Received June 24, 2019, accepted July 30, 2019, date of publication August 29, 2019, date of current version September 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2938410

Cell Fault Management Using Machine Learning Techniques

DAVID MULVEY[1], CHUAN HENG FOH[1], (Senior Member, IEEE), MUHAMMAD ALI IMRAN[2], (Senior Member, IEEE), AND RAHIM TAFAZOLLI[1], (Senior Member, IEEE)

[1]5G Innovation Center, University of Surrey, Guildford GU2 7XH, U.K.
[2]School of Engineering, University of Glasgow, Glasgow G12 8QQ, U.K.

Corresponding author: David Mulvey (d.mulvey@surrey.ac.uk)

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) Global Research Fund through Project under Grant EP/P028764/1.

ABSTRACT This paper surveys the literature relating to the application of machine learning to fault management in cellular networks from an operational perspective. We summarise the main issues as 5G networks evolve, and their implications for fault management. We describe the relevant machine learning techniques through to deep learning, and survey the progress which has been made in their application, based on the building blocks of a typical fault management system. We review recent work to develop the abilities of deep learning systems to explain and justify their recommendations to network operators. We discuss forthcoming changes in network architecture which are likely to impact fault management and offer a vision of how fault management systems can exploit deep learning in the future. We identify a series of research topics for further study in order to achieve this.

INDEX TERMS Cellular networks, self healing, cell outage, cell degradation, fault diagnosis, deep learning, explainable AI.

I. INTRODUCTION

The pressure to achieve greater data rates from limited radio spectrum resources is driving changes in cellular network architecture as 5G evolves. A decentralised Radio Access Network (RAN) architecture has emerged where groups of small, densely deployed cells are associated with a single macrocell, with signalling transmission retained by the macrocell but user traffic largely devolved to the small cells [1], [2] and [3]. In addition, optional interfaces between baseband and RF processing have been defined, enabling the potential virtualisation of some RAN functions.

In the new RAN architecture, Coordinated Multi-point (CoMP) transmission techniques, based on Multi User Multiple Input Multiple Output (MU-MIMO) configurations, are used to maximise radio throughput [4]. Small cells in urban settings use three-dimensional MIMO, with planar array antennas containing significant numbers of antenna elements [5].

At the same time, other work has been going on to address the topic of energy efficiency, given that the energy consumption of cellular networks is typically dominated by that of the base stations. This will require complex strategies to adjust cell coverage and turn base stations off and on again depending on traffic levels, without disruption to users [6].

These additional capabilities included in the new RAN architecture will require it to have many more configurable parameters than previous generations [7], [8] and [9], whose settings may vary according to local conditions. This will mean that the classic strategy of building a set of rules to handle faults is likely to begin to break down as the ruleset becomes too large, complex and difficult to maintain. In this situation there is an opportunity to exploit the benefits of machine learning (ML) based techniques, in particular deep learning, which do not require an explicit causal model, such as a ruleset, in order to be effective.

We define fault management as the set of tasks required to detect cell faults and then identify and implement corrective actions to restore full operation. We also include any activities required to determine the root cause of a fault, in order to take steps to prevent a recurrence. We exclude administrative tasks for tracking faults and organising remedial work from the scope of this definition.

The associate editor coordinating the review of this article and approving it for publication was Angelos Antonopoulos.

These tasks may be carried out by the network management team or may be at least partly automated. In the latter case, such features are being specified and developed for mobile networks under the banner of the Self Organising Network (SON). The EU SOCRATES project has defined use cases for the SON, leading to subsequent formalisation in the 3GPP standards [10]–[12] and [13]. The main threads within the SON are self configuration, self optimisation and self healing, all of which have some relevance to fault management, especially self healing. The top level SON standard [11] provides for both partial and full automation. In the former case, referred to as open loop SON, operator confirmation is required before control actions are implemented. In the latter case, known as closed loop SON, the system is fully automatic requiring no operator intervention.

Three approaches to fault management have been described in the literature: rule based systems, algorithmic approaches and most recently machine learning techniques.

Rule based systems have the key advantage that presenting their chain of reasoning to a user is relatively straightforward, but have the major disadvantage that they require the use of network domain expertise to set up and maintain the ruleset.

Algorithmic approaches can be very effective but lack the transparency of rule based systems and are typically limited to a narrow problem area, requiring the use of diverse algorithms to cover the complete fault management problem space, each requiring input from both network domain experts and algorithm specialists to set it up and maintain it.

Machine learning (ML) approaches, by contrast, can offer several benefits. At the root cause analysis stage, ML techniques can be used to trawl through very large volumes of fault data to suggest possible symptom-cause linkages which an expert can then review.

For real time fault management, ML techniques [14]–[16] and [17] have the advantage that they can be trained from fault data with limited domain expert input, and can then be retrained semi-automatically as the network changes. Unlike algorithmic methods, ML techniques are typically able to cover a broad range of problem areas, reducing the need for input from staff with knowledge of specialist algorithms. Deep neural networks, a recent development in ML [18], [19] and [20], are able to process very large and complex input data sets without the need for much of the dedicated handcrafted preprocessing code required by rule based and algorithmic techniques.

An issue has emerged, however, in that the most promising recent ML techniques such as deep learning, unlike earlier approaches such as Bayesian networks, do not attempt to build a casual model of the network and instead exploit correlations between data items reported by the network. Hence these techniques take no account of underlying engineering principles in arriving at their decisions. A key challenge, therefore, will be to find a way to give such systems the ability to explain and justify their recommendations.

The main contributions of this paper are as follows:

- we review and discuss the application of ML techniques to cell fault management from an operational perspective, considering fault management as a cooperative activity between the network management team and a range of electronic systems
- we propose a revised fault management lifecycle based on emerging industry practice
- we cover root cause analysis in addition to the classic self healing tasks (detection, diagnosis and compensation) covered by previous surveys
- we cover deep learning as well as earlier ML approaches and their non-ML predecessors
- we review the approaches which can be taken to enable an ML subsystem to explain its recommendations
- we propose a standard set of metrics against which to evaluate fault management systems

This paper is organised as follows. After reviewing related work, we propose a revised lifecycle for fault management aligned with the most recent operational practice, and discuss the issues in capturing the data required to support the processes within this framework. We discuss pre-ML techniques and their limitations before moving on to introduce the key ML approaches. We then survey the application of ML techniques to fault management in mobile networks, based on the key building blocks of a typical fault management system. After this, we consider the impact of recent changes in network architecture and identify the gaps between the key attributes of a future fault management system and the current state of the art. Finally we list areas for future research to close these gaps.

II. RELATED WORK

Previous surveys in fault management focus on SONs as fully automated systems, with some discussion of how ML approaches can be applied to self healing. In this paper, we also include system-operator interaction, over the full range of fault management activities as defined above including root cause analysis, taking into account the most recent developments in deep neural networks.

Aliu *et al.* [21] cover all aspects of self organisation in cellular networks, with particular emphasis on self optimisation; the authors note that relatively little work has been reported on self healing in relation to the other SON functions. The paper gives an overview of cell outage detection and compensation and provides a useful taxonomy of machine learning techniques applicable to SONs generally. This includes neural networks but coverage of these is restricted to self organising maps. Among other future research topics the paper highlights the need for work to ensure satisfactory interworking between self healing and energy saving cell hibernation schemes.

Klaine *et al.* [22] also survey machine learning applied to self organising networks, under the headings of self optimisation, self configuration and self healing. Self healing is

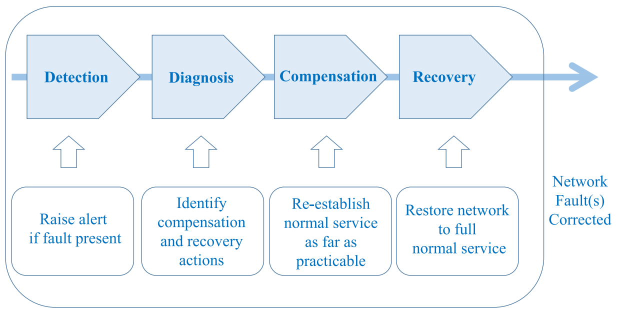
Detection
Diagnosis
Compensation
Recovery
Raise alert if fault present
Identify compensation and recovery actions
Re-establish normal service as far as practicable
Restore network to full normal service
Network Fault(s) Corrected

FIGURE 1. Self healing lifecycle.

defined to cover fault detection, fault classification and cell outage management (detection through to fault compensation). A wide range of earlier machine learning techniques is covered and discussed. There is one reference covering neural networks applied to self healing, which uses a feedforward network.

Asghar *et al.* [23], by contrast, focus on self healing and survey a broad range of techniques which have been applied to detection, diagnosis and compensation for cell outages. These include ML approaches, but discussion of the more recent techniques including NNs is limited. They raise a number of very interesting points on future challenges, mainly recommending algorithmic approaches as the way forward rather than the application of ML techniques.

All the above papers focus on closed loop rather than open loop SON. Self healing is taken to be an automatic subsystem which is able to run autonomously, and so the capabilities required to interact with a human user are not covered.

In the current paper, however, we consider fault management as a cooperative activity between the network management team and automated technologies designed to alleviate their workload. Hence we cover both open and closed loop SON, as well as support for expert investigations into underlying issues. We consider more recent techniques which are just beginning to be applied to fault management, such as deep neural networks and deep reinforcement learning. We also include the analysis of successful operator actions as an alternative perspective to fault data analysis. As part of this, we also assess the person to machine interaction issues that will need to be addressed to enable network management teams to work effectively with these new tools.

III. FAULT MANAGEMENT FRAMEWORK

A. DEFINITIONS

For the purposes of this paper, we consider a fault to be a state of the radio network or one or more of its elements which causes the network to fail to meet its service specification. For the RAN we can define the service being provided as the connectivity from the core network across the RAN to the end user's mobile device. The types of faults we consider, with typical examples, are given in Table 1.

TABLE 1. Typical examples of cell faults.

Fault Type	Example
Individual Hardware Failure	Radio frequency cable fault or damage to antenna element
Individual Software Failure	Defect in new software release
Misconfigured Parameters (Unintentional)	Radio parameter set to default rather than as required by local surroundings
Misconfigured Parameters (Deliberate Disruption)	Reconfiguration following penetration of network defences
Compensation Failure	A resilience or self healing feature compensates for a single failure such as an antenna fault but the failure is not rectified; a similar failure then occurs which cannot be compensated for
Multiple Hardware Failures	Damage to internal environment, due to eg fire, flood or theft, results in simultaneous hardware failures
Multiple Software Failures	Incorrect software release downloaded
External Coverage Impairment	Change in external environment eg new building leads to radio signal attenuation

A set of *symptoms* indicating a fault, observed at the service level and from other evidence within the network, may be due to one or more underlying *causes*, in the context of a set of network *conditions*.

Causes of faults may include hardware failures, software defects, design flaws, misconfigured parameters, incorrect actions by the network operations team and unauthorised external interventions (cyber attacks) which have succeeded in penetrating the network's security defences.

Fault symptoms may have several possible causes; an unacceptably high dropped call rate, for example, could be due to factors such as a hardware failure, an incorrect setting of a parameter e.g. antenna tilt, or even a change in the local surroundings which causes a reduction in radio signal strength.

Different types of faults may need to be handled differently. Specific unintentional misconfigurations, for example, can be fixed with the likelihood that the issue will remain resolved. Deliberate disruptions, however, may recur unless steps are taken to prevent external attack or mitigate its effects. Even in the unintentional case it may be necessary to prevent future failures by measures such as retraining or additional checks. We discuss the issue of analysing the root cause of a fault and devising preventive measures in more detail below.

At the compensation stage a failure is compensated for by a self healing or system resilience feature (such as an automatic switch from main to standby) but the failure is typically not yet rectified, so that it becomes a *dormant fault*.

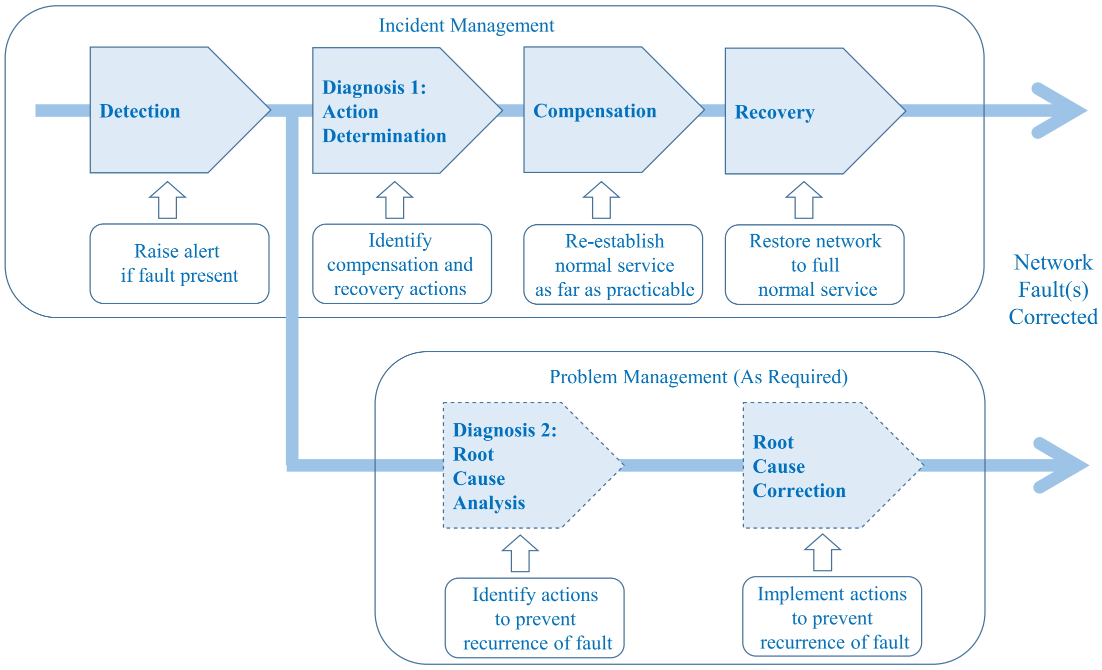
Incident Management
Detection
Diagnosis 1: Action Determination
Compensation
Recovery
Raise alert if fault present
Identify compensation and recovery actions
Re-establish normal service as far as practicable
Restore network to full normal service
Network Fault(s) Corrected
Problem Management (As Required)
Diagnosis 2: Root Cause Analysis
Root Cause Correction
Identify actions to prevent recurrence of fault
Implement actions to prevent recurrence of fault

FIGURE 2. **Fault management lifecycle.**

In this paper we consider all the fault types listed in Table 1. We assume that there is a separate operational process in place (outside the scope of this paper) for dormant faults to be logged and managed, which reports such faults to the fault management system so that they can be considered together with the presenting fault symptoms.

