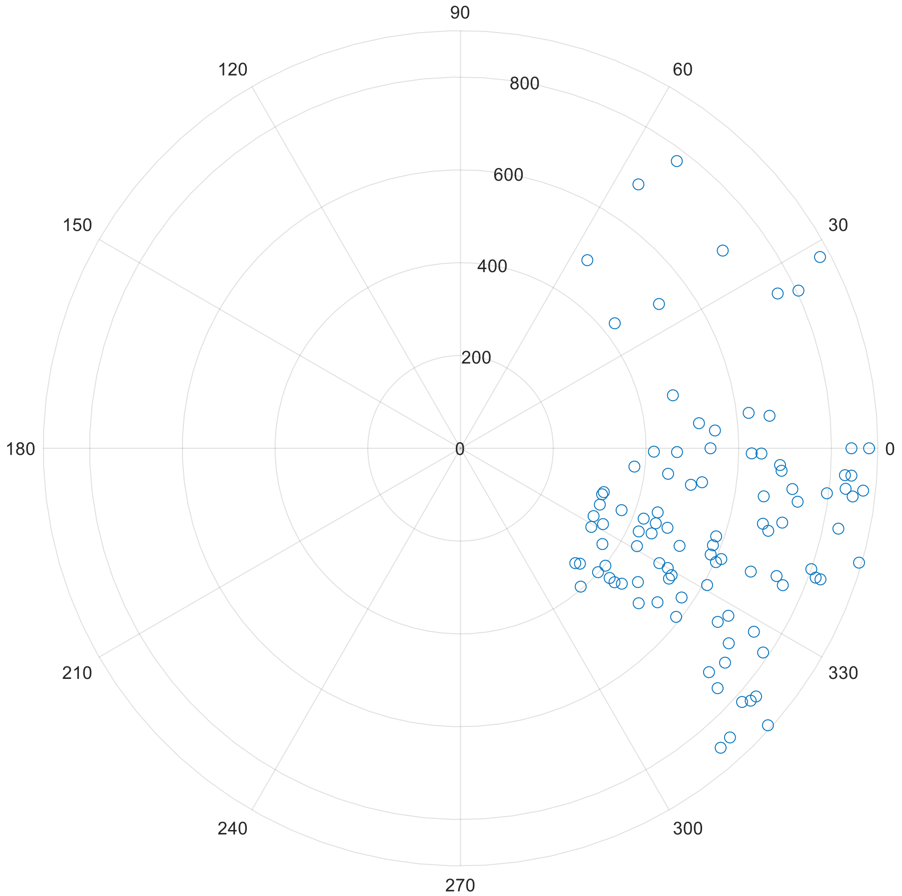


Figure 67 - Approximation of user distribution observed in Bromley. The scaling of the concentric circles is indicative only.

Significantly, the distribution of the users over the beams appears to have slightly different characteristics than the distributions for non-clustered random users (Figure 68). The lowest possible value is visited only a few times in comparison with the many times that it was visited for the non-clustered examples. It does, however, reach the next lowest value many more times, but this is quite far away from the lowest value. It may be that, without a sufficiently high temperature or number of iterations, this non-optimal value would be chosen. The specific circumstances would determine if this were a reasonable solution for the selection of beams or not.

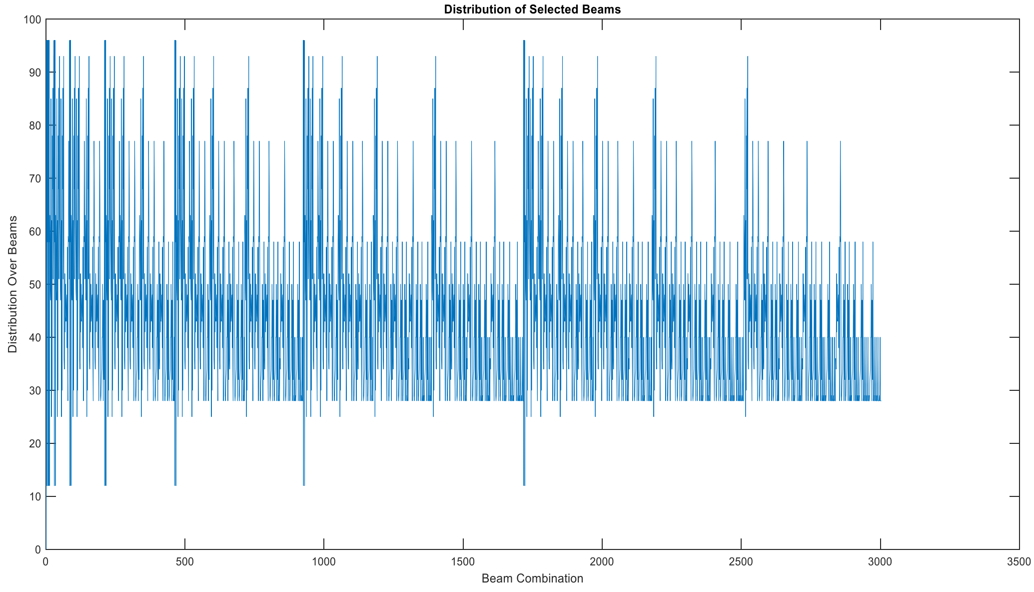


Figure 68 - Values for distribution of the users over the beams

Indeed, nearly 200 iterations are required to obtain the correct solution when the initial temperature is set to 100, and nearly 100 iterations to obtain the next closest solution (Figure 69).

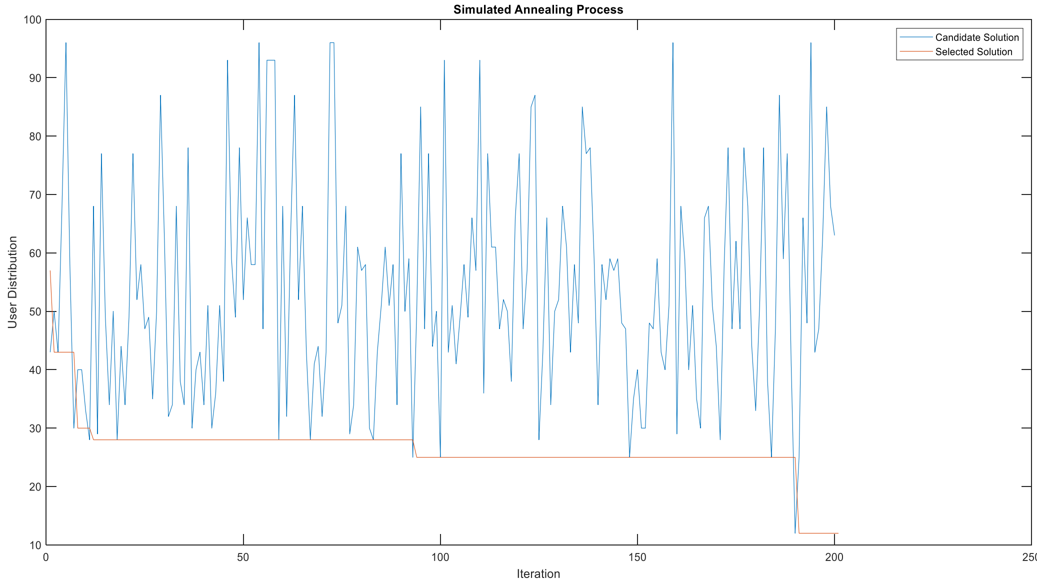


Figure 69 - Obtained candidate solutions and selected solutions for SA algorithm

However, setting an initial temperature to 200 allows for a correct solution to be obtained in only seven iterations.

4.7.16 Network Deployment Recommendations

From a network deployment and management perspective, the research presented in this section has a significant implication.

