IEICE TRANS. COMMUN., VOL.E107–B, NO.6 JUNE 2024

PAPER

# LSTM Neural Network Algorithm for Handover Improvement in a Non-Ideal Network Using O-RAN Near-RT RIC

**Baud Haryo PRANANTO**[†a)], ***Student Member***, **ISKANDAR**[†b)], **HENDRAWAN**[†c)], ***and*** **Adit KURNIAWAN**[†d)], ***Nonmembers***

**SUMMARY** Handover is an important property of cellular communication that enables the user to move from one cell to another without losing the connection. It is a very crucial process for the quality of the user's experience because it may interrupt data transmission. Therefore, good handover management is very important in the current and future cellular systems. Several techniques have been employed to improve the handover performance, usually to increase the probability of a successful handover. One of the techniques is predictive handover which predicts the target cell using some methods other than the traditional measurement-based algorithm, including using machine learning. Several studies have been conducted in the implementation of predictive handover, most of them by modifying the internal algorithm of existing network elements, such as the base station. We implemented a predictive handover algorithm using an intelligent node outside the existing network elements to minimize the modification of the network and to create modularity in the system. Using a recently standardized Open Radio Access Network (O-RAN) Near Realtime Radio Intelligent Controller (Near-RT RIC), we created a modular application that can improve the handover performance by determining the target cell using machine learning techniques. In our previous research, we modified The Near-RT RIC original software that is using vector autoregression to determine the target cell by predicting the throughput of each neighboring cell. We also modified the method using a Multi-Layer Perceptron (MLP) neural network. In this paper, we redesigned the neural network using Long Short-Term Memory (LSTM) that can better handle time series data. We proved that our proposed LSTM-based machine learning algorithms used in Near-RT RIC can improve the handover performance compared to the traditional measurement-based algorithm.

***key words:*** *cellular, handover, 5G, LTE, machine learning, lstm, neural network*

## 1. Introduction

Handover is one of the crucial processes in cellular communication especially in high mobility users such as vehicular terminals. This process can be defined as the process that prevents ongoing communication from getting interrupted as the mobile equipment changes its attachment point such as cells [1]. However, some disruptions may occur in active communication due to packet losses and delays and these disruptions may result in significant loss of performance [2]. In the 5G era, this handover process is getting more crucial due to the usage of a higher frequency spectrum [3] that causes a smaller cell range.

The traditional handover algorithm is usually reliable in the ideal network condition. However, in some non-ideal network conditions, such as the presence of a coverage hole, this algorithm may not be reliable and result in transmission failure. We prove this through our simulation described in Sect. 8.

Apart of the traditional handover algorithm that will be described in Sect. 2, many other algorithms are proposed including the machine-learning-based algorithms [4] (described in Sect. 3). Neural networks are one of the most popular machine-learning-based methods for handover improvement [5]–[10]. The main issue with those proposed methods is their real-world implementation because machine learning is not originally part of the cellular networks [11]. Most of the machine learning algorithms to improve handover performance require major modifications in the existing cellular networks for their implementation. This will raise many problems in the network deployment and implementation stage.

O-RAN Alliance consortium [12] introduces the Near Real Time Radio Intelligent Controller (Near-RT RIC), a new additional network element in the radio access network (RAN) that can host applications to control base stations. Using this Near-RT RIC, a machine learning algorithm for improving the handover process can be implemented modularly without major modifications to the existing cellular networks. Our implementation of Near-RT RIC will be further described in Sect. 4.

In our previous papers, we described the implementation of machine learning in Near-RT RIC to take advantage of its modularity aspect. The machine learning algorithm can be implemented modularly outside the base station without modifying the current software of the base station. We proved that this method performs better compared to the traditional handover algorithms in a simulated non-ideal network, in this case, a network with coverage holes. We measured the performance in terms of data transmission (i.e., file download) success rate if the user moves along the network and performed a handover. The target cell is determined by several methods, the traditional algorithm and our proposed machine-learning-based algorithm.

As described in [13], we modified the Near-RT RIC original software to fit our simulation case. We modified the vector autoregression (VAR) algorithm used the original in Near-RT RIC software to consider the UE movement and

Manuscript received August 18, 2023.
Manuscript revised November 19, 2023.
Manuscript publicized January 30, 2024.