B. FAULT LIFECYCLE

Many of the studies we reference in this paper are based on the lifecycle for self healing systems (see Fig. 9), which focuses on the immediate activities required to get the network back into operation following a fault [24]. Recent operational fault management practice, however, has been based on a slightly different lifecycle which has a wider scope. We propose a revised and extended lifecycle, which is intended to reconcile these two approaches (see Fig. 2).

The self healing systems lifecycle described in [24] consists of a single phase of fault handling with four stages: detection, diagnosis (also known as localisation), compensation (also known as mitigation) and recovery. On completion of the fault recovery stage the self healing process is complete, in that the system has now been restored to full normal operation.

The ISO20000 fault management standard, however, which is coming to be accepted in industry, considers fault management as having two phases:

1. Incident Management, where the primary objective is to restore service following detection of a fault.

2. Problem Management, where the objective is to investigate in depth a single complex fault, or a number of apparently related faults, in order to devise suitable corrective action.

It can be seen that the lifecycle for self healing maps neatly onto that for incident management. Problem management, on the other hand, requires the addition of a second phase consisting of two stages which we may call root cause analysis and root cause corrective action.

At present, diagnosis and root cause analysis are not always distinguished clearly in the literature, although the ML approaches to the two areas may well be different. To clarify this, we propose to divide the activity currently referred to as fault diagnosis into two separate parts, with different functions, to allow the relevant ML techniques to be considered separately.

We may call the first part Action Determination, representing the diagnostic activities within Incident Management. Here the goal is simply to determine which compensation action to take given the symptoms. The second part can then be mapped on to the Root Cause Analysis stage of Problem Management. This part is potentially more demanding in that it is now necessary to analyse the fault in sufficient detail to be able to devise suitable corrective action to prevent a recurrence.

IV. FAULT MANAGEMENT DATA

In this section we consider the data required to implement the fault management framework outlined in the previous section, how this can be collected and what issues can arise. In later sections we will then go on to discuss how these issues can be addressed.

A. DATA SOURCES

In order to carry out fault detection and diagnosis, the system needs access to live data and to historic network data,

TABLE 2. Examples of key data for cell fault management.

Data Item	Granularity	Reference
Radio Reference Signal Received Power and Quality (RSRP/RSRQ) Channel Quality Indicator (CQI)	Per user	[28]
Radio Resource Control connection re-establishment requests	Per user	[28]
Call/data session connection attempts	Per cell	[29]
Call/data session set-up success rate	Per cell	[24]
Handover success rate	Per cell	[24]
Rate of incoming handovers to a given cell, as reported by neighbour cells	Per cell	[30]
Inter- Radio Access Technology handover rate	Per cell	[31]
Call/data session drop rate	Per cell	[32]
Data throughput	Per user	[24]
Number of users	Per cell	[24]
Total cell data rate	Per cell	[24]
eNode B central processing unit loading	Per cell	[29]
eNode B configuration records	Per cell	[31]

captured both during normal operation and also when a variety of faults are present. Key sources of data include alarms, other events, Key Performance Indicators (KPIs), radio coverage reports and network configuration data.

In [25], a systematic framework is put forward for defining and managing the datasets relevant to SONs, including self healing systems, which can be drawn upon to consider the dataset required for fault management. The set of standard KPIs for cellular networks defined in [26] also forms a useful starting point; however Szilagyi *et al.* highlight that additional, lower level, data is likely to be required for cell fault detection and diagnosis [27]. Typical examples of key data referred to in fault management studies are given in Table 2.

B. DATA COLLECTION

To be useful for fault management, all data needs to be logged with a timestamp and also a spatial reference, which should at least identify the cell ID. For radio measurements it is highly desirable that the spatial reference should also include the location of the mobile at the time the measurements were made [33]. Relevant aspects of the network configuration at the time of logging also need to be recorded [24].

The 3GPP standards specify a mechanism for automatic cell data collection known as the cell trace facility, which reports the data to a central trace collection entity [34], [35] and [36]. This facility provides the ability to selectively enable and disable different trace functions in different areas of the network. For example, the Radio Link Failure (RLF) function can be used to instruct a specific eNodeB to collect and report UE radio link failure messages.

The traditional method of obtaining radio coverage data is drive testing, which is, however, increasingly expensive. Consequently 3GPP set up the Minimise Drive Testing (MDT) initiative, and as a result of this work have now incorporated MDT data collection into the cell trace mechanism. This function collects UE measurements of radio KPIs such as RSRP and RSRQ, either regularly or in response to certain network events, and passes them to the cell trace facility for logging [37], [38] and [39].

At around the same time, mobile devices evolved to include a GPS location tracker, which was able to provide location data to a higher accuracy and resolution than previously possible. As well as significantly enhancing the quality of radio reporting data for fault management, it has been recognised that this can be used to improve many aspects of radio network performance including interference management, scheduling and handover decisions [40], [41].

C. DATA QUALITY ISSUES

Typical examples of low level quality issues which can arise are noise, missing data and irrelevant data. Radio data, for example, can be subject to unwanted disturbances due to shadowing and fast fading of the signal. Equipment status reports, on the other hand, may consist entirely of clean data but some reports may be lost in transmission. Even with the level of control provided by the cell trace facility, reports may include data which is not relevant to the problem being addressed. Alternatively the volume of low level data items may be too high for efficient processing, or the data may only be available in a continuous stream whereas the ML technique may require data to be submitted in batches.

At a higher level, it may not be straightforward even to detect that a fault has occurred, given the available data. This is the case with the so-called "sleeping cell" problem [42].

This scenario arises because some faults, such as RF cable failures, cannot reported to the network management centre although they may cause a radio outage. If such a fault occurs, the user service may be significantly impacted without the network management centre being aware of the problem.

In some situations, individual data items may each be weakly correlated with the occurrence of a given fault or one of its causes, but for certain combinations of data items the correlation may increase significantly. Another possibility is that some of the data items may be correlated with each other, so that the dataset contains a level of redundancy which could lead to inefficient processing.

Alternatively far too much potentially relevant data may be generated, such as when a single low level fault, e.g a power supply failure, causes multiple alarm messages to be triggered, making it difficult for the network operators to determine the underlying cause.

In the next three sections we look at how all these issues can be overcome, either by suitable pre-processing of the data or by the fault management techniques themselves. Before this, we explain why ML approaches emerged by considering pre-ML techniques and their limitations.

V. PRE-ML FAULT MANAGEMENT SYSTEMS

Pre-ML fault management techniques can be divided into two principal categories: logic based and algorithmic.

Logic based approaches use a set of rules to explicitly encode knowledge about the relationships between fault symptoms and causes. Algorithmic techniques, on the other hand, incorporate expert knowledge implicitly within the software implementing the algorithm. We discuss each of these approaches in turn and look at the limitations of both methods.

A. LOGIC BASED APPROACHES

In logic based approaches, the rules may be based on predicate logic (where predicates are either true or false) or one of a number of developments of this to allow predicates to be associated with a probability rather than a binary truth value [43].

The earliest implementations were based on a hard coded program, with access to the rules embedded in the program at appropriate points. The rules themselves were encoded in a table or defined explicitly using a rule syntax. A development of this was the expert system, which separates out control into a separate entity, the inference engine, which is responsible for selection of which rules to activate [44]. In this architecture the rules are held in a data store known as the knowledge base.

A key application of logic-based systems is to address the multiple alarm issue described in Section IV above. This uses a technique called alarm correlation, in which low level alarms are filtered and aggregated based on a ruleset, to provide a more effective presentation of the network status to the operators [45].

A further refinement is the model based approach, which separates out the behaviour of each type of network element from the network topology, and models the expected normal behaviour of each element to enable this to be compared with the actual behaviour in order to determine whether a fault exists [46].

As a sophisticated example of this approach, Yan *et al.* developed a root cause analysis toolset called G-RCA [47]. This is designed for IP networks and includes a service dependency model incorporating topological relationships as well as dependencies between protocol layers. Candidate diagnosis rules are extracted from historic data using spatial and temporal ''joining rules'' specifying the allowable gap in time or distance in space between symptoms and potential causes. The resulting rules are verified by domain experts (the network operators) and then incorporated into a causality graph which controls the diagnosis of incoming symptoms.

B. ALGORITHMIC TECHNIQUES

The logic based approach is sufficiently generic to cover a wide variety of faults. Algorithmic techniques, on the other hand, are typically designed to address one very specific issue. An example of this is the problem of compensation for radio transmit/receive array failures, where Yeo *et al.* used a genetic algorithm to optimise radio performance of the failed array, and subsequently improved on this approach by using a particle swarm optimisation algorithm [48] and [49]. Closely related to this is the problem of compensation for cell outages, for which some recent examples of algorithmic techniques again include particle swarm optimisation and also use of a proportional-fair utility algorithm [50] and [51].

C. LIMITATIONS

Logic based systems have proved effective in use but suffer from a number of serious limitations. Strict application of binary logic has been found to result in too many special conditions and so it has become necessary to group together various sets of symptoms and use probabilistic logic. Even so, rule bases can grow to the point where maintaining consistency becomes a major issue [43]. Given the complexity of 5G and the expected numbers of parameter settings, it may prove infeasible to use rule based systems at the level of detail required for effective diagnosis.

Expert input is required to set up and maintain the rule base and this can be scarce and expensive; the expert may not necessarily be able to articulate their knowledge so knowledge capture can be challenging [52]. Even the more recent systems with automated extraction of candidate diagnostic rules can require significant input from domain experts and software specialists to verify the logic initially and to maintain it as changes are made to the network.

Logic based systems require the same pre-processing techniques to handle low level data issues as for ML systems (see Section VII below), and in addition may also require extensive domain and problem specific data conversion routines at the front end to turn complex analogue measurements, such as comparison against a profile, into simple predicates for processing by the ruleset.

Algorithmic approaches entail similar pre-processing overhead, together with significant expert input to code, set up and maintain the fault management subsystem. In addition such techniques are typically designed to solve one particular issue and do not generalise to other issues, which may lead to a large number of different low level software modules to be supported.

VI. OVERVIEW OF MACHINE LEARNING FOR CELL FAULT MANAGEMENT

By contrast with traditional logic based techniques, ML approaches can automate much of the work of setting up and maintaining the fault management system, so that expert input is only needed to validate the system rather than to specify all the details. Unlike algorithmic approaches, ML techniques can in many cases be used to handle a range of issues rather than being crafted to address one specific problem.

This section provides an overview of selected ML techniques which have been used in cell fault management studies or have shown potential for cell fault management from their

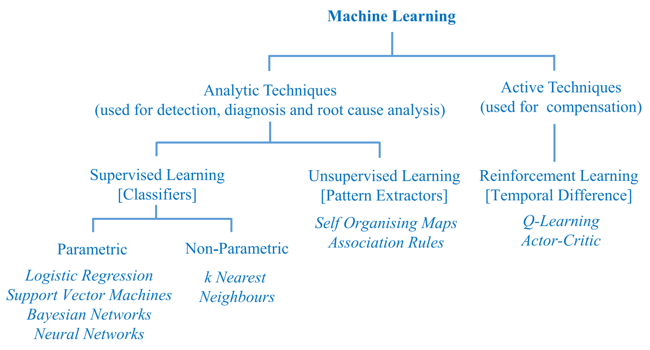
Machine Learning
Analytic Techniques (used for detection, diagnosis and root cause analysis)
Active Techniques (used for compensation)
Supervised Learning [Classifiers]
Unsupervised Learning [Pattern Extractors]
Reinforcement Learning [Temporal Difference]
Self Organising Maps
Association Rules
Q-Learning
Actor-Critic
Parametric
Non-Parametric
Logistic Regression
Support Vector Machines
Bayesian Networks
Neural Networks
k Nearest Neighbours

FIGURE 3. Taxonomy for ML techniques in cell fault management.

successful application in similar work. We also critically evaluate the advantages and disadvantages of ML approaches in relation to previous approaches.

For the purposes of this paper we use Murphy's definition of ML: "we define machine learning as a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision-making under uncertainty" [53]. We can divide ML techniques applicable to fault management into two types. The first uses analytical techniques, where the system provides useful information derived from a raw data set [53], [54]. The second employs active techniques, where the system takes actions subject to a feedback and reward system [55]. To date, detection and diagnosis have been carried out using purely analytical approaches whereas active techniques have been exclusively applied to the compensation stage.

A taxonomy diagram for the principal ML techniques which have been used in cell fault management studies is given in Fig. 3.

A. ANALYTICAL TECHNIQUES

1) INTRODUCTION

All analytical ML techniques use two principal input data sets: *training data* made up of historical data, from which learning can take place, and active *live data* samples processed by the system when in operational use. In the ML world, the attributes of the input data are referred to as *features*; hence the *dimensionality* of the data is the number of features.

We can consider the analytical techniques as having the following attributes, which are described in more detail below:

- output type (continuous, discrete)
- supervision mode (supervised, unsupervised)
- training method (parametric, non parametric)
- scope (global, local)

If the output type of an ML system is continuous, being used to predict some property derived from the input data set, the system is known as a *regression* system. By contrast, systems in which the output type is discrete, so that each output represents a class to which each input has been assigned, are referred to as *classification* systems.

The supervision mode is dependent on the composition of the training dataset. In *supervised learning* the training data set includes values for the output data as well as the input data; these will be either predicted values in the regression case or class labels in the classification case. In *unsupervised learning* no predictions or labels are provided; the system uses input data only.

The training method used may be *parametric* or *non parametric*. A parametric method fits a model to the training data during a *training phase* by adjusting the model parameters to minimise a suitable cost metric. The model is then used during a subsequent *operational phase* to predict from or classify live data. A non parametric method, on the other hand, uses the training data directly during the operational phase rather than learning a predictive or classification model beforehand.

The scope of either method may be *global*, in which case the algorithm takes as input the whole of the training dataset and any parameters are constant across the whole data range, or *local*, in which case the algorithm considers limited regions of the data space at a time and any parameters have local validity only, typically with a method of minimising discontinuities at the borders between the regions.

Methods may be based on purely *linear* calculation techniques, or may in addition include *non linear* mathematical approaches. An important class of model based approaches which combines both of these is the neural network (NN). NNs consist of a set of nodes, each of which applies a specific linear weighting to each of its inputs and then may apply a non-linear transformation to compress the result. Nodes are typically organised in layers, providing input and output and also often including internal or hidden layers. Recent general advances in NNs have focused on so-called deep NNs, which for the purposes of this paper we may define as NNs with two or more hidden layers [56].

A good example of the NN approach is the feedforward NN (FFNN), typically used in cellular networks as a classifier. This is trained by optimising the weights, using both the forward and the backward paths through the network, to minimise a "loss function" giving a measure of the difference between the labelled classification and that predicted by the NN. During the operational phase, classification then takes place using the forward path only.