As described at the start of this section, the synchronisation process allows for a sweeping of beams over a coverage area to allow for the initial connection and synchronisation of user equipment. This process precedes the process of establishing data throughput. The specifics of the beam parameters are not defined in the standards and can, at least in theory, be established by the MNO. However, the number of beams that can be used and tested is limited in both a temporal and physical sense. Physically because of the limitations in hardware and potential difficulties in making changes to the parameters, especially if the site is already in operation. Temporally because, if a large number of candidate beams are used that vary between sweeps, then it will be a length process to obtain signal reports from each user related to each of the candidate beams. It is highly likely, especially in dense urban environments, that the characteristics of the coverage area in terms of the location of user equipment, will have changed several times by the time that results are obtained, by which time such results will not be relevant to the coverage environment.

The advantages that this research brings can be considered in both a real-time and non-real-time sense. The non-real-time sense related to planning and design. As previously discussed, running propagation models is time consuming and the number of beams that an operator may consider could easily be several thousand or more. The examples in this section have used simplifications of propagation models using the RSRP surfaces described combined with standard Rayleigh propagation models. This was done due to limited computational resources and because the aim of this study is to investigate potential benefits in the design and management of the beam management process with synchronisation, rather than to investigate the propagation characteristics of specific environments. However, when designing Radio Access Networks in a way that does require the use of more complex propagation models, these would need to be run only a few times in comparison to the many times that would be needed if run for each possible beam configuration. This is because it has been shown that, from the perspective of coverage, an optimal (or near optimal) set of beams is likely to have been found after only a small number of iterations through the set of candidate beams, relative to the thousands of iterations necessary for an exhaustive search of all the possible candidates.

The application of the process of beam selection discussed in this section could potentially be applied with self-learning networks in real time, however this would require further research. This is mainly because of the complexities in applying such concepts to specific network architectures, a process that would require careful investigation, and because of the much higher risks associated with applying optimisation in real time, since a sequence of bad solutions could potentially be very detrimental to network performance, leading to a degradation in user experience that could then lead to financial risks, especially if user experience is consistently worsened, or safety risks if a minimum level of service is not maintained. However, a careful application of beam optimisation methods in real-time could potentially yield great benefits for performance if one considered a scenario such as the following:

Suppose that a mobile BS has access to a large set of potential beams but is using a standard subset of evenly spaced beams of equal width. These eight beams are used for sending the synchronisation blocks to the users wishing to connect to the network. It may be possible to add a threshold condition to the computation architecture responsible for synchronisation, that states, for example, that the user distribution is acceptable below a certain value. In other words, once the beams are such that they are each serving a number of users for which that number of users does not vary between beams by more than a specified amount, then this is acceptable. If, on the first sweep of the beams, this number is not an achieved, then the computational hardware could choose a different set of beams based on the methods described in this section. Since such a method has been shown in theory to yield a near optimal solution in a relatively small number of iterations, it should be possible for the BS to obtain its threshold value within a small number of sweeps, which may be fast enough to be useful before a significant change to the user environment. Likewise, when the environment does change, it may be possible to adapt to this environment more quickly than if an exhaustive search of the beams was necessary, especially if the network operator already obtains knowledge of likely changes, such as changes that usually occur during the evening rush hour. It may then have already determined a set of possible optimal beam from the previous day, or a sequence of previous days, that can then form the initial set of beams, thus making any optimisation even faster.

4.7.17 Implementations

It is possible to infer a method for applying the insights obtained from the investigations described, although the specifics of how this would be implemented would form part of further research. The proposed technique is intended to optimise coverage within telecommunications networks during the synchronisation phase of transmission at the base station, allowing for resources to be allocated more effectively depending on the distribution of users around the base station and the environment in which the system operates. The method described in this document makes use of a series of stages that are run alongside the synchronisation process, which is defined within the existing telecommunications standards and therefore not part of this section. Some of these additional stages, the detail of which form the basis for this recommendation, are dependent upon the output of the standard synchronisation process. This section includes both the structure of the additional stages, how they operate and how they are connected in the application of coverage optimisation.

The use of the proposed approaches to coverage optimisation during synchronisations allows for the following advantages over existing methods:

1. Ensuring that, within any given service area, the number of users who are able to successfully connect to the network is at the maximum possible value given the available resources. The actual maximum number will be dependent upon the physical configuration of the base station and the allowed adaptability of the beamforming permitted by the base station's hardware and software.
2. Adapting the synchronisation coverage dynamically based on the current distribution of users within a service area.
3. Adapting the synchronisation coverage based on distributions of users that are predicted to occur based on historical data and/or statistical models.
4. Obtaining the optimal beam configuration for synchronisation coverage from a selection of candidate beams in a shorter time than would be required if each beam configuration would need to be tested separately.

The elements of the proposed method are shown in Figure 70. These elements can be connected in three configurations that are described later. The elements are shown within the boxes in the diagram, and the lines between them show the other elements to which each element is connected when in the configuration for option 1. In this description, an element refers to a step within the beam selection process. The first column of elements on the left-hand side, except for the 'Active Beam List' element, represent the standard steps required for synchronisation and therefore do not form part of this description. The functioning of the other elements, aside from those forming part of the standard synchronisation process, are described in this section. These descriptions are followed by explanations of how the elements are configured in the three separate configuration options. It should be noted that the functioning of some of the elements are associated with explanations of how they should function, whereas others are not precisely defined and could be implemented in several different ways that are not defined as part of the information here. Within the explanations, 'controller' is used as a generic term to define the computation entity that is responsible for managing the beam selection process described here.

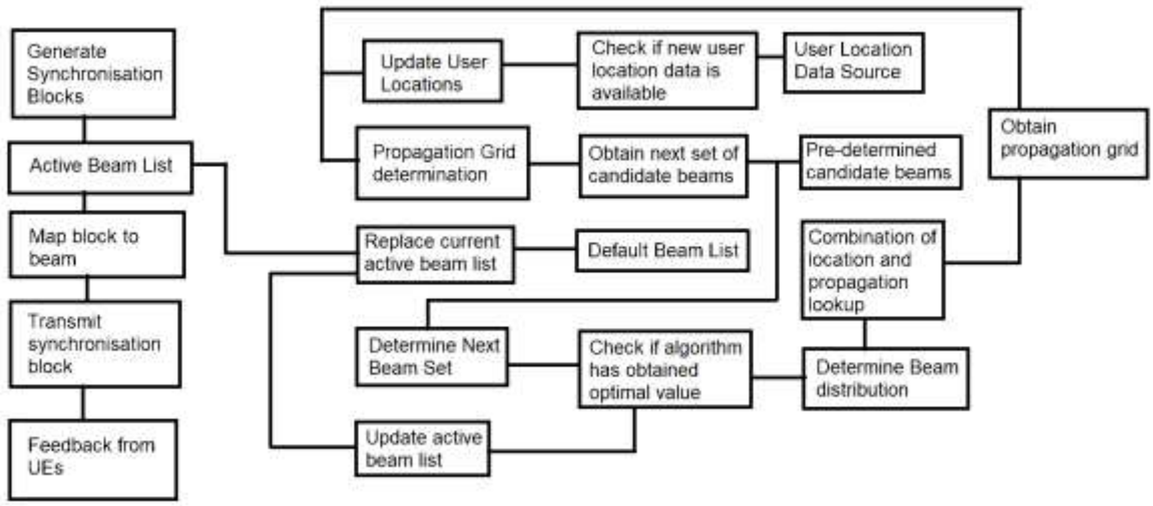


Figure 70 - Proposed implementation elements configured for option 1.

<u>Active Beam List</u>

This element contains the information about the beams to which the synchronisation blocks are currently mapped as part of the synchronisation process. The information provides sufficient detail for the base station to determine beam azimuth angle, elevation angle and width. The step represented involves obtaining the current beam information and providing it to the next element, which maps the beam to the synchronisation block.

Update User Locations

This controller updates the user locations if new user location data is available. The user location data is stored as a class of x-y coordinates within a grid that has the same dimensions as the x-y plane of the RSRP curve, which is described later. It is necessary, as part of this step, to verify the formatting of the user location data and to update this formatting if necessary.