[†]The authors are with School of Electrical Engineering and Informatics, Bandung Institute of Technology, Indonesia.
a) E-mail: baud.prananto@students.itb.ac.id
b) E-mail: iskandar@itb.ac.id
c) E-mail: hendrawan@itb.ac.id
d) E-mail: adit@itb.ac.id

DOI: 10.23919/transcom.2023EBP3139

compared the performance of this proposed method with the traditional handover algorithm. In the simulation result, we showed that this method can improve the handover performance in a network with a coverage hole.

In our next publication ([14], we extended our research by replacing the VAR algorithm with a Multi-Layer-Perceptron (MLP) neural network. It is proven that this method is also superior compared to the traditional handover algorithm. However, this simple neural networks still underperformed the VAR method. In this paper, we performed further improvement in the neural network to increase the performance.

### 1.1 Research Motivation and Contribution

The motivation of this research is to improve the machine learning method implemented in Near-RT RIC to solve the handover reliability issue in a non-ideal network. Using Near-RT RIC, the machine learning algorithm to control the handover process can be implemented modularly without major modification of the existing network elements.

In our previous research, we used a simple MLP neural network and it was still underperformed since it did not consider the time-series nature of the input data. We used the user measurement data as the input to determine the target cell in the handover process. In our MLP design, we statically use several last measurement data as the input and thus cannot consider the temporal feature of the time-series input. We suggest that this was the main cause of the underperformance.

The contribution of this research is designing and implementing a Long Short-Term Memory (LSTM)-based handover algorithm in Near-RT RIC to control the handover process. LSTM is chosen as it is better to handle the time-series data and consider the temporal feature of the data. We consider this time-series data handling for machine learning-based handover as our novel contribution.

In this paper, we performed modifications in Near-RT RIC original software, more precisely in the QoE Predictor xApp. We have done two modifications in our previous research and we will briefly review them in Sect. 5: adapt the VAR (the original algorithm used in Near-RT RIC) to consider the UE movement and replaced the vector autoregression with MLP neural network. Our newly proposed method to improve our previous ones is the LSTM neural network that will be described in Sect. 6. We have done simulations to test the performance of those methods and we also studied the effect of training data amount on the handover performance (described in Sect. 7). Finally, we compared the performance of our proposed methods with the traditional handover algorithm and showed that the machine-learning-based handover in Near-RT RIC performs better in a non-ideal condition, in this case, a network with a coverage hole (described in Sect. 8).

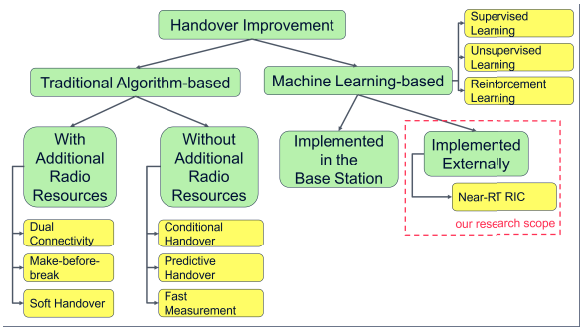


**Fig. 1** Solution taxonomy for handover improvement.

### 1.2 Research Scope and Limitation

This research focuses on the usage of a machine learning algorithm in Near-RT RIC to control the handover process. In this study, we compare the performance of the handover control in Near-RT RIC with the baseline traditional handover algorithm.

There are various solutions to improve the handover performance and we organized the solutions taxonomy in Fig. 1. We focus only on the machine-learning-based solution that is implemented externally for modularity reasons, and we compare the result of our methods with the baseline traditional handover algorithms. The innovations that are implemented on top of the traditional handover algorithm such as soft handover, conditional handover, make-before-break, are not considered and not compared with our proposed machine-learning-based algorithm in Near-RT RIC. The other state-of-the art machine learning algorithms for handover improvement as described in Sect. 3 are also not compared with our method.

### 1.3 Paper Organization

In this Sect. 1, we provide a gentle overview of our research, problem statement, motivation, contribution, and the scope of this study.

Section 2 describes the Traditional Handover Algorithm, the baseline algorithm for cellular mobility management that we would like to improve using our research. We present the way of working, the issues, limitations, and room of improvement of this traditional algorithm.

We use machine learning techniques to improve the Traditional Handover Algorithm. In Sect. 3, we describe the State of The Art of the usage of Machine Learning for handover improvement. Here we explored the previous works conducted to improve the handover algorithm using various machine learning techniques.

Our solution utilizes an O-RAN new network element called Near-RT RIC, which is described in Sect. 4. Here we also describe how the software is implemented and used for our solution.