An FFNN is unsuitable for processing input sequences as it can only consider a fixed set of inputs at a given time. This limitation would mean that an FFNN would be restricted to processing fixed length sequences and the number of input weights required would be the product of the number of features and the sequence length. To overcome this, the recurrent NN (RNN) feeds back the values of the states of each hidden layer and weights them to include in the calculations for the new state values for that layer for the next item in the sequence. This allows the weights to be shared between all items in the sequence and permits the processing of sequences of arbitrary length. As with the FFNN, the RNN is trained using labelled data.

A convolutional NN (CNN) by contrast, is designed to process two dimensional inputs, typically extracted from image data. As with an FFNN, a convolutional network consists of an input layer, an output layer and a number of hidden layers. The CNN, however, consists of two principal types of hidden layer: the convolutional layer and the pooling layer.

The convolutional layer carries out a set of weighted convolution operations on small subsets of the layer's inputs, using the same weights for each subset. The pooling layer, on the other hand, takes the outputs from the convolution layer and aggregates then across larger subsets of the data. This has the effect of making the results insensitive to particular aspects of the image such as the exact location or orientation of an item within the image. Repeating these processes over many hidden layers allows the CNN to learn its own features, such as lines or angles, which can become progressively more complex in the later hidden layers. Just as with the FFNN and the RNN, the CNN is trained using labelled data.

All the NNs described so far are typically used to predict the classification of data items, based on labelled training examples. The autoencoder, however, is designed to predict its own inputs. This can be useful when dealing with noisy inputs, when a so-called denoising autoencoder [57] can be used to recover a clean version of the inputs. The network consists of an encoder and a decoder, and is trained using a loss function providing a measure of the difference between the actual input and the autoencoder's prediction of it. For more detail on recent developments in neural networks applicable to wireless networks see [18], [19] and [20].

At present, relatively little use of neural networks has been made in cell fault management, although an FFNN has been used for cell outage detection [58]. We discuss later on in this paper how deep NNs can be used in cell fault management and the challenges that will need to be overcome in order to achieve this.

2) SUPERVISED LEARNING - CLASSIFIERS

Detection and diagnosis techniques typically make use of classifiers based on supervised learning; binary classifiers are sufficient for detection where there are only two possibilities, faulty or not faulty, whereas for diagnosis, where there may be several possible causes, multiclass classifiers are required.

A comparison of the major classifiers used in fault management is given in Table 3. The simplest is logistic regression, a parametric linear classifier, which finds a hyperplane to be used to separate the data, based on the minimum total squared distances from all the data points. It can be extended to permit non-linear boundaries by calculating new features which are polynomial or other functions of the original features.

The support vector machine (SVM) is also a parametric classifier but includes an internal non-linear transformation which allows it to handle non-linear boundaries. The SVM's distinctive feature is that it sets the boundary by taking into account just the points close to where the boundary is expected to be, referred to in the literature as the *support vectors*. These points are identified in relation to a specified margin on either side of the boundary. A refinement is to set a budget for misclassification errors (points deliberately allowed to be on the wrong side of the boundary or in the margin). The position of the boundary is then adjusted by an optimisation function to minimise the classification error subject to this budget.

The *k* nearest neighbours (*k*NN) method is a non-parametric approach. When used as a classifier, *k*NN requires training data to be gathered from normal and faulty operation, with the data labelled to distinguish between normal operation and each type of fault. It then classifies each live data point by majority voting based on the labels of its *k* nearest neighbours in the training set. A recently reported technique [59] is the Transductive Confidence Machine (TCM), which can be thought of as a variation of *k*NN which also uses a labelled

TABLE 3. Principal ML techniques 1: Classifiers.

Technique	Strengths	Limitations	Application
Logistic Regression	Binary and multiclass where data linearly separable, low usage of computing resources	Not able to handle intermingled classes	Detection or Diagnosis
Support Vector Machine	Binary classification based on small training data sets, limited intermingling between classes	Difficult to extend to multi-class; computationally expensive for larger training sets	Detection
k Nearest Neighbours	Binary or multiclass classification with low dimensional input	Requires access to training set during operation, computation resource requirement grows with training set size, poor performance with high dimensionality input	Detection or Diagnosis
Naive Bayesian Classifier (NBC) with Bayesian Network (BN)	Diagnosis in situations where fault causes understood and historic data available. Low computing resource required	Assumes that only one cause is present at a time and that the symptoms are independent given the cause. Requires significant expert input to set up and maintain BN	Diagnosis
Neural Networks (NNs)	Can handle multiple input features without requiring extensive preprocessing	More complex NNs may require intensive computing power and large datasets to train	Detection and Diagnosis

TABLE 4. Principal ML techniques 2: Pattern extractors.

Technique	Strengths	Limitations	Application
Self Organising Maps	Mapping a high dimensional dataset onto a 1D or 2D discrete data set representing data clusters. Computing resource requirement depends on data set size, dimensionality and number of clusters.	Additional cluster validation algorithms and expert input required.	Diagnosis
Association Rules (FP-Growth algorithm and modifi-cations)	Extraction of associations (eg symptom-cause) from historic fault data.	Unmodified algorithm requires frequent associations but modified version can work with rare associations (see Root Cause Analysis below).	Diagnosis

training set. There is also a type of kNN which can be used as anomaly detector. This method uses a training data set representing normal operation; for each live data point the system calculates a metric based on the distances from the k nearest neighbours in the training dataset and compares it with a threshold in order to detect anomalies.

The naive Bayesian classifier (NBC) works with a Bayesian Network representing symptom-cause relationships derived from historic fault data. It uses Bayes' theorem to rank the possible causes by probability given the symptoms. Expert input is required to set up the network but the cause probabilities can be estimated automatically if sufficient data is available.

An NN used as a classifier, irrespective of the number and type of the hidden layers, typically will have an output stage designed to estimate the probability of each input example being in each of the classes and use this to make a classification decision.

3) UNSUPERVISED LEARNING - PATTERN EXTRACTORS

All the above techniques depend on a training set with every data item labelled individually, which can require a large amount of expert input. As a result unsupervised learning techniques have been developed (see Table 4), which automatically extract a small number of patterns from large quantities of data. The candidate patterns can then be reviewed by experts, significantly reducing their workload in comparison with searching the data manually. Having done this the classifier can then classify the data by comparing it with the patterns. Although these approaches are typically somewhat computationally intensive during the training phase, none of them requires significant computing resource during the operational phase, as only the classifier needs to be run at this point.

One approach to pattern extraction is *cluster analysis*, which aims to group the data into similar types or clusters. An approach which has been used in fault diagnosis is self organising maps [60], which are a type of neural network that projects a high dimensional training data space onto a very low dimensional (typically 1 or 2D) discrete data space representing a small number of clusters.

Another approach, which has been used in the root cause analysis of faults, is to use an *association rule* extractor algorithm [61] to scan network event logs and traces to automatically detect possible associations between a given set of symptoms and potential causes, together with measures of the strength of the associations. The associations are then expressed as a set of rules or a causal graph for review by a human expert.

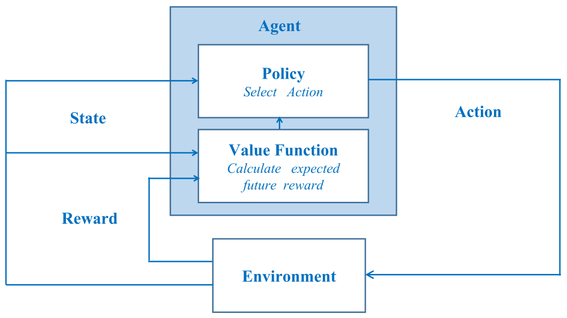

FIGURE 4. Reinforcement learning architecture.

TABLE 5. Principal ML techniques 3: Reinforcement learning.

Technique	Strengths	Limitations	Application
Temporal Difference Q-Learning with Fuzzy Logic	Model of underlying system (radio network) not required for optimisation	Relatively slow to converge	Compensation
Temporal Difference (Actor-Critic)	As above	Slightly faster to converge than Q-Learning (see Compensation section below)	Compensation

B. ACTIVE TECHNIQUES

These are based on the principle of *reinforcement learning* (RL), in which a part of the system known as the *agent* (see Fig. 4) takes actions on the *environment* from which it receives reward signals, which are used to influence the next action.

The agent's actions are determined by a *policy,* which has to balance the benefits of *exploitation*, in other words taking the action with the current highest expected future reward, against *exploration,* which means taking another action with a lower currently expected future reward in order to seek even higher rewards in the future. To assist with this, another part of the system known as the *value function* is used to estimate the total expected future reward of taking an action from a given system state. For a task such as compensation, where the goal is to achieve a stable result, the agent's interaction with the environment can be represented by a finite sequence called an *episode*.

The two most popular RL techniques used in cell fault management are Q-learning [62] and Actor-Critic learning [33]. These have each been used to compensate for cell outages by adjusting the power levels and/or antenna tilt of neighbouring cells. They are both based on the Temporal Difference (TD) approach, in which the system evaluates the expected reward by looking ahead one step only. This gives the TD approach the advantage that it is not necessary to provide it with a model of the environment, as it can function by taking one action at a time and observing the outcome.

In Q-learning, the policy is fixed. A typical policy might specify that the system should normally take the action with the highest calculated future reward (exploitation), but in a small proportion of cases it should try another action (exploration). At each step of the process the value function for the action just taken, denoted by Q, is updated based on the feedback received from the environment. Hence Q is learned from experience as the system explores different actions, and at each step the most recent value of Q influences the system's behaviour via the policy.

In the Actor-Critic approach, by contrast, the system modifies the policy according to experience. The policy consists of a state-action table specifying the required probability of taking each action in a particular state. The critic forms an error signal comparing the outcome of a given action with the expected reward, which the actor uses to adjust the probability for this action in the policy table. So if the outcome of a particular action is positive, the probability of taking it again will be increased, but if the outcome is negative it will be decreased.

A key limitation of these RL techniques is that the size of the state-action table is proportional to the product of the number of system states s and the number of possible actions a which can lead to scalability issues. To address this, the deep RL technique introduces a deep neural network to carry out the mapping from states to actions [63]. The basic RL technique is then used to train the neural network to identify the action with the highest reward for each state, based on the experience of the RL subsystem. In order to average out the effect of specific conditions during a given episode, the state-action-reward data for each episode is stored in a replay memory to enable training samples for the neural network to be drawn randomly from multiple episodes.

C. ADVANTAGES, DISADVANTAGES AND CHALLENGES

In comparison with algorithmic solutions, ML approaches have the following general advantages and disadvantages.

Advantages:

1) built-in ability to process a much larger number of input features
2) can be retrained automatically eliminating the need for manual retuning
3) can be applied to a range of issues using standard libraries hence reduced need for specialist algorithm expertise
4) reduced dependence on details of specific problem also reduces the need for domain knowledge
5) deep learning techniques can also dispense with much of the preprocessing code required by algorithmic methods and earlier ML approaches

Disadvantages:

1) significant volumes of training data required; collection of sufficient fault data may be a substantial organisational challenge

In relation to logic based systems, similar advantages and disadvantages apply, with the following additions:

Advantages:

1) the ability to function without an explicit causal model removes the difficulty of maintaining consistency of a rule base as the problem domain becomes more complex

Disadvantages:

1) significantly more difficult to present reasoning in support of recommendations

Hence the key challenges to overcome in support of the introduction of ML techniques are:

1) systematic collection of fault data
2) development of the ability of ML systems to explain and justify their recommendations

The first of these is primarily an organisational rather than a research issue and can be left to mobile network providers. The second has been recognised as a key blocker to progress and intensive research is now under way to address this, as we discuss below.

D. SUMMARY

In this section we have described the principal ML techniques which have been used in cell fault management, and for each group of techniques we have explained in broad terms which activity within fault management the techniques are most applicable to. We have covered at a high level the advantages, disadvantages and current challenges with ML techniques. In the next section we will drill down to look in more detail at the application of each technique to specific fault management activities and the specific strengths and limitations of each approach.

VII. APPLICATION OF MACHINE LEARNING TO FAULT MANAGEMENT IN CELLULAR NETWORKS

To date, much of the work on application of ML techniques to cellular network fault management has concentrated on the

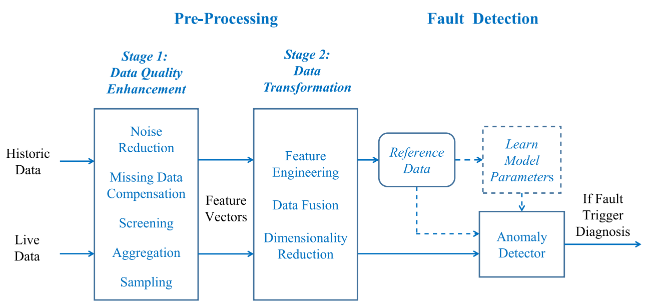

FIGURE 5. Preprocessing and fault detection.

"sleeping cell" problem referred to in Section IV above, and related cell performance degradations [42] and [27]. Some recent work, however, has looked at more general faults in mobile networks [61].

Between them, the studies surveyed have covered detection, diagnosis (action determination and root cause analysis) and compensation. For implementation in the network environment, all techniques require a further stage at the beginning which we can call pre-processing. In this section we review the ML techniques used by these studies, considering each stage in turn. For ease of reference, Table 13 at the end of this section lists all the studies presented here by stage(s) covered.

A. PRE-PROCESSING

The purpose of the pre-processing stage is to address the low level data quality issues identified in Section IV above and to transform the data into a form which can be utilised efficiently by the detection stage.

From the perspective of machine learning, we can identify two stages of pre-processing, as shown in Fig. 5. The purpose of stage 1 is to reduce data volume while improving the quality, and present the input data as a series of feature vectors as required by the ML subsystem. Stage 2, on the other hand, transforms the features for optimal processing by the ML subsystem.

1) STAGE 1 - DATA QUALITY ENHANCEMENT

Stage 1 techniques are used to address the low level data quality issues identified in Section IV above. Tailored digital filtering techniques may be used for *noise reduction*. Missing data can be handled by *missing data compensation*, in which dummy or interpolated data is used to fill gaps which would otherwise disrupt processing. Specific *screening* code may be implemented to remove irrelevant data. *Aggregation* techniques such as counting or accumulation of data values over a set period can be used to reduce the volume of data [29]. *Data sampling* is used to transform an unlimited input time sequence into a finite set of vectors to enable the ML subsystem to treat the inputs as samples from a larger population. Typically a sliding window is used to capture successive sets of samples over a fixed time period [64].

TABLE 6. Dimensionality reduction techniques in fault management.

Technique	Summary	Strengths	Limitations	References
Multi Dimensional Scaling	Matrix method which seeks to minimise changes in distances between data items	Relatively simple algorithmic approach available	Mapping hard for domain experts to interpret	[33], [65]
Diffusion Maps	Technique equivalent to lossy compression applied to graphical representation of the data	Preserves local distances	As above. Complex algorithm requires specialist input to maintain	[66]
Graphical Techniques	Represent relationships between data items in compact form, such as a neighbour relations graph	Mapping can be readily interpreted by domain experts	Significant expert input or complex auto discovery code required to set up and maintain mapping	[42]

2) STAGE 2 - DATA TRANSFORMATION

Examples of Stage 2 techniques include *feature engineering*, *data fusion* and *dimensionality reduction*.