Check if new user location data is available

The controller checks the user location data source to determine if new user location data is available. If it is, it proceeds to the update user locations step.

User Location Data Source

User location data is provided by an external source. The user locations are not estimated by the hardware as part of the synchronisation process,

Propagation Grid Determination

The determining of the propagation grid can occur while the algorithm is running, or before the algorithm is used through the creation of a library where the grid can be obtained when necessary. Each candidate beam is associated with a propagation grid that is used to estimate the RSRP value from the perspective of each UE. The grid is obtained from an 'RSRP curve,' and has the dimension that correspond with the coordinate system that is used to describe the user locations. The RSRP curve is obtained by the following process:

The users are placed evenly around a base station. Only a third of the total area is occupied with users to simulate a typical tri-sector base station configuration. The range for these examples is set out to a pre-determined distance to reflect the extent of the cell area. Additionally, an area near to the base station is excluded to avoid near-field effects and complications with overlapping beams. A model for 5G is then run for each of the beams. On each occasion each user reports back its measured RSRP value for the beam. The model that is used is not defined and may range from a simple inverse square model to a more elaborate and detailed electromagnetic simulator. These reported RSRP values at specific points, correspond to points within a two-dimensional RSRP surface. This surface is then extended to cover the entire area of interest from the point of view of the propagation environment.

The generation of the normalised RSRP surfaces allows for the problem to be expressed more precisely as a combinatorial optimisation problem. This formulation is developed with attention specifically to the assumption that it is preferable to provide even coverage of broadcast synchronisation beams for a given user distribution and environment.

Check if algorithm has obtained optimal value

The notion of the optimal set of beams is defined in a specific way. This corresponds with the explanation provided earlier.

Obtain Next Set of Candidate Beams

The next set of candidate beams are obtained through a simulated annealing process. The process itself is well established, having been developed around 1970, but the specific way that it is applied to the relevant matrices and vectors is novel, to the best of the author's knowledge. The explanation assumes that a random set of candidate beams are chosen and tested according to the methods described in the relevant sections. This initial set creates a beam distribution, showing the differences between the number of users served by the beam that serves the largest number of users minus the number of users served by the beam that serves the least number of users. In addition to the initial set of beams, a maximum number of algorithm iterations is also set, and may be altered depending on the situation in which the algorithm is intended to be run. A maximum step size is also set, which is used to set an upper limit on how far away from the next estimation can be. In this context, this means that all of the possible combination of available beams are indexed, and so the next tested combination of beams can only be within the range defined by the positive and negative expression of the maximum step size, starting at the index value of the current set of beams.

An initial temperature is also set, the value of which has been pre-determined through experimentation. The temperature is used to determine whether a result is accepted or not. In other words, it determines whether or not the beam distribution obtained from the current iteration of the algorithm is accepted. If is accepted, then this beam configuration is recorded (and possibly implemented depending on the specific implementation. The possible implementation configurations are described later in this document.) If there is a previously selected beam configuration, then this previous value is discarded and replaced by this new beam configuration. The entire process continues until either the maximum number of iterations has been reached, or the beam distribution has reached a pre-set threshold value.

Pre-Determined Candidate Beams

This is the set of both the parameters of the beams that could be selected, including their widths, azimuths and elevations, and the combinations in which these beams could appear.

Obtain Propagation Grid

This is the RSRP surface described earlier.

Replace Current Active Beam List

This function changes the set of beams that are applied to the synchronisation signal blocks. Depending on the way that the functions are applied, this could happen often throughout the iterations of the beam selection algorithm, or only once a final set of beams has been chosen.

Default Beam List

This is the list of beams that are used by default before any optimisation has been applied. This list is stored permanently and is also applied in the event of a system reset.

Combination of location and propagation lookup

The x-y coordinates for each set of users are mapped onto the propagation grid, thus providing the RSRP value for each user for the beam. This is done for each of the beams under consideration.

Determine Next Beam Set

If the process has not reached the optimal value according to the set criteria, the next beam set is selected randomly within the confines specified by the maximum step size.

Determine Beam Distribution

Once the combination of each of the locations with each propagation lookup table for the beams under considering has been performed, the number of users served by each beam is calculated. This is done by considering the maximum RSRP value obtained for each user, and which beam that corresponds to. It is then possible to calculate the number of users served by a beam by finding how many users are provided with a maximum RSRP value by that beam. The number of users in the beam that serves the minimum number of users is subtracted from the number of users in the beam that serves the maximum number of users to obtain the beam distribution.

Update Active Beam List

Once a new beam set has been determined, the active beam list is updated before being passed to the synchronisation functions. This may occur regularly, or only once the method has determined the optimum set of beams, depending on the configuration option used.

Configuration Options

There are (at least) three possible ways of configuring the functional blocks described in this document. The one that is chosen will depend on how frequently the active beam list, i.e. the set of beams that is actually used to provide coverage to users, is updated. It will also depend on the source of the data that is used to inform the choice of beams, and whether these data are entirely based on predictions, or whether real-time feedback from the UEs is used to inform the choice of beams. The rate at which the active beams will be updated will also depend on whether it is deemed necessary to update this list as the simulated annealing algorithm obtains new improved results, or whether the system should wait until the algorithm has completed all iterations before updating the list.

Option 1: Perform all optimisation before updating active beam list.

Option 2: Perform optimisation using predicted data, updating as improved results become available.

Option 3: Perform optimisation based on obtained results from UE feedback, with assumed user locations.

4.7.18 Future Work

There are three main categories of future work related to this discussion:

Firstly, the implementation of more supervised learning techniques on both the obtained data from the simulation and the refined user data. The aim of applying supervised learning to the simulation data is to investigate the suitability of different algorithms for categorisation for this type of problem. The application of learning to the user data is an attempt to simplify the process of obtaining the simulation data by identifying clusters instead of having to simulate each point individually.

Secondly, it will be necessary to investigate further the simulation process itself, as the current method may not be accurate enough to have confidence in the results. At first, it may be possible to modify the approach described using different types of propagation models for different environments (as the current method is based on a standard Rayleigh model). However, it will be necessary to decide upon the level of complexity required within the channel model for this use case.

Finally, it will be necessary to consider possible implementations of AI for beam selection within networks. At this stage, this is a more speculative part of the research.

4.8 Radio Coverage Prediction

In this section, we present research work associated with radio coverage prediction using Machine Learning. First, real network coverage data is extracted using the minimisation of drive test (MDT) technique for a timescale of two months – January and August. The data from two different seasons led us to understand the impact of season on radio coverage. Following this, a machine learning model using the deep neural network technique is developed. This model assists in predicting radio signal strength in areas with limited and/or no network coverage data. Finally, prediction results from the trained model were compared against 3GPP standardised WINNER II propagation model whilst classical machine learning techniques such as linear regression and simple neural network were utilised to measure the accuracy of the trained model.

4.8.1 Introduction

Measuring customer experience on mobile data is of utmost importance for global mobile operators as it provides the ability to optimise network performance based on user needs and demand. One of the key performance indicators for mobile network coverage analysis and management is signal strength – reference signal received power (RSRP). Radio data can be gathered through a range of different methods such as geolocation, drive test, open-source, crowd data etc. To measure RSRP values, drive tests are usually performed by network operators for network data collection. This requires significant human efforts, explicit hardware, and substantial capital expenditure (CAPEX). Estimating and predicting RSRP values has been a key component in the Quality of Service (QoS) analysis. There are two approaches identified for predicting RSRP.

a) The first is using the 3GPP standardised propagation model. Here field measurements are taken from different terrain and scenarios, such as urban, rural, suburban, macro, or microcells - it summarizes the propagation rules based on several test values. Typical empirical-based path loss models as documented in 3GPP TR 38.901. Such modelling depicts the channel properties in a general and coarse way, which may not be accurate enough for specific environments. Although modelling the RAN and UE behaviour that relies on the above models is achievable, such modelling depicts the channel properties in a general and coarse way, which may not be accurate enough for reflecting a specific environment.
b) A second method is a data-driven, machine learning approach. Environmental and radio features in a defined polygon or area are first extracted using a defined method. In the presented work, the minimization of drive test (MDT), a 3GPP standard technique, is used to gather network measurements. Here, an individual's mobile device, also referred to as user equipment (UE), that is logged in the network collects measurement data and feedback it to the base station (BS). Figure 71 demonstrates one month of radio data collected using the MDT technique from one of the cell sites. This cell site has three sectors namely sector A, B and C. The data plotted on the figure is processed data that is all the outliers has been removed using Inter quartile range (IQR) technique. Each point represents a UE associated with one of the sectors of the cell. UE location i.e., latitude, longitude and signal strength are recorded. The area has been divided into (10*10) metre discrete bins to reduce the effects of minor observation errors.