Next in Sect. 5, we reviewed our previous methods that we already published in earlier publications [13], [14]. Here

we explain how we modified the original Near-RT RIC software to utilize our proposed machine learning methods: the modified VAR and MLP Neural Network.

In Sect. 6, we introduce our newly proposed method to be used in Near-RT RIC using LSTM. Here we explain the novel algorithm to improve the handover performance.

To test our proposed method, we designed a simulation that will be explained in Sect. 7. We also explain how we collect and utilize the data to prove the effectiveness of our proposed methods.

The result of our simulation with the proposed methods is discussed in Sect. 8. We show the improvement in handover performance compared to the traditional handover algorithm. The overall conclusion and possible future works are written in Sect. 9.

## 2. Traditional Handover Algorithm

In the traditional handover algorithm [15], the UE sends measurement reports to the serving base station about the condition of serving cell and neighbor cells. The measurement report is about the cell's signal strength (Reference Signal Received Power - RSRP) and/or signal quality (Reference Signal Received Quality - RSRQ). The serving base station will analyze the measurement report to determine the target cell for the handover destination. Usually the target cell is the best-measured neighbor cell (Fig. 2).

The handover process may interrupt the data transmission because it caused temporary disconnection of the UE from the serving cell (thus stopping the data transmission) and connects again to the target cell. The interruption is defined as Mobility Interruption Time (MIT) and 3GPP defines MIT as the shortest time duration supported by the system during which a user terminal cannot exchange user plane packets with any base station during transitions [17]. MIT can be calculated as [2]:

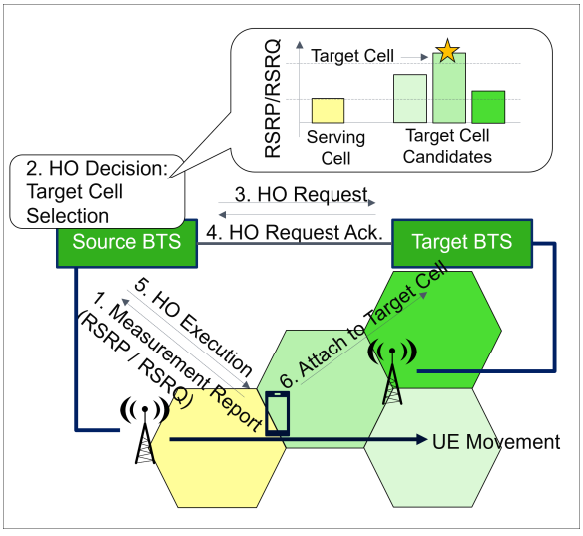


**Fig. 2** Traditional handover algorithm [16].

$$T_{\mathrm{MIT}} = \{(1 - P_{\mathrm{HOF}}) \times T_{\mathrm{HIT}}\} + \{P_{\mathrm{HOF}} \times T_{\mathrm{HOF}}\} \quad (1)$$

$T_{\mathrm{MIT}}$ = Total MIT
$P_{\mathrm{HOF}}$ = Probability of either a handover failure (HOF) or a radio link failure (RLF) during handover
$T_{\mathrm{HIT}}$ = Handover Interruption Time, MIT in a successful handover
$T_{\mathrm{HOF}}$ = Handover Failure Time, MIT in a HOF or RLF

The $T_{\mathrm{HOF}}$ contributes more significantly to MIT ($T_{\mathrm{MIT}}$), thus reducing the $T_{\mathrm{MIT}}$ can be better done by reducing $P_{\mathrm{HOF}}$. In LTE Network, $T_{\mathrm{HIT}}$ is reported around 50 ms while $T_{\mathrm{HOF}}$ ranges from several hundred milliseconds to a few seconds [18]. This means the best way is to avoid unnecessary handovers or handovers to the wrong cell. Target cell determination is very crucial in the handover process to minimize MIT.

The traditional handover algorithm is reliable in ideal conditions, where RSRP/RSRQ measurement always reflects the real condition of the network. Using this algorithm, the best target cell to continue the network connection is always the cell with the best RSRP/RSRQ measurement. In a non-ideal condition, the RSRP/RSRQ measurements may not reflect the real network condition.

An example of this non-ideal network is the presence of a cell coverage hole due to an obstacle. A UE may be handed over to a target cell with the best RSRP/RSRQ, but it enters the target cell's coverage hole after the handover, and the connection fails after that. In this case, the traditional handover algorithm is not reliable to determine the target cell correctly and ensure network connectivity. Our simulation proves the unreliability of the traditional handover algorithm in Sect. 8. This raises the need for machine-learning-based target cell determination.