The aim of feature engineering is to derive new features from the input data which can improve the performance of the ML subsystem or allow a technique to be used in situations where it would not otherwise be applicable. One example of feature engineering is the use of a Fast Fourier Transform (FFT) to detect periodic variations in a time sequence [29]. Another example is where polynomial terms are formed from the basic features, so that a non-linear boundary in the input feature space can be transformed into a linear boundary in the polynomial space.

Data fusion, on the other hand, addresses the situation where any individual data item is weakly correlated with the occurrence of a fault or one of its causes; by combining two or more data items it may be possible to produce a feature with higher correlation than any of its components.

Dimensionality reduction is needed where the number of input dimensions is sufficient to cause degradation of the ML system performance. It is particularly useful for removing redundant information in the case where different features are partially correlated with each other. The goal is to reduce the number of features while retaining as much of the key information from the input as possible. Three key techniques which have been applied to fault management are listed in Table 6.

At the current state of the art, the preprocessing phase requires a significant level of hand coding to tune the front end to match the input data to the ML technique being used.

This typically requires scarce specialist effort and can limit system flexibility in response to change.

B. DETECTION

The purpose of the detection stage is to determine whether a fault is present or not, without committing significant resources to diagnosis and compensation until a reliable decision has been made.

There are now many instances where ML approaches, both parametric and non parametric, have been proposed for use in detecting faults in cellular networks. These assume as a minimum that a set of data is available representing normal operation, against which anomalies representing faults can be detected. All these techniques operate at the correlation level, in other words they are not dependent on the availability of a causal model of the network.

A sleeping cell situation arises where a radio failure is not being reported to the network management system. Hence sleeping cell failures have to be detected indirectly, using related evidence such as radio signal strength, channel quality indicators and higher level indicators such as incoming handovers and dropped call rates. Similar data may be used to detect other anomalies, such as misconfigured parameters which impact the extent and quality of the radio coverage.

Parametric techniques described in the literature are listed in Table 7. In Time Domain Prediction, a KPI is compared with a predicted value at each time step. Three examples of this approach are network calculus, Auto Regressive Integrated Moving Average (ARIMA) modelling, and Grey modelling. In each case previous data from the sequence, representing normal operation, is used to learn the settings of the model parameters which minimise the prediction error.

Comparison of Statistical Distributions takes place between the live KPI data and a reference data set representing normal operation. To achieve this, it is necessary to fit a statistical distribution to the normal reference data set then either: (a) compare live KPI data directly with the stored distribution to generate a normalised ''KPI level'' representing the degree of abnormality of the relevant KPI or (b) fit a second distribution to the live data then either compare parameters with the stored distribution or compare the distributions directly.

Parametric Binary Classification is based on a labelled training dataset including both normal and fault data. Two approaches to achieve this are described in [42]. The first uses the Classification and Regression Tree (CART) technique to recursively partition the data into normal and anomalous regions to be used for classification of live data. The second, by contrast, uses a Linear Discriminant Function (LDF) to learn the parameters of a hyperplane to be used to separate normal and anomalous data.

In Anomaly Boundary Setting, a boundary is set between normal and anomalous data on an unlabelled normal training data set by excluding a specified number of outliers. The boundary is then used to classify live data as normal or anomalous. In [65] and [68], this is done by using a one class Support Vector Machine (SVM) which works generally as described in Section VI above. In the specific case of the one class SVM, the budget is used to allow a small number of outliers in the normal data to be misclassified as faults, in order to achieve optimum anomaly detection performance on live data. In [75], anomaly boundary setting is done by fitting a Gaussian distribution to normal data.

TABLE 7. Parametric approaches to fault detection.

Technique	Summary	Strengths	Limitations	References
Time Domain Prediction	Prediction using a time domain model trained on labelled historic data	Can provide short detection times	Requires significant expert input to tune and retune models	[33], [67], [68], [69]
Comparison of Statistical Distributions (SDs)	Live data compared with SD learnt from training data, either directly or via SD fitted to live data	Can relearn SDs following a change; scalable for large networks	Potentially slow response time if need to build up SD of live data to achieve accuracy	[27], [64], [68], [70], [71], [72], [73], [74]
Anomaly Boundary Setting	Set a boundary using unlabelled training data, excluding outliers, and use to classify live data	Expert input required only to review boundary	Risk of false positives due to new normal points beyond boundary	[65], [68], [75]
Neural Networks	NN trained on labelled historic data then deployed as classifier during operational phase	Can be economical in use of computing resources during operational phase	Typically requires intensive use of computing resources during training	[58]

With NNs used as classifiers, the network weights are optimised by against labelled normal and fault data. Once trained, the NN can then be used to classify incoming live data.

Feng *et al.* used a feedforward NN as a classifier in a cell detection scenario; they encountered difficulties due to the system becoming trapped in non-optimal local minima during training, degrading system accuracy. This was resolved by using a ''differential evolution'' algorithm as the NN optimiser [58].

Non-parametric techniques described in the literature are listed in Table 8. In non-parametric Binary Classification, anomaly detection is again treated as a binary classification problem using labelled training data representing normal and faulty operation. In this case, however, each live data point is

TABLE 8. Non parametric approaches to fault detection.

Technique	Summary	Strengths	Limitations	References
Binary Classification	Classify based on classes of nearest neighbours in labelled training set	High accuracy if large volume of training data	Requires fault data plus intensive expert input for labelling	[59], [66], [76], [77]
Distance or Density Comparison	Calculate distances between live data and points in an unlabelled training set representing normal operation	Can set anomaly threshold using unlabelled data set, reducing need for expert input	Risk of false positives due to new normal points above threshold	[65], [78], [79], [80]
Direct Cluster Analysis	Form clusters from unlabelled training set, classify clusters	Expert input required only to review threshold	Cluster allocations may not reflect physical realities	[81], [82]

classified as normal or anomalous by identifying its *k* nearest neighbours in the training set and then classifying it based on the majority of the neighbour classifications.

Distance or Density Comparison uses a training data set representing normal operation only. In this method, anomalies are detected either: (a) by computing some function of the distances from the *k* nearest neighbours in the training set and comparing this with a threshold or (b) by calculating the local data density (the number of points in unit volume of the feature space) for each live data point and comparing it with the global average density, or the average density of its *k* nearest neighbours.

Onireti *et al.* (see Table 12 below) report that in a macrocell scenario, the global *kth* nearest neighbour approach was more accurate than a local density based method [33].

Direct Cluster Analysis employs an unlabelled training data set representing both normal and faulty operation. This technique forms clusters from the training data and classifies the clusters (as opposed to the raw data) as normal or anomalous using other KPIs which differ in value between normal and anomalous operational states. Anomaly detection is then carried out by comparing the distances of each live data point to the normal and anomalous clusters.

The two types of training method have different strengths and weaknesses from a RAN deployment perspective. Parametric techniques require access to a central database during the initial training phase. Once the parameters have been learned, however, the system can in principle be deployed in the RAN without the need for further access to central data. Non parametric techniques, on the other hand, do not require an initial training phase but during live operation do require access to a central historic fault database.

TABLE 9. Approaches to fault diagnosis 1: Action determination.

Technique	Summary	Strengths	Limitations	References
Network Based with Causal Model	Build a model of network symptom cause relationships and use to identify the most likely cause of a given fault	Relatively simple to explain the output	For complex networks may become infeasible to build and maintain causal model	[83], [70], [84], [85], [86]
Network Based without Causal Model	Work directly with network symptom cause data exploiting correlations between symptoms and causes	Has the potential to scale to very large and complex networks	Difficult to provide rationale for output	[27], [66], [82], [87], [60]
Operator Action Analysis	Analyse operator action record to directly identify successful actions in response to given symptoms	No need to gather network data; can quickly devise strategies to correct most common faults	May be difficult to identify whether action successful or not; may be slow to adapt as network changes	[88]

C. DIAGNOSIS 1: ACTION DETERMINATION

Upon detecting a symptom, the task of action determination is to identify appropriate compensation actions in order to restore normal service as far as possible. In earlier studies, there was an attempt to construct a detailed causal model to support this. In more recent work, however, there has been a trend away from this towards a "black box" approach working at the correlation level. Approaches described in the literature are listed in Table 9.

A popular approach to diagnosis from the earliest studies onwards is to use the Bayesian Networks/Naive Bayesian Classifier method described in Section VI above. In this approach, typically the expert is needed at the start to define the logical relationships from which the network is built, but the probabilities required can then be extracted from historic data, if this is available [70], [83]–[85] and [86].

The use of the Naive Bayesian Classifier assumes that only one cause is present at a time and that the symptoms are independent given the cause. The studies acknowledge that this is likely to be unrealistic for some faults in an actual network but nonetheless report acceptable diagnostic performance for the scenarios studied.

Symptoms may be presented to the classifier in continuous form or alternatively they may be discretised first using one or more thresholds to generate binary values. The threshold levels can be set automatically using an ML technique called Entropy Minimisation Discretisation (EMD) [83].

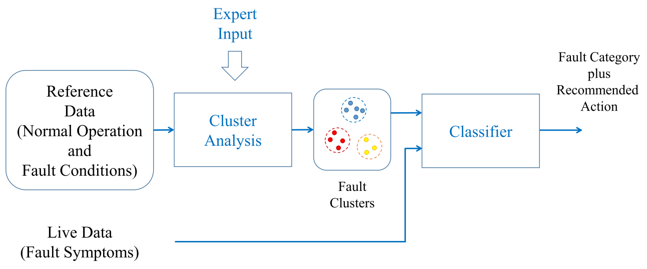
Expert Input
Reference Data (Normal Operation and Fault Conditions)
Cluster Analysis
Fault Clusters
Classifier
Fault Category plus Recommended Action
Live Data (Fault Symptoms)

FIGURE 6. **Action determination using cluster analysis.**

Barco *et al.* conclude that the continuous approach is preferable if large fault data sets are available, whereas if only small data sets are available the discrete approach should be used [85]. Another method is to retain the conditional probability calculations from the BN approach while relaxing the requirement to build an explicit causal model. In contrast with the BN approach to symptom data, Szilagyi *et al.* begin by deriving a KPI level, which is a standardised measure of the deviation of the current KPI value from that for normal operation [27]. The system then calculates the likelihood that the cause is present given each symptom (based on historic relative frequencies), and multiplies this by the KPI level to give a score for each candidate cause.

A more radical option is to directly classify the symptoms from historic data, in which each symptom is labelled with a cause, but without constructing a causal model. In [66], the k nearest neighbours approach is used to classify incoming symptom sets based on the classes of their k nearest neighbours in the training set.

More recently a hybrid approach has been put forward (see Fig. 6), which is to carry out a cluster analysis on the symptoms first, then use a network approach to relate the clusters of symptoms to potential causes. Ciocarlie *et al.* used a Hierarchical Dirichelet Process for the cluster analysis and an Markov Logic Network for classification [82]. They set up the network manually after which the system learnt the weightings from a training data set based on maximum likelihood estimation. Gomez-Andrades *et al.* used a self organising map to carry out the cluster analysis [87] and [60]. This maps a dataset of continuous data to a set of discrete points representing the clusters. After a degree of automatic quality checking, the clusters are verified by an expert before being used for classification of live data. Although this does require a degree of expert input, the effort required is very much less than if the clustering had not been carried out first.

A recent paper related to cellular networks, however, has taken a radically different approach to action determination [88]. The aim here is to automatically learn service management policies and rules for triggering compensation actions, from historic logs of faults and related operator actions. Symptoms are detected from anomalies in a rolling time sequence of key KPIs and associated with successful actions occurring within a time window of the anomaly; a logistic regression classifier is then trained from this data

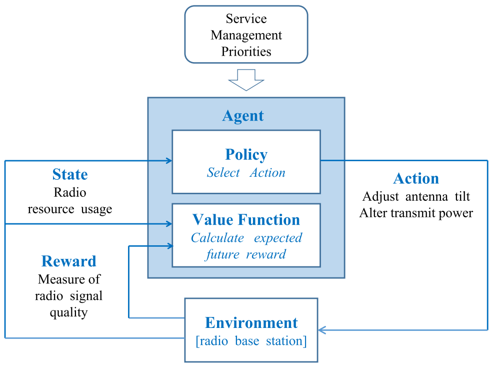

FIGURE 7. **Compensation using reinforcement learning.**

TABLE 10. **Approaches to compensation.**

Technique	Summary	Strengths	Limitations	References
Q Learning	Fixed policy: normally take action with highest expected reward but sometimes try different action; expected reward per action updated at each step.	Able to learn from experience without the need to build an explicit model of the network	Relatively slow to converge. Either need to learn on part of the actual network where learning can be tolerated or against a simulator	[89], [62], [90]
Actor-Critic	Policy updated at each step; probability of taking a given action adjusted to reflect the outcome when it was last taken.	As above	Faster to converge; same learning issue as above	[33], [91]

and used to classify new symptoms according to the action required. No attempt is made to determine the cause; the system operates at the correlation level and only considers what action previously resolved the problem. It is critical to this approach to consider only successful operator interventions based on the subsequent outcome; in some cases expert review of the historic logs is likely to be required to determine which these are.

D. COMPENSATION

The aim of compensation is to restore the best possible level of service given the remaining serviceable network resources, according to priorities set by a policy specified by the network operator.

Much of the work to date has concentrated on compensation for cell outages. For this scenario, the most popular approach is to adjust the downlink/uplink power levels and antenna tilt settings of the neighbouring base stations [92].

Machine-learning techniques utilising this method (see Fig. 7) have been based on the Temporal Difference (TD) approach to RL. Within this, two approaches have been taken:

1. Q-learning with fuzzy logic: [62], [89], [90]
2. Actor-Critic learning [91] and [33].

With the Q-learning approach, a convergence time for outage compensation of 1000-1500 steps of 200ms has been reported, giving a convergence time of 200-300 seconds [90]. Using the Actor-Critic approach, however, Onireti *et al* achieved convergence within 500 steps of 1ms each, which is around 0.5 seconds in total [33]. A key limit in the latter case was the LTE Transmit Time Interval of 1ms, as this determines how frequently actions can be carried out and feedback received from the network.

Deep RL has been used for self optimisation in radio networks [93] and [94] and is starting to be used in fault management. Mismar and Evans [95], for example, working with a cellular radio network, used the Q-learning approach with a deep feedforward network in order to select from a list of predefined fault handling actions given one of a set of alarms indicating the fault type.

A key issue for the introduction of active techniques such as RL and deep RL into live cellular radio networks is how to allow them to build their experience without disrupting the operation of the network. Two strategies would seem possible: train the system on the live network or train against some form of simulation of the network.