Data collected using MDT technique has its own challenges such as (a) the signal strength between UE devices at the same location and time could differ more than +-6dB (b) Inaccurate locations information and signal strength for indoor UEs (c) Some locations only contain limited data points due to imbalanced UE distribution. However, with the emergence of ML deep neural networks (NN), fine-grained propagation channel modelling for radio networks is possible. In Figure 75, we present the proposed data-driven, two-tier neural network (NN) model for RSRP prediction. Here, the regression relationship between a target location and RSRP is evaluated quantitatively based on the large historical RSRP data set that was obtained using the MDT technique.

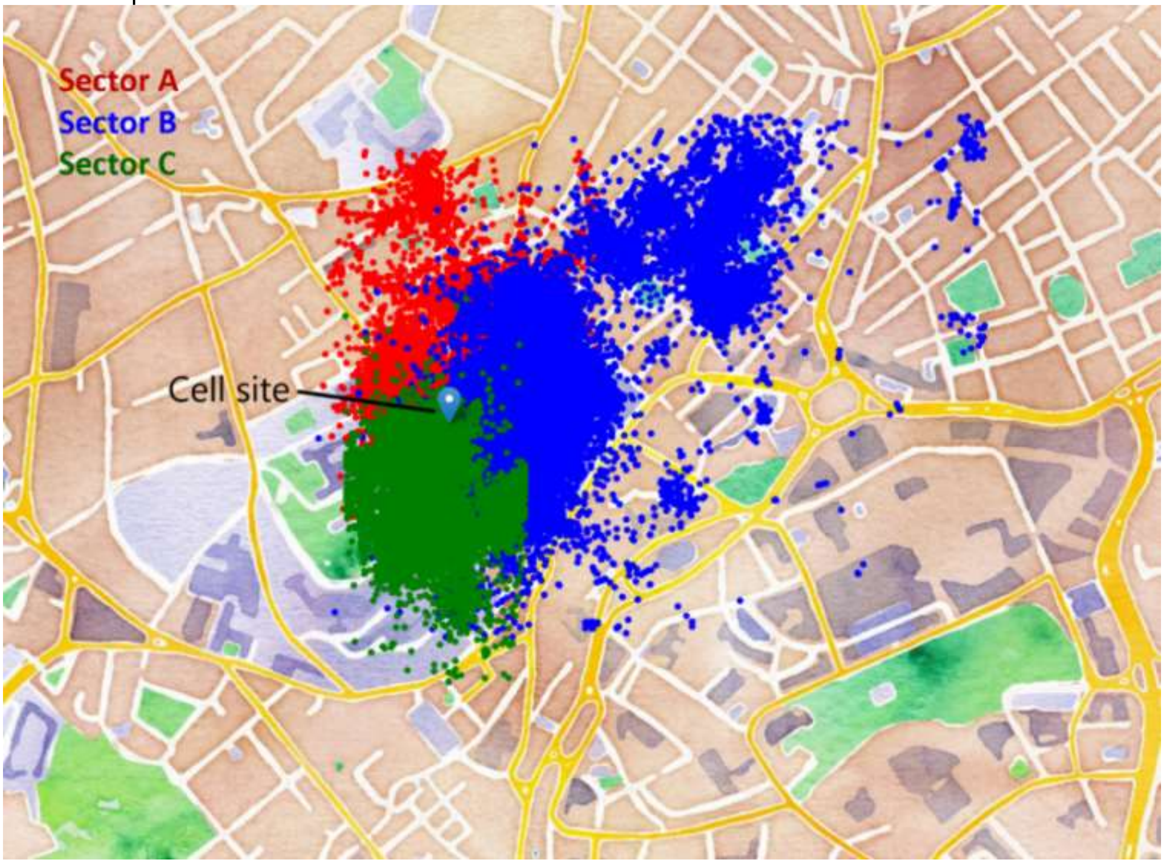


Figure 71 - Radio coverage data collected using MDT technique from one of the cell sites for the month of January.

4.8.2 Radio Coverage Prediction using Mathematical Model

Update 3GPP standardised 3D-Urban Macro (UMa) LOS propagation model was utilised as a baseline technique to determine the signal strength per bin. Here, the BS and UE locations were used as an input. Next, distance between the BS and each UE is calculated. Thereafter, break point distance calculation was performed to distinguish between the path loss equation to be implemented as the employed propagation model is dual slope. The path loss modelling is performed using the equation (46):

$$PL = \begin{cases} 22log_{10}(d_{3D}) + 10log_{10}(f_c) & 10m < d_{2D} < d'_{BP} \\ 40log_{10}(d_{3D}) + 28.0 + 20log_{10}(f_c) - 9log_{10}((d'_{BP})^2 + (h_{BS} - h_{UT})^2) & d'_{BP} < d_{2D} < 5000m \end{cases} \quad (46)$$

Where, $h_{BS}$is the height of base station that is assumed to 17.5m and $h_{UT}$is the height of user equipment assumed to be 1.5m and h_E is the effective ground height equals 1m. The cell power is assumed to be 23 dBm and antenna gain is 16.5. d_{3D} is the 3-D distance between BS and UE, f_c is the carrier frequency with value of 1800 MHz, d'_{BP} is the break point distance calculated as 4*$(h_{BS} - h_{UT})$*$((h_{UT} - h_E) f_c/c$

The signal strength is calculated using the equation (47):

$$\text{Signal strength(dB)} = \text{cell power} + \text{antenna gain} - \text{pathloss} \quad (47)$$

Next, linear regression method was utilised as a baseline machine learning technique to predict the signal strength using the historic RSRP data extracted using MDT technique. Due to the uncertainty of the transmission channel, outliers are removed using Inter quartile technique. The processed date in particular UE location (latitude, longitude), distance from the base station and corresponding signal strength is fed into the regression model. The output is the predicted signal strength with respect to distance. Figure 72, Figure 73 and Figure 74 demonstrates the signal strength prediction using the standardised UMa path loss model, break point distance and predicted signal strength with respect to distance using linear regression technique for each sector.