Besides the baseline traditional handover algorithm, which is part of the base station algorithm as compliance to 3GPP standard [16], there are some other handover algorithm that aims primarily to reduce MIT by reducing $P_{\mathrm{HOF}}$. Some innovations include fast measurements [19], soft-handover, dual connectivity [20], make-before-break [2], [21], conditional handover [5], [22], [23], and predictive handover [5], [24]–[28].

Fast measurement makes the source base station send a handover command before an abrupt deterioration of the radio link to the UE. The UE reacts faster to the channel changes and improves mobility robustness.

Soft handover, dual-connectivity, and make-before-break work similarly by making multiple separate connections to different radio resources simultaneously. This improves mobility robustness but increases the network complexity and requires more radio resources.

In traditional handover, the handover command is sent when the radio conditions start to get degraded [23]. Conditional handover prepares in advance multiple candidate target cells in the network. This enables the handover command to be sent to the UE earlier than at the traditional handover when the radio conditions are still good.

In predictive handover, the candidate target cells are predicted using various techniques, including user behavior and learning the network condition using machine learning techniques.

## 3. Machine Learning for Handover: State of the Art and Related Works

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed [29]. It studies the computer algorithms that improve automatically through experience [30].

Since the traditional handover algorithm is sometimes not reliable in a non-ideal network condition, some alternative methods are required to determine the target cell, and one of the approaches is the predictive handover using machine learning. Several studies [4] have implemented machine learning to improve handover performance using the predictive handover method (i.e. predict the target cell using machine learning).

Supervised learning is widely used for handover improvement. The neural networks (NN) method is one of the most popular techniques used in several studies [5]–[10]. Some studies use support vector machine [31] and K-nearest neighbor [32], [33]. Unsupervised learning techniques are also used by some studies, for example, K-means [27], [34] and Long Short-Term Memory [25]. Reinforced learning is used by some researchers that usually employ Q-learning algorithms [24], [35].

The neural networks method is popular in mobility management improvement studies. The basic idea behind these studies is to use the concept of neural networks to learn a mobility-based model for every user in the network and then make predictions of which cell the user is most likely to be next [4].

Several previous studies [6], [36] used neural networks for target cell selection in the handover process and performed simulations to justify their proposed method. We based our research on these works and improved them using our proposed methods. The previous research did not perform the implementation of the software in the Near-RT RIC platform and considered only the simple MLP neural network. As a novelty of our research, we present the implementation of the machine learning algorithm in the real Near-RT RIC platform. We also test several methods other than simple MLP neural networks, including the LSTM-based neural network that can better process time-series measurement data.

## 4. O-RAN Near-RT RIC

Open Radio Access Network (O-RAN) Alliance standardizes and introduces several new applications for open and intelligent RAN on top of the legacy cellular network. This enables the introduction of machine learning applications since machine learning is not originally part of the cellular

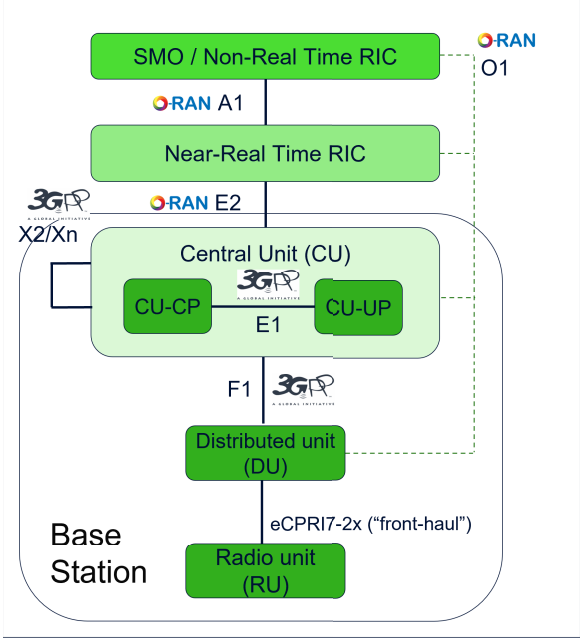


**Fig. 3** O-RAN architecture [12].

network standard. O-RAN introduces new network elements called Radio Intelligent Controller (RIC) to add intelligence to the cellular radio network. There are two variants of RIC: Near-Real Time (Near-RT) and Non-Real Time (NRT) RIC (Fig. 3).