The first approach has the advantage that the system can learn under the precise conditions in which it will operate, but for operational reasons it may be necessary to restrict the impact of any learning action the RL system is permitted to take. At present, however, it would appear that limited work has been done to investigate the effects of constraints on the range of explorative actions during learning.

The second approach at first sight addresses the issue of impact to the network but in this case it may be difficult to ensure that the simulation can be configured to be a realistic representation of any given location on the actual network. As a result there may be a gap between the simulator and the network requiring at least a degree of retraining on the live network. This leads to two challenging research opportunities. The first is to investigate the interplay between constraints on explorative action and RL performance and work out how this can be optimised. The second is to investigate how learning on a simulated environment which is representative of typical network conditions can be generalised so that the RL system can apply it to specific network locations with minimum retraining.

E. DIAGNOSIS 2: ROOT CAUSE ANALYSIS

The purpose of root cause analysis is to investigate the underlying causes of an issue and devise suitable action to prevent a recurrence. A key challenge is that such causes may not be documented and the underlying network behaviour may not be understood by more than a very few experts; it may even be the case that some aspects of network behaviour are not understood by anyone [96].

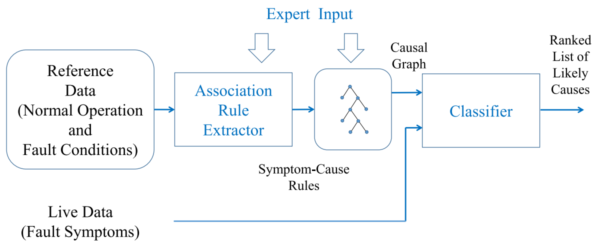
Expert Input
Reference Data (Normal Operation and Fault Conditions)
Association Rule Extractor
Causal Graph
Classifier
Ranked List of Likely Causes
Symptom-Cause Rules
Live Data (Fault Symptoms)

FIGURE 8. Root cause analysis using association rule extraction.

TABLE 11. Approaches to fault diagnosis 2: Root cause analysis.

Technique	Summary	Strengths	Limitations	References
Association Rule Extraction	Scan network event logs to detect associations between symptoms and causes	Initial symptom cause model not required	Currently network topology not presented separately from symptom cause relationships so may be difficult to track network changes	[61], [96]

Another key challenge is that, in comparison with fault detection, root cause analysis is much more domain dependent and therefore is a harder problem to address. As a result, significantly less work has been reported on this topic. A very comprehensive recent survey [97], however, lists and compares a range of models and techniques to support this task, including machine learning approaches.

Recent work has looked at ways to apply ML techniques to root cause analysis in the situation where the root cause is unknown and therefore undocumented, building on the achievements of the more recent rule based systems which incorporate an automatic rule extraction feature [47].

The general approach is to use an ML classification technique combine with an association rule extraction method (see Fig. 8 and also Table 11). The association rule extractor is used to scan network event logs and traces to detect possible associations between a given set of symptoms and potential causes, together with measures of the strength of the associations. The set of associations is then verified by the system operators and used to train the classifier so that incoming symptoms can be classified according to root cause.

Working with cellular network data, Yang *et al.* modified the FP-Growth association rule extraction technique, so that it could handle infrequent associations [61]. They then used *k* nearest neighbours to classify the symptoms. Nie *et al.* [96], studying Web based services, also took FP-Growth as their starting point, but in their case extended the algorithm to extract an initial causality graph from the historic data. In this case the classifier used was the Random Forest technique, which works by combining the results of multiple classification trees, each built from data randomly extracted from a training set. In this case the training set was the initial causality graph and the output was a refined version of the input graph. Kobayashi *et al.* used a causal inference algorithm to extract a causality graph from network data logs; some expert intervention is needed with this approach and the authors note that this technique will need to be combined with a classifier for use operationally [98].

The use of ML at the back end in association rule extraction approaches gives them an advantage in relation to the logic based approach described in [47] (see Section V above) in that it is not necessary to set up and maintain detailed diagnostic rules. It is still necessary to draw upon expert input to verify and rank the associations, but this requires much less effort than setting up a causal graph from scratch. Unlike [47], however, neither of the recent studies contain a separate model of the underlying service and network relationships. As a result there is as yet no capability for distinguishing between slowly and rapidly changing service-network relationships which could be used to assist incremental mining and training.

These methods open up the possibility of semi-automatically extracting symptom cause relationships from large volumes of fault data with efficient use of scarce expert input, in such a way that the result can be readily transferred to a rule based diagnostic system. As already discussed, however, in the future it may be necessary to move away from rule based systems for diagnosis to deep learning approaches working at the correlation level. This leads to two challenging research issues. Firstly, if causal level information does happen to be available, how feasible might it be for deep learning systems to use this to take short cuts to reduce training time and perhaps increase run time efficiency. Secondly, how can symptom cause linkages discovered during root cause analysis be exploited to provide an explanation capability consistent with the deep learning system behaviour.

F. COMPARATIVE QUANTITATIVE PERFORMANCE

Establishing comparative figures for fault management techniques is hampered by the fact that there are no standard metrics for reporting performance. In the cell outage detection case, however, a group of recent papers is available which report their results in terms of a Receiver Operating Curve (ROC) plot, ie true positive rate (TPR) against false positive rate (FPR) as a parameter is varied, typically the detection threshold. The area under the curve (AUC) is also a key measure; a random detector would score 0.5 and an ideal detector would score 1.0. Typical examples of the best results currently available are shown in Table 12.

TABLE 12. Cell outage detection techniques: Quantitive comparision.

Technique	Area Under Curve (AUC)	TPR for FPR = 10%	Reference
Local Outlier Factor Anomaly Detector (LOFAD)	0.85	c.65%	[33]
kth Nearest Neighbors	0.91	80%	[33]
Support Vector Machine	0.94-0.98	85-90%	[65]
Transductive Confidence Machine	Not given	100% claimed	[59]

G. SUMMARY

Table 13 provides a simple cross reference between the studies referenced in this paper and the stages of the fault management lifecycle. As can be seen, considerably more work has been carried out on detection than on the other stages. For this stage the best currently available approach would appear to be to use an anomaly detector (binary classifier) in supervised learning mode, such as a Support Vector Machine (a parametric approach) or *k* Nearest Neighbours (a non-parametric approach). Both techniques have the limitation, however, that a significant amount of hand coded pre-processing of the input data is required.

TABLE 13. Mapping of ML studies to fault management stages.

Fault Management Stage	References
Pre-Processing	[29], [33], [42], [53], [64], [65], [66], [83]
Detection	[27], [33], [42], [59], [64], [65], [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [79], [80], [81], [82]
Diagnosis 1: Action Determination	[27], [70], [82], [83], [84], [85], [86], [87], [60]
Compensation	[33], [92], [89], [62], [90], [91], [88]
Diagnosis 2: Root Cause Analysis	[96], [47], [61], [97]

The research base for the action determination (diagnosis 1) stage is not as solid as for detection; from the available works, however, the best of the traditional approaches would appear to be to carry out a cluster analysis on historic fault data in unsupervised learning mode, in order to separate out potential fault groups for review by an expert, followed by use of an ML classifier such as *k* Nearest Neighbours to allocate incoming symptom data to a fault group. A radically different approach which has recently been proposed is to learn compensation recommendations directly from historic logs of successful operator interventions.

For the compensation stage, studies have concentrated on the specific case of a cell outage, where the best performing approach is currently Actor-Critic RL, aiming to adjust transmission power levels and antenna tilt angles of adjacent cells based on measurements of radio signal quality levels in the outage cell area and those of its neighbours.

For the root cause analysis (diagnosis 2) stage, the state of the art would appear to be to use an automatic rule extraction technique working with historic fault data, typically using a modified version of the FP-growth algorithm. The rules are typically expressed in a causality graph, which then has to be reviewed by an expert before being presented to a back end ML classifier.

Overall, some progress is being made on reduction of the amount of expert input required, by the use of an initial unsupervised learning stage to reduce the volume of data to be reviewed. There is a general trend away from techniques based on a causal chain, towards approaches which operate at the correlation level. While this has led to improved results, at the same time it creates a problem in providing network operators with a supporting justification for the ML system's recommendations which is based on engineering principles and is consistent with whatever information can be provided by root cause analysis.

In the following section we report on the DARPA XAI initiative, which accepts that correlation-level approaches such as deep learning are likely to be here to stay and seeks to remedy their current explainability deficiencies.

VIII. DISCUSSION

In this section we put forward a vision for the attributes of future fault management systems for 5G networks and discuss how forthcoming changes in network architecture may impact fault management systems. We then assess the gaps between this vision and the current state of the art, sketch how these might be addressed and identify a number of current research issues with a view to closing these gaps. We discuss the difficulties with emerging correlation-level techniques, such as deep learning, in providing a justification for their recommendations, and describe the DARPA XAI initiative which is designed to address this issue.

A. ATTRIBUTES OF FUTURE FAULT MANAGEMENT SYSTEMS

The fault management system may in the future contain one or more ML elements, each supporting different stages of the fault management lifecycle, which need to be efficiently implemented within the context of the emerging 5G architecture. The system must be reliable and require minimal attention during operation. This implies that it will need to be designed to be resilient to its own faults. It should interwork with network equipment supplied by multiple vendors. It should also configure itself automatically, both on initial deployment and following a change to the network. The self healing function should work harmoniously with other network functional elements for self configuration and self optimisation. The fault management system will need to be compatible with emerging "zero-touch" (fully automated) network provisioning and change management systems. However the system is implemented, the network operations team must see it as trustworthy and easy to work with. The top level SON standard [11] specifies: "SON solutions shall provide an easy transition from operator controlled (open loop) to autonomous (closed loop) operation, as the network operator gains more trust in the reliability of the SON".

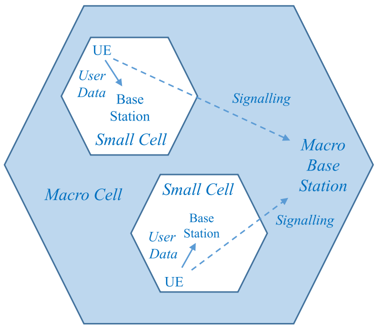

FIGURE 9. Split-cell RAN architecture.

At present, there is no consensus in the literature as to the key criteria to be used to evaluate the relative performance of proposed alternative ML techniques, although several recent papers use the Receiver Operating Curve (ROC) as the preferred measure of accuracy. To assist in arriving at a standard approach, in Table 14 we propose a set of metrics with suggested methods of measurement. These are divided into two categories: (a) output metrics, or desired performance attributes for the system, and (b) input metrics, which quantify the resources required to achieve the specified system performance. It can be anticipated that different target scores for each metric will apply at different stages of the lifecycle; for example a longer response time is likely to be acceptable for root cause analysis than for action determination.

B. IMPACT OF NETWORK ARCHITECTURE EVOLUTION

1) OPERATION IN THE PRESENCE OF FAULTS

The need to reduce the cell size in order to increase the user data rate has led to the 5G split-cell RAN architecture referred to in the introduction (see also Fig 9). The need for large numbers of densely deployed base stations is likely to result in a higher fault rate per unit area, especially in urban regions. At the same time, in order to contain operating costs, it will be essential to manage the restoration activity carefully in order to make optimal use of expensive site visits. This will demand a new approach to resilience in the RAN, so that it can continue to operate in the presence of multiple faults in any one local area, which can then be resolved during a single site visit.

Considerable work has already been done on resilience frameworks for communications networks in general, much of which will remain applicable to 5G networks [99], [100] and [101]. Beyond this, a key development for 5G is virtualisation, which we discuss in the next section.

These trends have a number of implications for fault management. Firstly, the design of the self healing function will need to be harmonised with that of the resilience features built

TABLE 14. **Proposed performance assessment metrics.**

Metric	Metric Type	Definition	Measurement Approach
Accuracy	Output	Area under Receiver Operating Curve (ROC)	Record and plot true positive v false positive rates
Bias	Output	Skew between diagnosis ROCs for different causes	Record and plot ROCs per cause
Response Time	Output	Time to complete detection, diagnosis, compensation or root cause analysis	Record differences in time stamps of events
Sensitivity	Output	Ability to tolerate noisy or missing data	Measure minimum signal to noise ratio plus maximum missing data percentage and data length to achieve stated accuracy
Flexibility	Output	Ability to adapt automatically to network changes or updates to network elements	Time and resources to relearn
Versatility	Output	Ability to transfer learning from one vendor's network equipment to another's	Time and resources to relearn
Computing Power	Input	Order of magnitude of computing power needed both locally (in the RAN) and centrally	Theoretical estimates eg $O(n^3)$ followed by practical measurements
Data Storage	Input	Memory and disk requirements locally and centrally	Practical measurements using standard IT tools
Expert Time	Input	Level of effort of specialist input needed (a) to set up the system and (b) to maintain it	Record time spent per grade of effort

into the network. Secondly, the presence of dormant faults will need to be taken into account by the self healing function. Thirdly, the self healing function will need to be able to deal with faults which impair its own effectiveness or that of the resilience features.

The upside, however, is that the existence of dormant faults, reported with some degree of labelling, should provide a much richer source of fault data than available up to now.

2) VIRTUALISATION IN THE RAN

Two important recent developments are Network Features Virtualisation (NFV), in which key network features are decoupled from the underlying hardware, and Software Defined Networking (SDN), which decouples the IP network control and user planes. These have had considerable impact on the core network architecture and are now beginning to spread to the RAN. The Cloud RAN (C-RAN) approach achieves functional virtualisation by partitioning the RAN into two parts, Base Band Units (BBUs) and Remote Radio Heads (RRHs) [6], [102]. This allows the radio frequency (RF) processing to be devolved to the RRHs, while the base band processing can be handled by a pool of virtualised BBU functions, allocated to centralised physical processors in such a way as to optimise processing speed and energy consumption. Routing of data between these functions, by the SDN-based IP network, is under the control of a similar pool of virtual processers. The balance between centralised virtual RAN processing and devolved physical processing is dependent on the provision of sufficient fixed link bandwidth, notably that of the so-called "fronthaul" link between the BBU pool and the RRHs. However, it may be assumed that at least part of the baseband processing will now be virtualised, which will impact fault management, especially the diagnosis task.

In any network, tracing of functional fault symptoms to physical causes depends on knowing the mapping of functions to physical processors. For the BBU pool within a C-RAN architecture this mapping is likely to be complex and will change dynamically. Traditional rule based approaches, such as those used to set up fault alarms, may become infeasible as the ruleset could become very large and would need to be dynamically updated to track the function mapping in the network. The problem becomes even more challenging if we consider the desirability of being able to map the service to the RAN and IP network functional elements providing it.

Hence there are a range of research opportunities to investigate how advanced ML techniques can be used to maintain an up to date mapping of services and network functions to physical hardware elements, perhaps by a process of continual online learning and discovery, and present this effectively to the network management team.