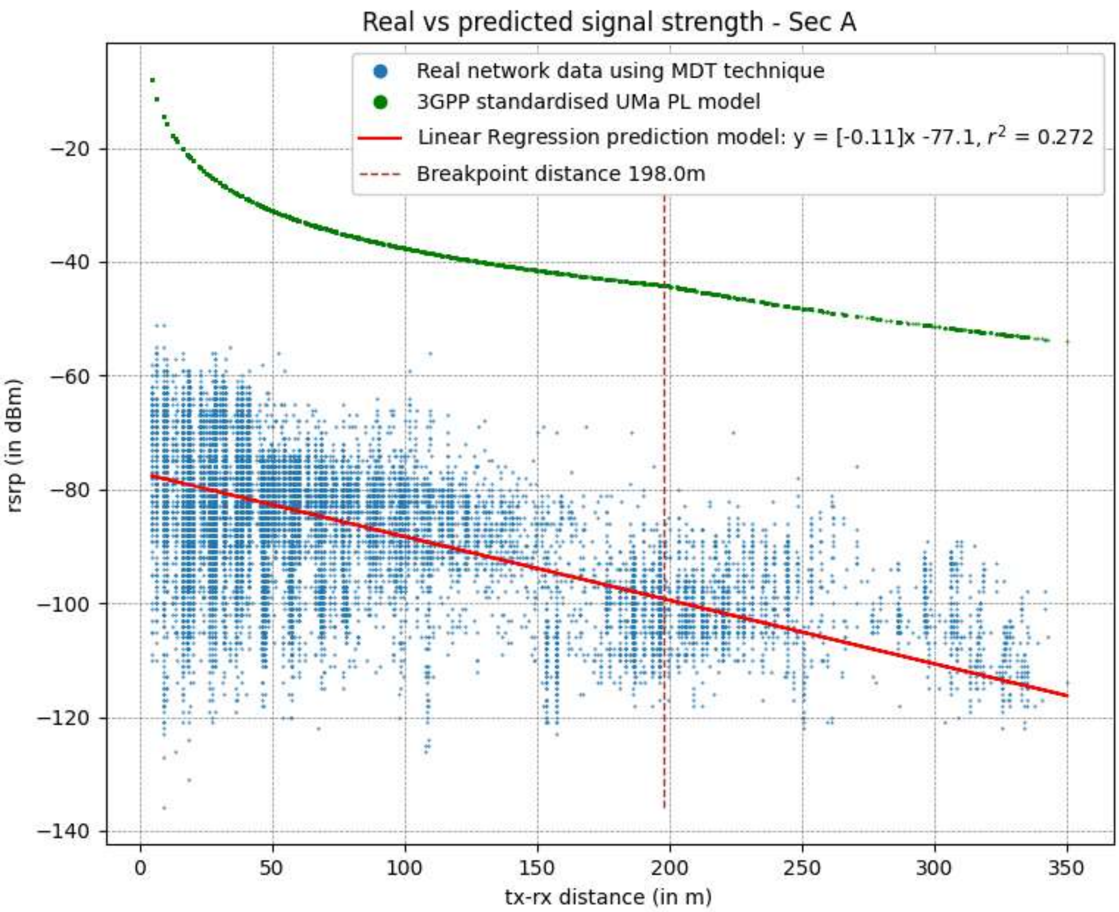


Figure 72 - Graph presents the real network data extracted using MDT technique, signal strength with respect to distance using 3GPP standardised UMa PL model and Linear regression technique for Sector A

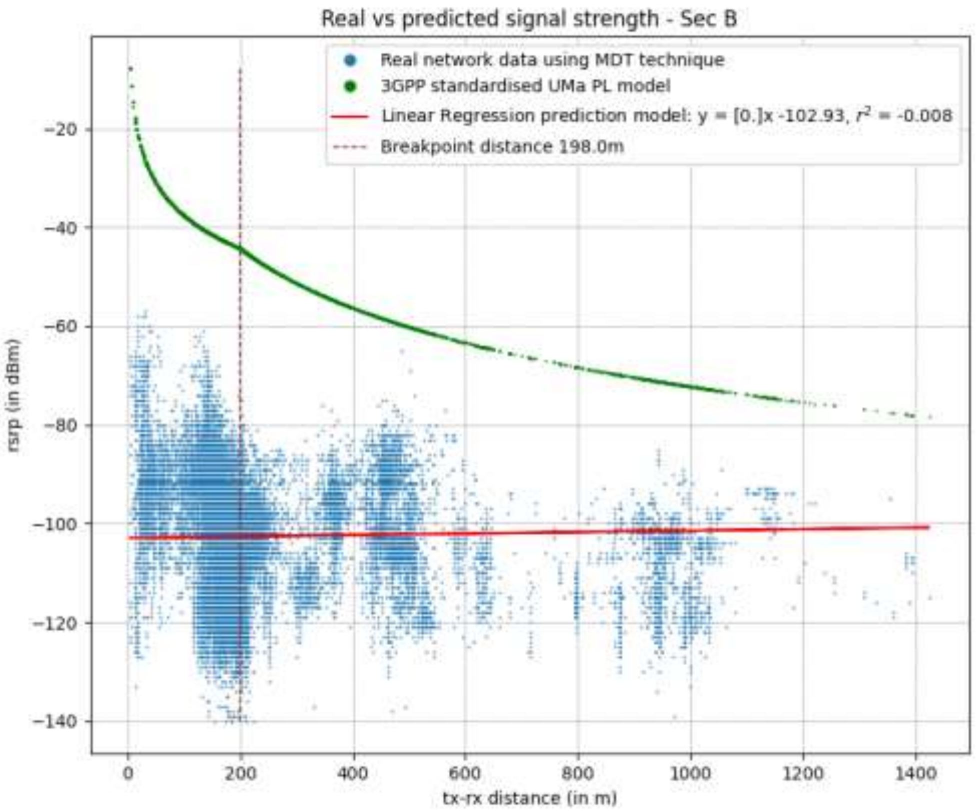


Figure 73 - Graph presents the real network data extracted using MDT technique, signal strength with respect to distance using 3GPP standardised UMa PL model and Linear regression technique for Sector B

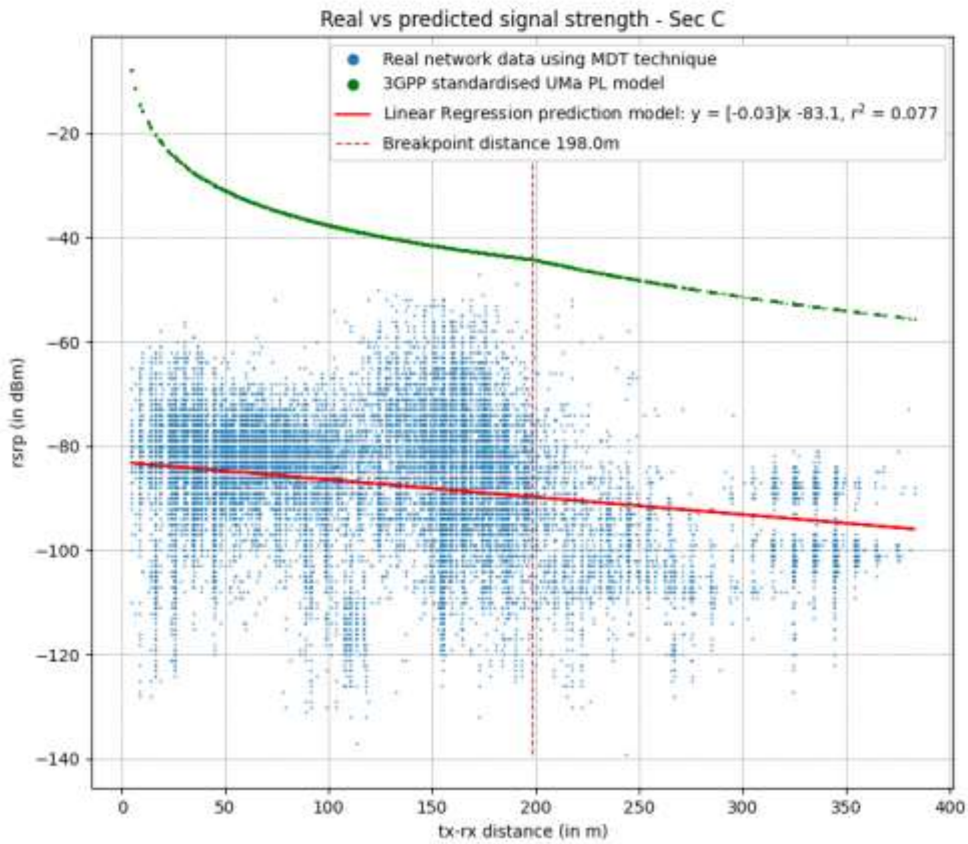


Figure 74 - Graph presents the real network data extracted using MDT technique, signal strength with respect to distance using 3GPP standardised UMa PL model and Linear regression technique for Sector C

4.8.3 Radio Coverage Prediction using Machine Learning

It is known that the key factors affecting signal transmission exist in the characteristics of the environment, such as how far the signal transmits, and how many reflections, absorption, and scatterings it encountered during this period, even the building materials and vegetation. But these characteristics are difficult to model accurately. Although ray-tracing methods are aiming to restore the transmission path as much as possible, urban-scale raytracing is too time-consuming and complicated to be realistic. Moreover, simpler images would improve not only the training of the model and the hyperparameter search but also the final performance of the methodology. Therefore, in this research work, we do not seek accurate tracing results. We focus on how to describe the possible impact of the signal transmission path with UE location and environmental information. Figure 75 presents the proposed two-tier NN model with emphasis on NN architecture, training data generation and training scheme.