Near-RT RIC hosts applications that require real-time response, such as mobility management applications like handover control. Because of this response requirement, Near-RT RIC is typically implemented in an Edge Cloud, a virtual environment that is placed physically near the radio network. The effectiveness of Edge Cloud is already proven to implement RAN elements [37].

NRT-RIC hosts applications that do not require immediate response such as network monitoring and optimization. It can be implemented in Central Cloud and typically colocated with the existing network management system.

There are some use cases defined by O-RAN Alliance [38] to be implemented in O-RAN to provide RAN openness and intelligence, for example, Context-Based Dynamic HO Management for V2X, Flight Path-Based Dynamic UAV Radio Resources Allocation, QoE Optimization, and Traffic Steering. However, the exact implementation of the use case is given to specific vendors. For example, Nokia prioritizes Traffic Steering and Network Anomaly Detection use case for its RIC solution [39].

Several studies already use O-RAN RIC architecture for many applications such as connection management [40], mobility management [41], and scheduling policy optimization [42]. Various machine learning algorithms are implemented in RIC including reinforcement learning [43].

The Near-RT RIC can be implemented in any virtualized environment. In our research, we installed it on an Ubuntu-based virtual machine by installing the open-source

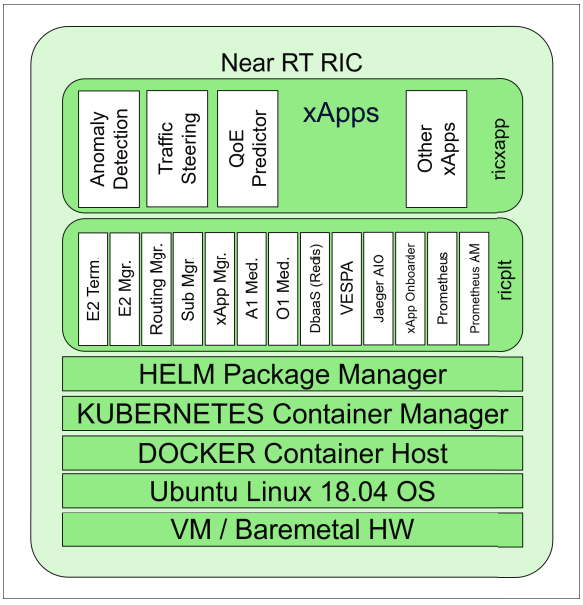


**Fig. 4** The software architecture of RIC [44].

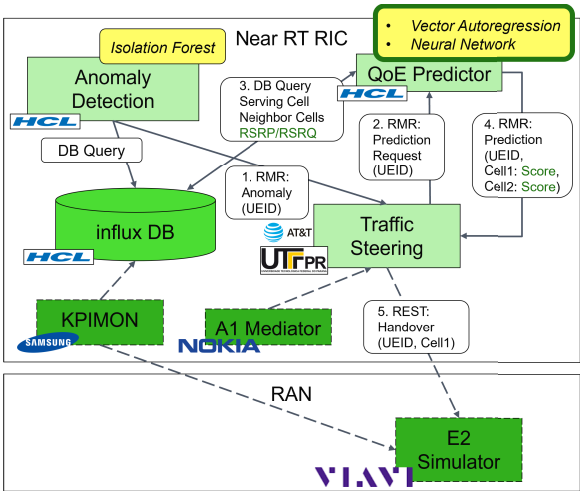


**Fig. 5** Anomaly detection use case of Near-RT RIC [45].

software provided by the O-RAN Software Consortium (SC) [44]. The software is container-based and contains several applications called xApps. This architecture can be viewed in Fig. 4.

The Anomaly Detection use case [45] is one of the already existing Near-RT RIC use case examples from O-RAN SC that mostly corresponds to our research need. However, we have to perform some modifications to fit our simulation scenario.

The software contains three xApps: Anomaly Detection, Traffic Steering, and Quality of Experience (QoE) Predictor. The xApps exchange messages in RMR protocol, the Near-RT RIC internal communication. Currently, in this research, the Near-RT RIC works stand alone without any connection to the RAN, and all simulation data is stored in the database.

The scenario begins with the Anomaly Detection xApp detects an anomalous UE, for instance, the UE experiencing degradation of RSRP. In this research, this information is obtained from the database but in the real implementation, this information is notified by RAN (i.e. base station). The Anomaly Detection xApp then informs the anomaly to the Traffic Steering xApp.