3) EDGE COMPUTING AND DATA STORAGE

Meanwhile the emergence of the Internet of Things (IoT), in particular vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) networks [103], is placing stringent latency requirements on the network, at the application layer as well as the network layer and below. The higher data rates available in 5G are also likely to raise user expectations on latency for delivery of large volumes of visual media content. These requirements can only be met by including physical computing and data storage elements at the edge of the network, working in cooperation with virtualised central functions in an architecture referred to as a "fog network" [104].

Further work is necessary to clarify the operational scenarios of such kinds of networks. One issue, for example, is whether each physical element will support just one application service (eg V2V or video caching) and if not what would be the best way to support services with different characteristics. A second key issue is how edge computing and storage elements would be managed - whether by the network service provider or by a separate organisation responsible for applications. Research opportunities may include how to extend the service mapping to include edge computing elements and the applications they support, together with how to harmonise the self healing function with the self optimising functions required to managing caching and local processing.

C. GAPS BETWEEN TARGET ATTRIBUTES AND CURRENT RESEARCH

Evolution of 5G will mean that with multiple radio chains in each base station driving separate antenna elements, cell output degradation may become a more likely scenario than a complete cell outage, opening up a gap for further investigation. Although research to date has covered most of the criteria listed in Table 14, there is work to be done on to determine the appropriate balance between centralisation and distribution of the system architecture in order to minimise resource requirements. There are also currently gaps in the areas of system flexibility and versatility. Beyond this, there has been relatively little emphasis in the literature on system trustworthiness and how the ML system can work most effectively with the network operations team. We discuss these gaps in more detail below.

1) SYSTEM SENSITIVITY

Up to now, research has focused on the cell outage problem. However, with the introduction of 3D transmission and more advanced forms of MIMO, base stations will now have large numbers of antenna elements, typically configured as a planar array. Each element has to be driven by a radio chain; although a hybrid architecture can be used to share RF chains between antenna elements, it will still be necessary to have multiple radio chains in future base stations [105]–[108] and [109]. In this new scenario, it is possible that base station faults will result in degradations of the cell radio output more often than complete outages, placing greater demands on detection sensitivity.

Fortunately, the widespread availability of MDT radio coverage reporting data provides an opportunity to increase system sensitivity by recording the radio coverage profile for each cell during normal operation. This is already proposed for energy management, and will enable fault management systems to reduce the impact of unwanted variables such as pathloss by comparing reported KPI values with normal values for a given location [110]. In some situations, such as with femtocells and densely deployed small cells, the number of users can be very small, so that radio reports may be sparse or not available for some areas. This can be addressed with ML techniques such as collaborative filtering [73]. This technique exploits the correlation between signal strength reports from users and those from adjacent base stations to provide estimated radio reports to fill in the gaps. It may also be possible to take advantage of the emergence of the IoT to exploit the presence of such devices, many of which are likely to have a fixed location, to improve the accuracy of the stored coverage profile [111].

2) SYSTEM FLEXIBILITY

Current difficulties with network management systems deployed in live networks include excessive effort to configure and test the system, and fragility of the system in the face of network upgrades, reconfigurations and other changes. A key factor currently limiting system flexibility is the number of lines and complexity of the hand crafted code at the front end which has to be modified and retested in order to accommodate external changes.

Recent advances in the application of deep NNs (typically convolutional NNs and autoencoders) in other engineering areas, which have input processing requirements as demanding as mobile networks, suggest that it should be possible to reduce the need for domain and problem-specific code in stage 1 of the pre-processing, and potentially incorporate much of stage 2 into the deep NN so that the functions of data fusion, feature engineering and dimensionality reduction can be carried out automatically [112]–[118].

3) SYSTEM VERSATILITY

An issue closely related to flexibility is interoperability between the network management system and network elements provided by different vendors. Achievement of this currently requires significant standardisation and testing effort to align the interfaces of the different equipment instances with that of the network management system and may be hindered by commercial issues [119]. To improve on this, once an ML based fault management system had been trained on one vendor's equipment, it would ideally be possible to retrain it, without code changes, to interface to similar equipment from another vendor.

In the ML literature this is referred to as the transfer learning issue, for which a useful survey of approaches is given in by Pan and Yang [120]. The authors of this paper define transfer learning as the extraction of knowledge from a *source task* in order to apply it to a *target task*. In the interoperability case, the source and target domains are very similar, hence there would appear to be an opportunity for research into the application of transfer learning approaches to maximise system versatility.

4) SYSTEM TRUSTWORTHINESS

The papers referenced above typically focus on solutions to current technical issues. If new techniques are to be successfully introduced into a live network, however, it will be critically important for the system to earn the confidence of the network management team. Apart from meeting basic requirements such as data quality and system reliability, this

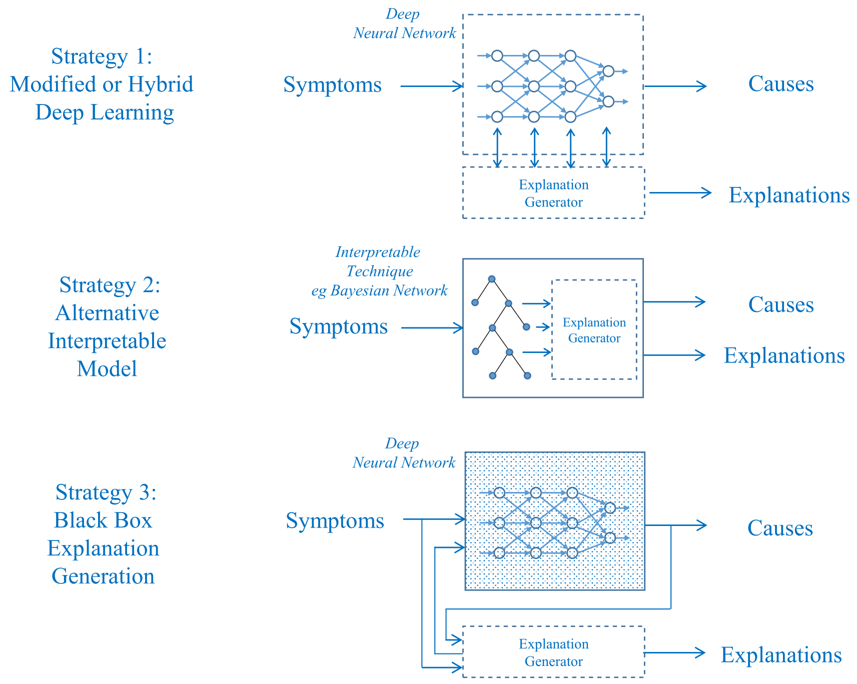

FIGURE 10. Explainability implementation techniques.

will mean that the fault management system will need to be aligned operationally with the network management processes, especially the procedures for approval of proposed changes to the network. These typically require a senior network manager to evaluate and approve recommendations for change.

This in turn will mean that the system will need to provide some kind of explanation in support of fault diagnoses and proposed fault compensation actions. This is particularly challenging with recent deep learning approaches, which draw their power from operating at the correlation level rather than by building a causal model from which an explanation could be derived.

This issue has been recognised by the research community under the heading "Explainable AI", and is beginning to gain considerable traction following the launch in 2016 of the XAI programme by the US Defense Advanced Research Projects Agency (DARPA) [132].

The DARPA launch document identifies three related research challenges:

1) How to produce more explainable models
2) How to design the explanation interface
3) How to understand the psychological requirements for effective explanations

Some work has already been done, based on cognitive research, to identify the general attributes of an effective machine learning explanation.

Ribeiro *et al.* and Miller *et al.* provide useful pointers to what should be included in an effective explanation, based on research on cognitive psychology [131] and [133]. Miller *et al* highlight that when people evaluate the quality of an explanation, naturally they expect that causes cited in an explanation should be correct but they also highly rate usefulness and relevance [133]. Hence simple explanations based on selected key causes are preferred to complex explanations provided they are sufficiently accurate for the needs of the task in hand.

The DARPA document identifies the following illustrative strategies for producing more explainable models (see Fig. 10):

1) Develop modified or hybrid deep learning techniques that learn more explainable features, explainable representations, or explanation generation facilities
2) Develop alternative machine learning techniques that learn more structured, interpretable or causal models
3) Develop techniques that would experiment with any given machine learning model - as a black box - to infer an approximate, explainable model

As can be seen from Fig. 10, the third strategy is an active one, in which the inputs to the black box can be varied to calculate the sensitivity of the output to changes to each of the inputs. There is the option that the first strategy could also be active, if it were found useful to allow the explanation generator to influence the deep learning subsystem's internal states.

Tables 15-17 give examples of approaches to each of these implementation strategies which could be applicable to cell fault management. A recent, more general survey of XAI approaches is given in [134]. A useful discussion on how to select the most appropriate representation in the case of the first strategy is given in [135].

TABLE 15. Modified or hybrid deep learning strategies.

Approach	Summary	References
Explainable PCA	Use expert input to guide the selection of the dimensions of the reduced dimensionality space to align with interpretable variables	[121]
Forward Path Weightings	Generate explanations for a given input directly from the weightings used in the forward path. Classifier has to be additive (no cross-coupling between the features in the classification path).	[122]
Forward Path Approximation	Use a Taylor series expansion to form a linear approximation to the forward path for a given input and output, can calculate the input-output gradient to select the inputs to which the output most sensitive.	[123]
Intermediate Loss Function	Use one or more intermediate loss functions to encourage interpretability eg for image processing allocate single image object to each late stage filter in a convolutional neural network.	[124]
Reverse Relevance Propagation	For a particular prediction and knowing the structure of the DNN, work backwards from the output to calculate the relevance of each feature of the input data to the particular prediction by the DNN	[125], [126], [127]
Reverse Path Deconvolution	Working with a CNN, a reverse path Deconvolutional NN (DNN), derived from the CNN, is used to reconstruct the key input pixels supporting the classification.	[128]

TABLE 16. Alternative interpretable model strategies.

Approach	Summary	References
Predicate Logic	Data set mining to identify frequently occurring event-precursor rules with small number of precursors. Compare rule set with accepted expert judgement.	[129]
Interpretable probabilistic model	Analyse reference examples to build an inherently explainable, structured probabilistic model.	[130]
Interpretable causal graph	Learn a causal graph from input examples and refine using expert knowledge.	[96]

Of the three strategies, the third one has the advantage that it can work with any deep learning technique. The disadvantage, however, is that it is limited to the information that can be discovered by varying the input around a series of set points.

The second strategy requires the minimum additional work to generate an explanation but comes with the limitations of earlier model-based ML approaches, in particular the need for expert input.

The first strategy requires the explanation generator to be tailored to the structure of the deep learning technique,

TABLE 17. Black box explanation generation strategies.

Approach	Summary	References
Input Relevance Analysis	For a given set of inputs, vary one input at a time to determine sensitivity and relevance of output to this input	[125]
Locally Valid Approximation	Build set of interpretable models alongside the deep NN, each of which is valid only for small variations to a given input.	[131]

but the ability to access internal data extends the range of explainability options, given that it can be used in conjunction with the third strategy if necessary.

Whichever technique or combination of techniques is chosen, there are a number of challenging open research issues on how to enable the fault management system take into account the operational context as well as the technical aspects of a given fault when providing a justification for proposed actions. These include the prioritisation of the fault in relation to current and expected traffic as well as risk analysis of proposed actions in relation to current and anticipated network conditions.

IX. FUTURE RESEARCH DIRECTIONS

The key hot topics at present are:

1) how to exploit the benefits of deep learning to achieve improved performance with less preprocessing code
2) developing the fundamental techniques to enable deep learning systems to explain their recommendations

Beyond this, the future direction of research in this field is likely to be shaped by two key factors:

1) the changing architecture and characteristics of the mobile network
2) the challenges arising from the need to implement ML approaches in an operational context

Key research topics arising from the evolving network architecture include the following:

1) keeping ahead of growth in network complexity especially in the area of diagnosis
2) handling the impact of virtualisation where the relationships between the connectivity service, the functional entities providing it and the underlying physical network elements become complex and dynamic
3) how to address similar issues arising from the emergence of edge computing and data storage elements, operating at the application level, which may be under the control of a different service provider
4) how to maintain harmonious interworking between self organising functional areas as these become more complex in response to network evolution

The research challenges arising from the need to implement ML solutions in an operational context include:

1) how to build in an awareness of contextual issues such as fault prioritisation based on network traffic and risk

assessment of proposed actions, balancing risk in relation to priority
2) investigate how exploratory action by active ML systems such as RL and deep RL can be constrained to levels which are acceptable in a live network without unduly impacting their performance
3) generalising system learning, especially for active ML systems, so that training on a representative simulated environment can applied to specific network locations with minimum retraining
4) how best to exploit the results of root cause analysis to optimise the performance of ML based self healing systems and support the development of accurate explanations

A key enabler for recent progress in the development of deep NNs appears to have been the existence of very large, representative datasets such as the MNIST handwritten digit images database [136], and other industry specific databases, which are publicly available and can be used to train and evaluate candidate ML techniques. The application to cell fault management would be to create an industry wide 5G fault database which could then be used to stimulate the development of improved ML approaches.

X. CONCLUSION

We have seen that a wide range of studies have already shown that ML techniques can be successfully applied to support root cause analysis and provide fully automated self healing functions in the form of detection, diagnosis and compensation of faults. A significant body of research work on detection of cell outages, together with compensation strategies to restore service, has led to an emerging consensus on which are the most appropriate ML techniques to use for these specific tasks.

At the present time, however, the state of the art is encountering two key limitations. From a technical perspective, the need for a hard coded pre-processing stage in support of current ML techniques is constraining their flexibility to accommodate changes in the network. From an operational standpoint, the focus on full automation has meant that relatively little attention has been given to situations where it will continue to be necessary for a human operator to be included in the decision making loop.

Research in other engineering domains with similar fault management issues suggests that recent ML techniques based on deep learning can be applied in cellular networks to resolve the flexibility issue, if sufficient fault data can be provided. Ideally such data would be made available as an open source database to stimulate research and development.

Deep learning techniques, however, typically rely on correlation based optimisers rather than explicit causal models, which makes it hard for people to audit their conclusions. This issue is being addressed by the DARPA XAI research initiative, which has identified a number of potential strategies by which deep learning systems could be given the ability to justify their recommendations, and is supporting intensive research work to demonstrate the feasibility of one or more of these approaches.

Hence we can expect to see rapid advances in the application of machine learning techniques to cell fault management in the very near future, with particular emphasis on enhanced deep learning approaches able to interact productively and build trust in cases where it remains necessary for a human to be included in the loop.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of members of the University of Surrey 5GIC (http://www.surrey.ac.uk/5gic) for this work.