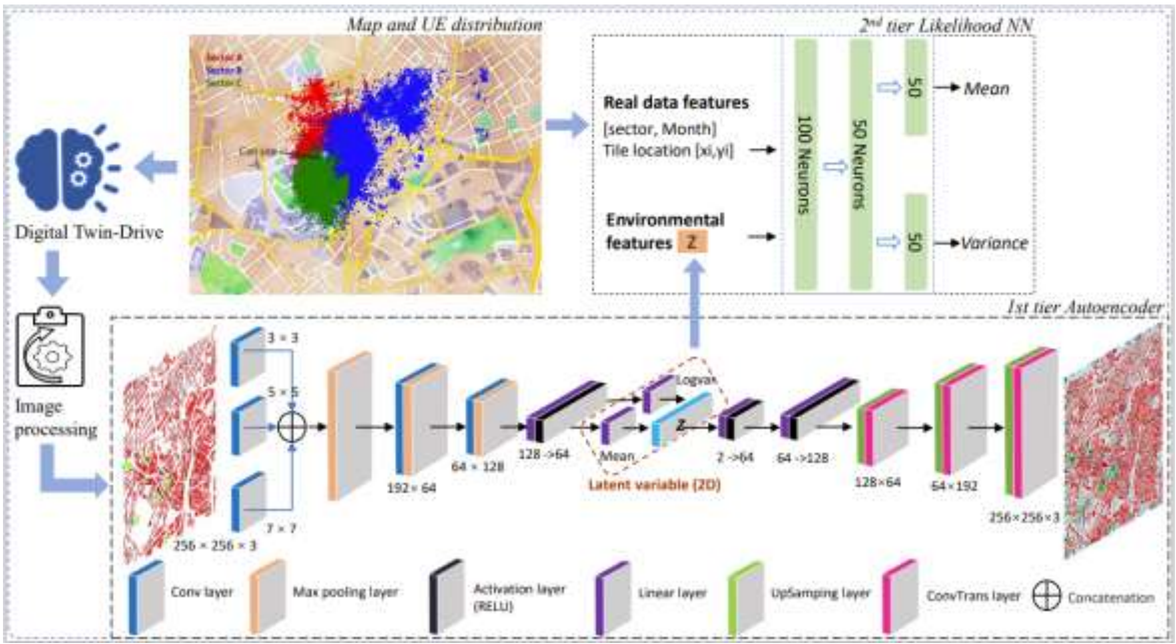


Figure 75 - Proposed two-tier neural network architecture

It is a generative model that exploits MDT data on a digital twin (DT) framework to predict signal strength. The first tier is designed as a CNN-based VAE to extract relevant environmental features while the second-tier network is designed as a fully connected network with two heads that outputs the mean and variance of RSRP in each location. The underlying representation of VAE is z - this parameter will assist the training of the two-tier neural network. Here, not only is the historical real data of RSRP utilised but also geographical statistics information is considered. The correlation between locations and RSRP distributions is then mapped through data compression. Regarding the extraction of environmental features, we propose that according to the particularity of the environment served by each BS, it is necessary to construct images that can reflect the transmission environment from BS to UEs, thereafter, extract critical information from the images as auxiliary training features of the RSRP prediction model.

Figure 76 presents one of the BS-UE association images generated using the DRIVE simulator. The red polygons represent typical buildings, and green polygons represent the foliage. BS is represented by a black circle, and UE is marked by a blue triangle. The right side of the figure shows zoomed areas of BS and UE. The connection between the BS and the UE is highlighted by a light blue line. 10000 such top-view geographical images with resolution (256*256*3) are collected to train the VAE. The BS-UE association modelling is accomplished by a modified digital twin (DT), while the process of image dimension reduction to feature extraction is completed by a convolutional neural network (CNN)-based variational autoencoder (VAE). Once the model is trained, the encoder section of VAE serves as an environment feature extractor. This parameter is then fed as an input feature along with other environmental and network data to the second tier of the NN model to assist in the training of the RSRP prediction.

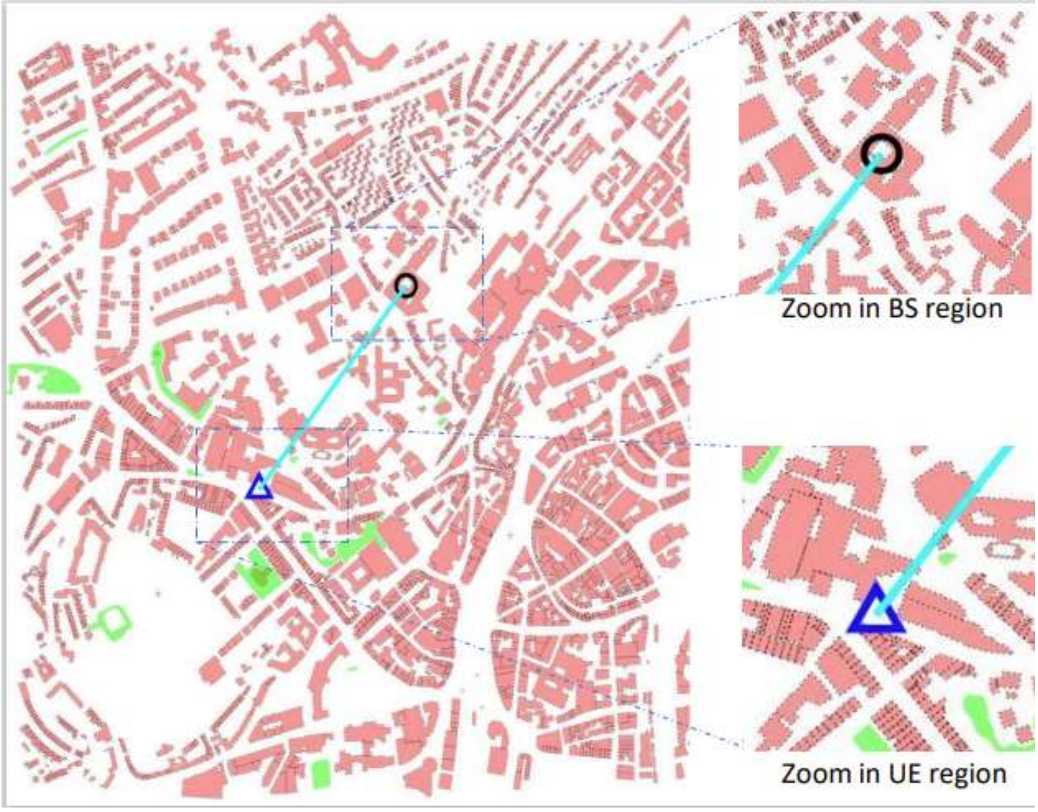


Figure 76 - BS-UE association images generated from DRIVE simulator

The second tier is designed as a likelihood model. The first two layers are with 100 neurons and 50 neurons respectively, and the last layer has two heads with 50 neurons each which output the mean and variance of each bin. Here, the environmental features and real MDT data features are adopted, formulating an integrated training process. Since the number of training features is small, a multi-layer perception (MLP) trained in supervised learning can satisfy the task requirements. Due to changes in the transmission environment, the RSRP value recorded at each location is time varying. From a statistical point of view, the RSRP values recorded at this location conform to a normal distribution. Therefore, the MLP model is designed as a likelihood model with mean and variance outputs, which takes the output of the encoder and BS-recorded features as training inputs.