REFERENCES

[1] H. Ishii, Y. Kishiyama, and H. Takahashi, "A novel architecture for LTE-B: C-plane/U-plane split and phantom cell concept," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2012, pp. 624–630.
[2] T. Nakamura, S. Nagata, A. Benjebbour, Y. Kishiyama, T. Hai, S. Xiaodong, Y. Ning, and L. Nan, "Trends in small cell enhancements in LTE advanced," *IEEE Commun. Mag.*, vol. 51, no. 2, pp. 98–105, Feb. 2013.
[3] A. Mohamed, O. Onireti, M. A. Imran, A. Imran, and R. Tafazolli, "Control-data separation architecture for cellular radio access networks: A survey and outlook," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 446–465, 1st Quart., 2015.
[4] J. Lee, Y. Kim, H. Lee, B. L. Ng, D. Mazzarese, J. Liu, W. Xiao, and Y. Zhou, "Coordinated multipoint transmission and reception in LTE-advanced systems," *IEEE Commun. Mag.*, vol. 50, no. 11, pp. 44–50, Nov. 2012.
[5] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.
[6] M. Peng, Y. Sun, X. Li, Z. Mao, and C. Wang, "Recent advances in cloud radio access networks: System architectures, key techniques, and open issues," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2282–2308, Aug. 2016.
[7] P. K. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, "Design considerations for a 5G network architecture," *IEEE Commun. Mag.*, vol. 52, no. 11, pp. 65–75, Nov. 2014.
[8] E. Hossain and M. Hasan, "5G cellular: Key enabling technologies and research challenges," *IEEE Instrum. Meas. Mag.*, vol. 18, no. 3, pp. 11–21, Jun. 2015.
[9] M. Peng, Y. Li, Z. Zhao, and C. Wang, "System architecture and key technologies for 5G heterogeneous cloud radio access networks," *IEEE Netw.*, vol. 29, no. 2, pp. 6–14, Mar./Apr. 2015.
[10] L. C. Schmelz, J. Van Den Berg, R. Litjens, K. Zetterberg, M. Amirijoo, K. Spaey, I. Balan, N. Scully, and S. Stefanski, "Self-organisation in wireless networks: Use cases and their interrelations," in *Proc. Wireless World Res. Forum Meeting*, vol. 22, 2009, pp. 1–5.
[11] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Self-Organizing Networks (SON); Concepts and Requirements (Release 13)*, document 3GPP TS 32.500, 2016-01, v13.0.0, 2016.
[12] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Self-Configuration of Network Elements; Concepts and Requirements (Release 13)*, document 3GPP TS 32.501, 2016-01, v13.0.0, 2016.
[13] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Self-Organizing Networks (SON); Self-Healing Concepts and Requirements (Release 13)*, document 3GPP TS 32.541, 2016-01, v13.0.0, 2016.
[14] M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1996–2018, 4th Quart., 2014.
[15] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1136–1159, Jul. 2013.

[16] X. X. Wang Li and V. C. M. Leung, "Artificial intelligence-based techniques for emerging heterogeneous network: State of the arts, opportunities, and challenges," *IEEE Access*, vol. 3, pp. 1379–1391, 2015.

[17] X. Zhou, M. Sun, G. Y. Li, and B.-H. F. Juang, "Intelligent wireless communications enabled by cognitive radio and machine learning," *China Commun.*, vol. 15, no. 12, pp. 16–48, Dec. 2018.

[18] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," Oct. 2017, *arXiv:1710.02913*. [Online]. Available: https://arxiv.org/abs/1710.02913

[19] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2224–2287, 3rd Quart., 2019.

[20] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," *IEEE Commun. Surveys Tuts.*, to be published.

[21] O. G. Aliu, A. Imran, M. A. Imran, and B. Evans, "A survey of self organisation in future cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 336–361, Feb. 2013.

[22] P. V. Klaine, O. Onireti, R. D. Souza, and M. A. Imran, *The Role and Applications of Machine Learning in Future Self-Organizing Cellular Networks*. Hershey, PA, USA: IGI Global, 2018.

[23] A. Asghar, H. Farooq, and A. Imran, "Self-healing in emerging cellular networks: Review, challenges, and research directions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 1682–1709, 3rd Quart., 2018.

[24] R. Barco, P. Lazaro, and P. Munoz, "A unified framework for self-healing in wireless networks," *IEEE Commun. Mag.*, vol. 50, no. 12, pp. 134–142, Dec. 2012.

[25] A. Imran, A. Zoha, and A. Abu-Dayya, "Challenges in 5G: How to empower SON with big data for enabling 5G," *IEEE Netw.*, vol. 28, no. 6, pp. 27–33, Nov. 2014.

[26] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Key Performance Indicators (KPI) for Evolved Universal Terrestrial Radio Access Network (E-UTRAN): Definitions (Release 14)*, document 3GPP TS 32.450, 2017-04, v14.0.0, 2017.

[27] P. Szilagyi and S. Novaczki, "An automatic detection and diagnosis framework for mobile communication systems," *IEEE Trans. Netw. Service Manage.*, vol. 9, no. 2, pp. 184–197, Jun. 2012.

[28] Q. Liao, M. Wiczanowski, and S. Stańczak, "Toward cell outage detection with composite hypothesis testing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 4883–4887.

[29] E. J. Khatib, R. Barco, I. Serrano, and P. Muñoz, "LTE performance data reduction for knowledge acquisition," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2014, pp. 270–274.

[30] I. de-la-Bandera, R. Barco, P. Muñoz, and I. Serrano, "Cell outage detection based on handover statistics," *IEEE Commun. Lett.*, vol. 19, no. 7, pp. 1189–1192, Jul. 2015.

[31] A. Gómez-Andrades, R. Barco, and I. Serrano, "A method of assessment of LTE coverage holes," *EURASIP J. Wireless Commun. Netw.*, vol. 2016, no. 1, p. 236, Oct. 2016. doi: 10.1186/s13638-016-0733-y.

[32] P. Muñoz, I. de la Bandera, E. J. Khatib, A. Gómez-Andrades, I. Serrano, and R. Barco, "Root cause analysis based on temporal analysis of metrics toward self-organizing 5G networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2811–2824, Mar. 2017.

[33] O. Onireti, A. Zoha, J. Moysen, A. Imran, L. Giupponi, M. Imran, and A. Abu-Dayya, "A cell outage management framework for dense heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2097–2113, Apr. 2016.

[34] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Subscriber and Equipment Trace; Trace Concepts and Requirements (Release 14)*, document 3GPP TS 32.421, 2017-04, v14.0.0, 2017.

[35] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Subscriber and Equipment Trace; Trace Control and Configuration Management (Release 15)*, document 3GPP TS 32.422, 2017-09, v15.0.0, 2017.

[36] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Telecommunication Management; Subscriber and Equipment Trace; Trace Data Definition and Management (Release 14)*, document 3GPP TS 32.423, 2017-04, v14.0.0, 2017.

[37] W. A. Hapsari, A. Umesh, M. Iwamura, M. Tomala, B. Gyula, and B. Sebire, "Minimization of drive tests solution in 3GPP," *IEEE Commun. Mag.*, vol. 50, no. 6, pp. 28–36, Jun. 2012.

[38] J. Johansson, W. A. Hapsari, S. Kelley, and G. Bodog, "Minimization of drive tests in 3GPP release 11," *IEEE Commun. Mag.*, vol. 50, no. 11, pp. 36–43, Nov. 2012.

[39] *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Universal Terrestrial Radio Access (UTRA) and Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Measurement Collection for Minimization of Drive Tests (MDT); Overall Description; Stage 2 (Release 13)*, document 3GPP TS 37.320, 2016-03, v13.1.0, 2016.

[40] R. Di Taranto, S. Muppirisetty, R. Raulefs, D. Slock, T. Svensson, and H. Wymeersch, "Location-aware communications for 5G networks: How location information can improve scalability, latency, and robustness of 5G," *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 102–112, Nov. 2014.

[41] H. Abou-zeid, H. S. Hassanein, and R. Atawia, "Towards mobility-aware predictive radio access: Modeling; simulation; And evaluation in LTE networks," in *Proc. 17th ACM Int. Conf. Modeling, Anal. Simulation Wireless Mobile Syst.*, 2014, pp. 109–116.

[42] C. M. Mueller, M. Kaschub, C. Blankenhorn, and S. Wanke, "A cell outage detection algorithm using neighbor cell list reports," in *Proc. IWSOS LNCS*, vol. 5343. Berlin, Germany: Springer-Verlag, 2008, pp. 218–219.

[43] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA, USA: Morgan Kaufmann, 2014.

[44] R. N. Cronk, P. H. Callahan, and L. Bernstein, "Rule-based expert systems for network management and operations: An introduction," *IEEE Netw.*, vol. 2, no. 5, pp. 7–21, Sep. 1988.

[45] G. Jakobson and M. Weissman, "Alarm correlation," *IEEE Netw.*, vol. 7, no. 6, pp. 52–59, Nov. 1993.

[46] P. Frohlich, W. Nejdl, K. Jobmann, and H. Wietgrefe, "Model-based alarm correlation in cellular phone networks," in *Proc. 5th Int. Symp. Modeling, Anal., Simulation Comput. Telecommun. Syst.*, Jan. 1997, pp. 197–204.

[47] H. Yan, L. Breslau, Z. Ge, D. Massey, D. Pei, and J. Yates, "G-RCA: A generic root cause analysis platform for service quality management in large IP networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 6, pp. 1734–1747, Dec. 2012.

[48] B.-K. Yeo and Y. Lu, "Array failure correction with a genetic algorithm," *IEEE Trans. Antennas Propag.*, vol. 47, no. 5, pp. 823–828, May 1999.

[49] B.-K. Yeo and Y. Lu, "Fast array failure correction using improved particle swarm optimization," in *Proc. Asia Pacific Microw. Conf.*, 2009, pp. 1537–1540.

[50] W. Li, P. Yu, M. Yin, and L. Meng, "A distributed cell outage compensation mechanism based on RS power adjustment in LTE networks," *China Commun.*, vol. 11, no. 13, pp. 40–47, 2014.

[51] L. Kayili and E. Sousa, "Cell outage compensation for irregular cellular networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2014, pp. 1850–1855.

[52] M. Z. Bell, "Why expert systems fail," *J. Oper. Res. Soc.*, vol. 36, no. 7, pp. 613–619, 1985.

[53] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.

[54] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer, 2011.

[55] R. S. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, vol. 1. Cambridge, MA, USA: MIT Press, 1998, no. 1.

[56] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[57] C. Xiao, D. Yang, Z. Chen, and G. Tan, "3-D ble indoor localization based on denoising autoencoder," *IEEE Access*, vol. 5, pp. 12751–12760, 2017.

[58] W. Feng, Y. Teng, Y. Man, and M. Song, "Cell outage detection based on improved BP neural network in LTE system," in *Proc. 11th Int. Conf. Wireless Commun., Netw. Mobile Comput. (WiCOM)*, 2015, pp. 1–5.

[59] J. Wang, N. Q. Phan, Z. Pan, N. Liu, X. You, and T. Deng, "An improved TCM-based approach for cell outage detection for self-healing in LTE hetnets," in *Proc. IEEE 83rd Veh. Technol. Conf. (VTC Spring)*, May 2016, pp. 1–5.

[60] A. Gómez-Andrades, P. Muñoz, I. Serrano, and R. Barco, "Automatic root cause analysis for LTE networks based on unsupervised techniques," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2369–2386, Apr. 2016.

[61] K. Yang, R. Liu, Y. Sun, J. Yang, and X. Chen, "Deep network analyzer (DNA): A big data analytics platform for cellular networks," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 2019–2027, Dec. 2017.

[62] M. N. ul Islam and A. Mitschele-Thiel, "Reinforcement learning strategies for self-organized coverage and capacity optimization," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 2818–2823.

[63] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[64] S. Nováczki, "An improved anomaly detection and diagnosis framework for mobile network operators," in *Proc. 9th Int. Conf. Design Reliable Commun. Netw. (DRCN)*, Mar. 2013, pp. 234–241.

[65] A. Zoha, A. Saeed, A. Imran, M. A. Imran, and A. Abu-Dayya, "Data-driven analytics for automated cell outage detection in self-organizing networks," in *Proc. 11th Int. Conf. Design Reliable Commun. Netw. (DRCN)*, Mar. 2015, pp. 203–210.

[66] J. Turkka, F. Chernogorov, K. Brigatti, T. Ristaniemi, and J. Lempiäinen, "An approach for network outage detection from drive-testing databases," *J. Comput. Netw. Commun.*, vol. 2012, Sep. 2012, Art. no. 163184.

[67] A. Bouillard, A. Junier, and B. Ronot, "Hidden anomaly detection in telecommunication networks," in *Proc. 8th Int. Conf. Netw. Service Manage.*, Oct. 2012, pp. 82–90.

[68] G. F. Ciocarlie, U. Lindqvist, S. Nováczki, and H. Sanneck, "Detecting anomalies in cellular networks using an ensemble method," in *Proc. 9th Int. Conf. Netw. Service Manage. (CNSM)*, Oct. 2013, pp. 171–174.

[69] Y. Zhang, N. Liu, Z. Pan, T. Deng, and X. You, "A fault detection model for mobile communication systems based on linear prediction," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Oct. 2014, pp. 703–708.

[70] S. Fortes, R. Barco, and A. Aguilar-García, and P. Muñoz, "Contextualized indicators for online failure diagnosis in cellular networks," *Comput. Netw.*, vol. 82, pp. 96–113, May 2015.

[71] S. Nováczki and P. Szilágyi, "Radio channel degradation detection and diagnosis based on statistical analysis," in *Proc. IEEE 73rd Veh. Technol. Conf. (VTC Spring)*, May 2011, pp. 1–2.

[72] A. Coluccia, A. D'Alconzo, and F. Ricciato, "Distribution-based anomaly detection via generalized likelihood ratio test: A general maximum entropy approach," *Comput. Netw.*, vol. 57, pp. 3446–3462, Dec. 2013.

[73] W. Wang, Q. Liao, and Q. Zhang, "COD: A cooperative cell outage detection architecture for self-organizing femtocell networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 11, pp. 6007–6014, Nov. 2014.

[74] S. Fortes, A. A. Garcia, J. A. Fernandez-Luque, A. Garrido, and R. Barco, "Context-aware self-healing: User equipment as the main source of information for small-cell indoor networks," *IEEE Veh. Technol. Mag.*, vol. 11, no. 1, pp. 76–85, Mar. 2016.

[75] B. Hussain, Q. Du, and P. Ren, "Semi-supervised learning based big data-driven anomaly detection in mobile wireless networks," *China Commun.*, vol. 15, no. 4, pp. 41–57, Apr. 2018.

[76] W. Xue, M. Peng, Y. Ma, and H. Zhang, "Classification-based approach for cell outage detection in self-healing heterogeneous networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2014, pp. 2822–2826.

[77] W. Xue, H. Zhang, Y. Li, D. Liang, and M. Peng, "Cell outage detection and compensation in two-tier heterogeneous networks," *Int. J. Antennas Propag.*, vol. 2014, May 2014, Art. no. 624858.

[78] F. Chernogorov, S. Chernov, K. Brigatti, and T. Ristaniemi, "Sequence-based detection of sleeping cell failures in mobile networks," *Wireless Netw.*, vol. 22, no. 6, pp. 2029–2048, Oct. 2015. doi: 10.1007/s11276-015-1087-9.

[79] F. Chernogorov, I. Repo, V. Räisänen, T. Nihtilä, and J. Kurjenniemi, "Cognitive self-healing system for future mobile networks," in *Proc. Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Aug. 2015, pp. 628–633.

[80] P. Yu, F. Zhou, T. Zhang, W. Li, L. Feg, and X. Qiu, "Self-organized cell outage detection architecture and approach for 5G H-CRAN," *Wireless Commun. Mobile Comput.*, vol. 2018, May 2018, Art. no. 6201386.

[81] Y. Ma, M. Peng, W. Xue, and X. Ji, "A dynamic affinity propagation clustering algorithm for cell outage detection in self-healing networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2013, pp. 2266–2270.

[82] G. F. Ciocarlie, C. Connolly, C.-C. Cheng, U. Lindqvist, S. Nováczki, H. Sanneck, and M. Naseer-ul-Islam, "Anomaly detection and diagnosis for automatic radio network verification," in *Proc. Int. Conf. Mobile Netw. Manage.* Cham, Switzerland: Springer, 2014, pp. 163–176.

[83] R. M. Khanafer, B. Solana, J. Triola, R. Barco, L. Moltsen, Z. Altman, and P. Lazaro, "Automated diagnosis for UMTS networks using Bayesian network approach," *IEEE Trans. Veh. Technol.*, vol. 57, no. 4, pp. 2451–2461, Jul. 2008.

[84] R. Barco, V. Wille, and L. Díez, "System for automated diagnosis in cellular networks based on performance indicators," *Eur. Trans. Telecommun.*, vol. 16, no. 5, pp. 399–409, 2005.

[85] R. Barco, P. Lázaro, L. Díez, and V. Wille, "Continuous versus discrete model in autodiagnosis systems for wireless networks," *IEEE Trans. Mobile Comput.*, vol. 7, no. 6, pp. 673–681, Jun. 2008.

[86] R. Barco, V. Wille, L. Díez, and M. Toril, "Learning of model parameters for fault diagnosis in wireless networks," *Wireless Netw.*, vol. 16, no. 1, pp. 255–271, Jan. 2010. doi: 10.1007/s11276-008-0128-z.

[87] A. Gómez-Andrades, R. Barco, I. Serrano, P. Delgado, P. Caro-Oliver, and P. Munoz, "Automatic root cause analysis based on traces for LTE self-organizing networks," *IEEE Wireless Commun.*, vol. 23, no. 3, pp. 20–28, Jun. 2016.

[88] S. Deb, Z. Ge, S. Isukapalli, S. Puthenpura, S. Venkataraman, H. Yan, and J. Yates, "Aesop: Automatic policy learning for predicting and mitigating network service impairments," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1783–1792.

[89] A. Saeed, O. G. Aliu, and M. A. Imran, "Controlling self healing cellular networks using fuzzy logic," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 3080–3084.

[90] J. Li, J. Zeng, X. Su, W. Luo, and J. Wang, "Self-optimization of coverage and capacity in LTE networks based on central control and decentralized fuzzy Q-learning," *Int. J. Distrib. Sensor Netw.*, vol. 8, no. 8, 2012, Art. no. 878595.

[91] J. Moysen and L. Giupponi, "A reinforcement learning based solution for self-healing in LTE networks," in *Proc. IEEE 80th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2014, pp. 1–6.

[92] M. Amirijoo, L. Jorguseski, R. Litjens, and R. Nascimento, "Effectiveness of cell outage compensation in LTE networks," in *Proc. IEEE Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2011, pp. 642–647.

[93] T. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.

[94] Y. Sun, M. Peng, and S. Mao, "Deep reinforcement learning-based mode selection and resource management for green fog radio access networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1960–1971, Apr. 2019.

[95] F. B. Mismar and B. L. Evans, "Deep Q-learning for self-organizing networks fault management and radio performance improvement," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, 2018, pp. 1457–1461.

[96] X. Nie, Y. Zhao, K. Sui, D. Pei, Y. Chen, and X. Qu, "Mining causality graph for automatic web-based service diagnosis," in *Proc. IEEE 35th Int. Perform. Comput. Commun. Conf. (IPCCC)*, Dec. 2016, pp. 1–8.

[97] M. Solé, V. Muntés-Mulero, A. I. Rana, and G. Estrada, "Survey on models and techniques for root-cause analysis," Jan. 2017, *arXiv:1701.08546*. [Online]. Available: https://arxiv.org/abs/1701.08546

[98] S. Kobayashi, K. Otomo, K. Fukuda, and H. Esaki, "Mining causality of network events in log data," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 1, pp. 53–67, Mar. 2018.

[99] P. Cholda, A. Mykkeltveit, B. E. Helvik, O. J. Wittner, and A. Jajszczyk, "A survey of resilience differentiation frameworks in communication networks," *IEEE Commun. Surveys Tuts.*, vol. 9, no. 4, pp. 32–55, 4th Quart., 2007.

[100] J. P. G. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Comput. Netw.*, vol. 54, no. 8, pp. 1245–1265, 2010.

[101] P. Smith, D. Hutchison, J. P. G. Sterbenz, M. Schöller, A. Fessi, M. Karaliopoulos, C. Lac, and B. Plattner, "Network resilience: A systematic approach," *IEEE Commun. Mag.*, vol. 49, no. 7, pp. 88–97, Jul. 2011.

[102] A. Checko, H. L. Christiansen, Y. Yan, L. Scolari, G. Kardaras, M. S. Berger, and L. Dittmann, "Cloud RAN for mobile networks—A technology overview," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 405–426, 1st Quart., 2015.

[103] K. Zheng, Q. Zheng, P. Chatzimisios, W. Xiang, and Y. Zhou, "Heterogeneous vehicular networking: A survey on architecture, challenges, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2377–2396, 4th Quart., 2015.

[104] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: Issues and challenges," *IEEE Netw.*, vol. 30, no. 4, pp. 46–53, Jul. 2016.

[105] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5G," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 74–80, Feb. 2014.

[106] P. Demestichas, A. Georgakopoulos, D. Karvounas, K. Tsagkaris, V. Stavroulaki, J. Lu, C. Xiong, and J. Yao, "5G on the horizon: Key challenges for the radio-access network," *IEEE Veh. Technol. Mag.*, vol. 8, no. 3, pp. 47–53, Sep. 2013.

[107] W. Roh, J.-Y. Seol, J. Park, B. Lee, J. Lee, Y. Kim, J. Cho, K. Cheun, and F. Aryanfar, "Millimeter-wave beamforming as an enabling technology for 5G cellular communications: Theoretical feasibility and prototype results," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 106–113, Feb. 2014.

[108] Y. Kim, H. Ji, J. Lee, Y.-H. Nam, B. L. Ng, I. Tzanidis, Y. Li, and J. Zhang, "Full dimension MIMO (FD-MIMO): The next evolution of MIMO in LTE systems," *IEEE Wireless Commun.*, vol. 21, no. 2, pp. 26–33, Apr. 2014.

[109] A. Liu and V. Lau, "Phase only RF precoding for massive MIMO systems with limited RF chains," *IEEE Trans. Signal Process.*, vol. 62, no. 17, pp. 4505–4515, Sep. 2014.

[110] E. Ternon, P. Agyapong, L. Hu, and A. Dekorsy, "Energy savings in heterogeneous networks with clustered small cell deployments," in *Proc. 11th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2014, pp. 126–130.

[111] M. R. Palattella, M. Dohler, A. Grieco, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[112] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring: A survey," Dec. 2016, *arXiv:1612.07640*. [Online]. Available: https://arxiv.org/abs/1612.07640

[113] F. Jia, Y. Lei, J. Lin, X. Zhou, and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data," *Mech. Syst. Signal Process.*, vol. 72, pp. 303–315, May 2016.

[114] V. T. Tran, F. AlThobiani, and A. Ball, "An approach to fault diagnosis of reciprocating compressor valves using Teager–Kaiser energy operator and deep belief networks," *Expert Syst. Appl.*, vol. 41, no. 9, pp. 4113–4122, 2014.

[115] H. Shao, H. Jiang, H. Zhao, and F. Wang, "A novel deep autoencoder feature learning method for rotating machinery fault diagnosis," *Mech. Syst. Signal Process.*, vol. 95, pp. 187–204, Oct. 2017.

[116] W. Sun, R. Zhao, R. Yan, S. Shao, and X. Chen, "Convolutional discriminative feature learning for induction motor fault diagnosis," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 1350–1359, Jun. 2017.

[117] L. Jing, T. Wang, M. Zhao, and P. Wang, "An adaptive multi-sensor data fusion method based on deep convolutional neural networks for fault diagnosis of planetary gearbox," *Sensors*, vol. 17, no. 2, p. 414, 2017.

[118] R. Zhao, R. Yan, J. Wang, and K. Mao, "Learning to monitor machine health with convolutional Bi-directional LSTM networks," *Sensors*, vol. 17, no. 2, p. 273, 2017.

[119] A. Gupta, "Network management: Current trends and future perspectives," *J. Netw. Syst. Manage.*, vol. 14, no. 4, pp. 483–491, 2006.

[120] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[121] C. Brinton, "A framework for explanation of machine learning decisions," in *Proc. IJCAI Workshop Explainable AI (XAI)*, 2017, p. 14.

[122] Q. Yu, J. Liu, H. Cheng, A. Divakaran, and H. Sawhney, "Multimedia event recounting with concept based representation," in *Proc. 20th ACM Int. Conf. Multimedia*, 2012, pp. 1073–1076.

[123] C. Gan, N. Wang, Y. Yang, D.-Y. Yeung, and A. G. Hauptmann, "DevNet: A deep event network for multimedia event detection and evidence recounting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2568–2577.

[124] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," vol. 2, no. 3, p. 5, Oct. 2017, *arXiv:1710.00935*. [Online]. Available: https://arxiv.org/abs/1710.00935

[125] W. Samek, T. Wiegand, and K.-R. Müller, "Explainable Artificial Intelligence: Understanding, visualizing and interpreting deep learning models," Aug. 2017, *arXiv:1708.08296*. [Online]. Available: https://arxiv.org/abs/1708.08296

[126] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digit. Signal Process.*, vol. 73, pp. 1–15, Feb. 2017.

[127] K. Amarasinghe, K. Kenney, and M. Manic, "Toward explainable deep neural network based anomaly detection," in *Proc. 11th Int. Conf. Hum. Syst. Interact. (HSI)*, Jul. 2018, pp. 311–317.

[128] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 818–833.

[129] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, "Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model," *Ann. Appl. Statist.*, vol. 9, no. 3, pp. 1350–1371, 2015.

[130] B. M. Lake, R. Salakhutdinov, and J. B. Tenebaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[131] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 1135–1144.

[132] D. Gunning, *Explainable Artificial Intelligence (XAI)*, document Broad Agency Announcement DARPA-BAA-16-53, DARPA, Arlington County, VA, USA, 2016.

[133] T. Miller, P. Howe, and L. Sonenberg, "Explainable AI: Beware of inmates running the asylum or: How I learnt to stop worrying and love the social and behavioural sciences," Dec. 2017, *arXiv:1712.00547*. [Online]. Available: https://arxiv.org/abs/1712.00547

[134] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[135] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[136] L. Deng, "The MNIST database of handwritten digit images for machine learning research [best of the Web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

DAVID MULVEY received the B.A. degree from the University of Cambridge, in 1978, and the M.Sc. degree in digital control systems from Bristol University, in 1986. After a career in systems engineering for telecommunications and mission-critical control, he is currently pursuing the Ph.D. degree with the 5G Innovation Center, University of Surrey, U.K. His main research interests include the application of deep learning techniques to the management of cellular radio networks. He is also a Fellow of the IET.

CHUAN HENG FOH received the M.Sc. degree from Monash University, Australia, in 1999, and the Ph.D. degree from the University of Melbourne, Australia, in 2002. After his Ph.D. degree, he spent six months as a Lecturer at Monash University. From 2002 to 2012, he was an Assistant Professor with Nanyang Technological University, Singapore. He is currently a Senior Lecturer with the University of Surrey. He has authored or coauthored over 100 refereed articles in international journals and conferences. His research interests include protocol design and performance analysis of various computer networks, including wireless local area and mesh networks, mobile ad hoc and sensor networks, the Internet of Things, 5G networks, and data center networks. From 2015 to 2017, he has served as the Vice-Chair (Europe/Africa) of the IEEE Technical Committee on Green Communications and Computing (TCGCC). He is also an Associate Editor of IEEE Access and the IEEE Wireless Communications.

MUHAMMAD ALI IMRAN received the M.Sc. (Hons.) and Ph.D. degrees from the Imperial College London, U.K., in 2002 and 2007, respectively. He has over 18 years of combined academic and industry experience, working primarily in the research areas of cellular communication systems. He is currently the Vice Dean of the Glasgow College, UESTC, and a Professor of communication systems with the School of Engineering, University of Glasgow. He is also an Affiliate Professor with The University of Oklahoma, USA, and a Visiting Professor with the 5G Innovation Centre, University of Surrey, U.K. He is also a Senior Fellow of the Higher Education Academy (SFHEA), U.K. He has been awarded 15 patents, has authored/coauthored over 300 journal and conference publications, and has been a Principal/Co-Principal Investigator on over £6 million in sponsored research grants and contracts. He has supervised over 30 successful Ph.D. graduates. He has an Award of Excellence in recognition of his academic achievements, conferred by the President of Pakistan. He also received the IEEE Comsoc's Fred Ellersick Award, in 2014, the FEPS Learning and Teaching Award, in 2014, and the Sentinel of Science Award, in 2016. He was twice nominated for the Tony Jean's Inspirational Teaching Award. He was a shortlisted finalist for The Wharton-QS Stars Awards, in 2014, the QS Stars Reimagine Education Award, in 2016, for innovative teaching, and the VC's Learning and Teaching Award at the University of Surrey.

RAHIM TAFAZOLLI is currently a Professor and the Director of the 5G Innovation Center, Institute for Communications Systems, University of Surrey, U.K. He is also the Head of one of Europe's leading research groups. He has authored over 500 research articles in refereed journals and international conferences and has been an Invited Speaker. He edited two books: *Technologies for the Wireless Future* (Wiley, 2004 and 2006). He was appointed as a Fellow of the Wireless World Research Forum, in 2011, in recognition of his personal contribution to the wireless world.