The real-world dataset is provided by BT Labs, which records the monthly data of about 16,000 bins served by one BS. Each bin covers a square of 10m*10m. The accuracy of trained model has been monitored using different statistical measure thresholds. We evaluate the models in terms of the average RSRP prediction error through a 20-fold cross-validation scheme, and early stopping is adopted in the training process with a stop patience of 8. The VAE was trained in an offline way. Experiments are performed using the Intel 2 E5-2640v4 CPU, 2 RTX 2080Ti GPU and 4 × 32G DDR4 SDRAM. The data pre-processing is performed by the CPU whilst the training stage relies on the GPU. The training is based on PyTorch. The training and validation set are divided according to the 80% and 20% of the total both for VAE and likelihood model. The batch size of VAE is 50 and for likelihood model is 3000. Both models use Adam to be an optimiser with the default learning rate. Figure 77 demonstrates more detailed boxplot results, which summarize the distribution characteristics of the MAE on the test set in 20-fold cross-validation for both MLP and the proposed model. In general, our proposed model trains more stable (with fewer outliers) and has a smaller and more concentrated error distribution as shown in Figure 77.

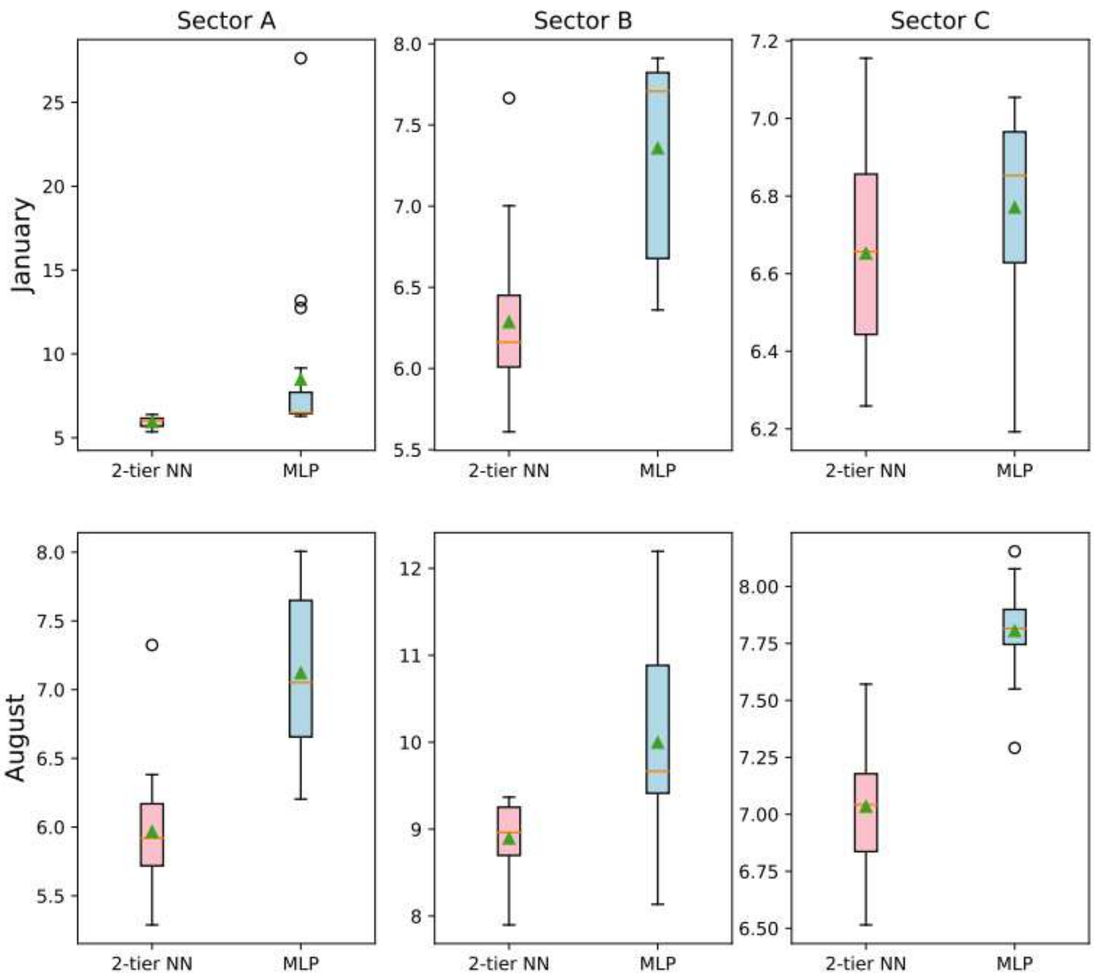


Figure 77 - The boxplot of cross-validated MAE for RSRP prediction based on different sectors and months

Table 9 presents MAE results of the empirical model, MLP model and proposed two-tier NN. Compared with the empirical model our proposed model can improve the prediction accuracy by about 2%, and the largest increase accrues on the subset January sector C, where the MAE is reduced from 10.71 dBm to 6.758dBm, about 38%. Meanwhile, compare with the simple MLP model, the prediction accuracy of our proposed model has an improvement by nearly 10%, and the largest improvement lies in the August sector A, around 16.4%.

Data information			Validation results (in dBm)		
Month	Sector	No. of samples	Empirical	MLP	Proposed model
Jan.	A	21236	7.29	6.478	5.840
	B	53208	7.99	7.323	6.243
	C	15172	10.71	6.758	6.636
Aug.	A	10699	8.06	7.104	5.941
	B	24361	10.08	9.726	8.623
	C	20228	8.78	7.790	7.012

Table 9 - Data information and model validation results for different data subset

4.8.4 Conclusion

A novel two-tier NN architecture is proposed to realise the accurate RSRP prediction. The VAE-based environmental feature extractor constitutes the first-tier network which is used to distil the critical information from BS-UE association top-view geographical images, where the image generation is finished in a modified DT (DRIVE) by using OSM of the given area. Meanwhile, the second tier is designed as a likelihood model which takes the outputs of the above extractor and real data features for training. The numerical results evaluated on real-world datasets show the gains of the proposed model in terms of prediction accuracy. The overall accuracy improvement is more than 20% and around 10% compared with the empirical and a simple MLP model respectively, and it can reach 38% and 16.4% improvement in the best validation case.

4.9 Section 4 – References

[4.1] Levine, Sergey, Aviral Kumar, George Tucker, and Justin Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems.", https://doi.org/10.48550/arXiv.2005.01643

[4.2] S. Sutton, A. G. Barto, and F. Bach, "Reinforcement learning: An introduction", MIT Press, 1998

[4.3] M. Bennis and D. Niyato, "A q-learning based approach to interference avoidance in self-organized femtocell networks," in 2010 IEEE Globecom Workshops, 2010, pp. 706–710.

[4.4] M. Dryjanski, L. Kulacz, and A. Kliks, "Toward modular and flexible open ran implementations in 6g networks: Traffic steering use case and o-ran xapps," Sensors, vol. 21, no. 24, 2021.

[4.5] S. Yue, J. Ren, J. Xin, D. Zhang, Y. Zhang, and W. Zhuang, "Efficient federated meta-learning over multi-access wireless networks," IEEE Journal on Selected Areas in Communications, vol. 40, no. 5, pp. 1556–1570, 2022.

[4.6] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," IEEE Robotics and Automation Letters, vol. 4, no. 4, pp. 4555–4562, 2019.

[4.7] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," arXiv preprint arXiv:1803.02999, 2018.

[4.8] J. Koneˇcný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in NIPS Workshop on Private Multi-Party Machine Learning,2016.

[4.9] S. Lin, G. Yang, and J. Zhang, "A collaborative learning framework via federated meta-learning," in 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), 2020, pp. 289–299.

[4.10] L. Zhang, C. Zhang, and B. Shihada, "Efficient wireless traffic prediction at the edge: A federated meta-learning approach," IEEE Communications Letters, pp. 1–1, 2022.

[4.11] A. Dogra, R. K. Jha, and S. Jain, "A survey on beyond 5g network with the advent of 6g: Architecture and emerging technologies," IEEE Access, vol. 9, pp. 67 512–67 547, 2021.

[4.12] C. Adamczyk and A. Kliks, "Reinforcement learning algorithm for traffic steering in heterogeneous network," in 2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2021, pp. 86–89.

[4.13] M. Pióro and D. Medhi, Routing, Flow, and Capacity Design in Communication and Computer Networks. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

[4.14] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges," 2022.

[4.15] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," 2017.

[4.16] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data," IEEE Transactions on Mobile Computing, pp. 1–1, 2021.

[4.17] M. Bennis and D. Niyato, "A Q-learning-based Approach to Interference Avoidance in Self-organized Femtocell Networks," in 2010 IEEE Globecom Workshops. IEEE, 2010, pp. 706–710.

[4.18] F. B. Mismar and B. L. Evans, "Q-learning Algorithm for VoLTE Closed Loop Power Control in Indoor Small Cells," in 2018 52nd Asilomar Conference on Signals, Systems, and Computers. IEEE, 2018, pp. 1485–1489.

[4.19] F. B. Mismar, B. L. Evans, and A. Alkhateeb, “Deep Reinforcement Learning for 5G Networks: Joint Beamforming, Power Control, and Interference Coordination,” IEEE Transactions on Communications, vol. 68, no. 3, pp. 1581–1592, 2019.

[4.20] R. Amiri, M. A. Almasi, J. G. Andrews, and H. Mehrpouyan, “Reinforcement Learning for Self Organization and Power Control of Two-tier Heterogeneous Networks,” IEEE Transactions on Wireless Communications, vol. 18, no. 8, pp. 3933–3947, 2019.

[4.21] Y. Wang, G. Feng, Y. Sun, S. Qin, and Y.-C. Liang, “Decentralized learning based indoor interference mitigation for 5g-and-beyond systems,” IEEE Transactions on Vehicular Technology, vol. 69, no. 10, pp. 12 124–12 135, 2020.

[4.22] F. H. C. Neto, D. C. Araújo, M. P. Mota, T. F. Maciel, and A. L. de Almeida, “Uplink power control framework based on reinforcement learning for 5g networks,” IEEE Transactions on Vehicular Technology, vol. 70, no. 6, pp. 5734–5748, 2021.

[4.23] Y. Cao, S.-Y. Lien, Y.-C. Liang, and K.-C. Chen, “Federated deep reinforcement learning for user access control in open radio Access networks,” in ICC 2021-IEEE International Conference on Communications. IEEE, 2021, pp. 1–6.

[4.24] P. Li, J. Thomas, X. Wang, A. Khalil, A. Ahmad, R. Inacio, S. Kapoor, A. Parekh, A. Doufexi, A. Shojaeifard et al., “RLOps: Development Life-cycle of Reinforcement Learning Aided Open RAN,” arXiv preprint arXiv:2111.06978, 2021.

[4.25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G.Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski et al., “Human-level Control Through Deep Reinforcement Learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015.

[4.26] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings et al., “Advances and Open Problems in Federated Learning,” Foundations and Trends® in Machine Learning, vol. 14, no. 1–2, pp. 1–210, 2021.

[4.27] M. Eskandari, S. Kapoor, K. Briggs, A. Shojaeifard, H. Zhu, and A. Mourad, “Smart Interference Management xApp using Deep Reinforcement Learning,” arXiv preprint arXiv:2204.09707, 2022.

[4.28] K. Briggs, “72.29 the billiard equations,” The Mathematical Gazette, vol. 72, no. 461, p. 217–218, 1988

[4.29] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient Learning of Deep Networks from Decentralized Data,” in Artificial intelligence and statistics. PMLR, 2017, pp. 1273–1282.

[4.30] P. Li, J. Thomas, X. Wang, H. Erdol, A. Ahmad, R. Inacio, S. Kapoor, A. Parekh, A. Doufexi, A. Shojaeifard et al., “Sim2real for Reinforcement Learning Driven Next Generation Networks,” arXiv preprint arXiv:2206.03846, 2022.

[4.31] Pollini, G. P., "Trends in handover design", IEEE Commun. Mag, p. 82-90, vol. 34, Mar 1996

[4.32] McDonald, B. Pazand and C., "A critique of mobility models for wireless network simulation", 6th IEEE/ACIS International Conference on Computer and Information Science, pp. 141--146, 2007

[4.33] Helmy, F. Bai and A., "A survey of mobility models", Chapter1-5-30-04, 2004, http://nile.usc.edu/~helmy/important/Modified-Chapter1-5-30-04.pdf

[4.34] R. Chataut and R. Akl, “Massive MIMO Systems for 5G and beyond Networks—Overview, Recent Trends, Challenges, and Future Research Direction,” *Sensors*, vol. 20, no. 10, p. 2753, May 2020, doi: 10.3390/s20102753.

[4.35] M. Dryjański, Ł. Kułacz, and A. Kliks, “Toward Modular and Flexible Open RAN Implementations in 6G Networks: Traffic Steering Use Case and O-RAN xApps,” Sensors, vol. 21, no. 24, p. 8173, Dec. 2021, doi: 10.3390/s21248173.

[4.36] J. Wang, H. Zhu, L. Dai, N. J. Gomes and J. Wang, "Low-Complexity Beam Allocation for Switched-Beam Based Multiuser Massive MIMO Systems," in IEEE Transactions on Wireless Communications, vol. 15, no. 12, pp. 8236-8248, Dec. 2016, doi: 10.1109/TWC.2016.2613517.

[4.37] J. Butler and R. Lowe, “Beam-forming Matrix Simplifies Design of Electrically Scanned Antennas,” *Electronic Design*, April 1961.

[4.38] R. Shafin, H. Chen, Y. H. Nam, S. Hur, J. Park, J. Zhang, J. Reed, and L. Liu. "Self-Tuning Sectorization: Deep Reinforcement Learning Meets Broadcast Beam Optimization," IEEE Transactions on Wireless Communications, vol. 19, no. 6, pp. 4038-4053, 2020, doi: 10.1109/TWC.2020.2979446.

[4.39] S. Chen, S. Sun, G. Xu, X. Su and Y. Cai, "Beam-Space Multiplexing: Practice, Theory, and Trends, From 4G TD-LTE, 5G, to 6G and Beyond," in IEEE Wireless Communications, vol. 27, no. 2, pp. 162-172, April 2020.

[4.40] A. Omri, M. Shaqfeh, A. Ali and H. Alnuweiri, "Synchronization Procedure in 5G NR Systems," in IEEE Access, vol. 7, pp. 41286-41295, 2019, doi: 10.1109/ACCESS.2019.2907970.

[4.41] J. Wang, H. Zhu, L. Dai, N. J. Gomes and J. Wang, "Low-Complexity Beam Allocation for Switched-Beam Based Multiuser Massive MIMO Systems," in IEEE Transactions on Wireless Communications, vol. 15, no. 12, pp. 8236-8248, Dec. 2016, doi: 10.1109/TWC.2016.2613517.

[4.42] S. P. Brooks and B. J. T. Morgan, "Optimization Using Simulated Annealing," Journal of the Royal Statistical Society: Series D (The Statistician), vol. 44, no. 2, pp. 241-257, 1995, doi: https://doi.org/10.2307/2348448.

[4.43] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," Science, vol. 220, no. 4598, pp. 671-680, 1983, doi: doi:10.1126/science.220.4598.671.

BT

CEMWORKS

IMST

interdigital

Loughborough University

NOKIA

thinkRF

University of BRISTOL

Universität Stuttgart

Vilicom
signalling the future

Bundesministerium für Wirtschaft und Energie

UKRI Innovate UK

Canada NRC·CNRC