Traffic Steering xApp then sends a message to QoE Predictor xApp sending the identity of the UE experiencing an anomaly. QoE Predictor xApp then predicts the score of QoE of the UE if the UE is placed in the neighboring cells. In the original software, this score is the throughput of the data transmission and is predicted using the vector autoregression (VAR) method. Therefore, QoE Predictor predicts the throughput experienced by the UE if it is placed in a certain cell.

This prediction is sent back to Traffic Steering xApp. Based on this prediction, it will perform some necessary actions. The action can be a handover command to the cell where the throughput prediction is the highest one. From this scenario, it is clear that QoE Prediction is the one that actually determines the target cell by performing a prediction of QoE (score) in each cell. The Traffic Steering xApp is just simply choosing the target cell with the highest score.

In this research, we modified the original Near-RT RIC xApps in the Anomaly Detection Use Case to adapt to our simulation scenario. We mainly performed modifications in QoE Predictor xApp as it is the one that actually performs predictions that will determine the target cell. We performed two modifications to the original QoE Predictor xApp. The first modification is to adapt the original software to our simulation scenario. The prediction is still done by the vector autoregression method. The second modification is completely replacing the vector autoregression with a neural network. The neural network design is based on our previous studies [46], [47] that yield optimum results.

## 5. Our Previous Methods: A Review

The original QoE Predictor xApp software provided by O-RAN SC predicts the QoE using the VAR method. However, the original software is not immediately usable for our research case so we have to perform some modifications in the original xApp. Our research aims to determine the target cell in a non-ideal network containing a coverage hole. This target cell is determined by the movement of the UE that is reflected in the RSRP/RSRQ measurements.

### 5.1 Modified VAR

Vector Autoregression (VAR) is a statistical time series model used to analyze the relationship between multiple variables. In a VAR model, each variable in the system is modeled as a function of its past values and the past values of all the other variables in the system. A VAR model of order p, denoted as VAR(p), is a set of linear equations that relate each variable in the system to its own past values and

the past values of all the other variables in the system up to p lags. The equations can be written in matrix form as:

$$Y_t = A_1 Y_{t-1} + A_2 Y_{t-2} + \ldots + A_p Y_{t-p} + u_t \quad (2)$$

where $Y_t$ is a $k$-dimensional vector of the current values of the $k$ variables in the system, $A_1, A_2, \ldots, A_p$ are $k \times k$ matrices of coefficients that capture the dynamic relationships between the variables at lags 1 to $p$, and $u_t$ is a $k$-dimensional vector of error terms that represent the unexplained part of the system at time $t$.

The original QoE Predictor xApp determines the target cell by predicting the throughput of each cell using time-series throughput data in the training data. However, this software only considers the position of the UE, i.e. what the neighbor cells are. It does not consider the movement and the trajectory angle of the UE. If we use the unmodified original software and training data, the prediction will always give the same target cell for all simulation cases.

Our proposed modified method using VAR can be expressed in the following pseudocode (Algorithm 1). The *italic* expression in Algorithm 1 indicates our modification.

**Algorithm 1** Predict the throughput of all cells

**Require:** list of all cells serving and neighboring UE
*and the RSRP measurements of those cells*
**for all** cells in list **do**
Query throughput of cell over time from Training Data
*(where RSRP measurement is similar with the one reported by UE)*
*Remove outliers of the query result*
Predict the next throughput of the cell using Vector Autoregression
**end for**
Report the throughput prediction of all cells in list to Traffic Steering xApp

To adapt the original software to our simulation scenario, we reconstructed the software and training data to put the UE movement into account. The UE movement and its trajectory angle can be reflected by the RSRP measurement variations. From the training data generation process described in Sect. 7, we construct the new training data that considers the UE movement to predict the next throughput by evaluating RSRP values. Using this modified training data, the target cell is determined by the UE movement, not only the UE position like the original QoE Predictor xApp.

### 5.2 MLP Neural Network

For our second method, we completely replaced the VAR in the QoE Predictor xApp with a neural network. In this preliminary stage, we use a very simple Multi-Layer Perceptron (MLP) neural network regression model to predict whether the download is successful or failed using RSRP and RSRQ samples as input.

This method works in a different approach than the previous one. The VAR method views the problem as a prediction problem, this MLP neural network method views the problem as a classification problem. The prediction problem in the VAR method requires throughput data and determines the target cell by choosing the neighbor cell with the highest predicted throughput. The throughput data is not immediately available in real-life handover cases so we decide to use the already available RSRP/RSRQ data. It is possible to perform prediction of RSRP/RSRQ data but in a non-ideal network, RSRP/RSRQ data do not directly reflects the throughput or QoE of the user.

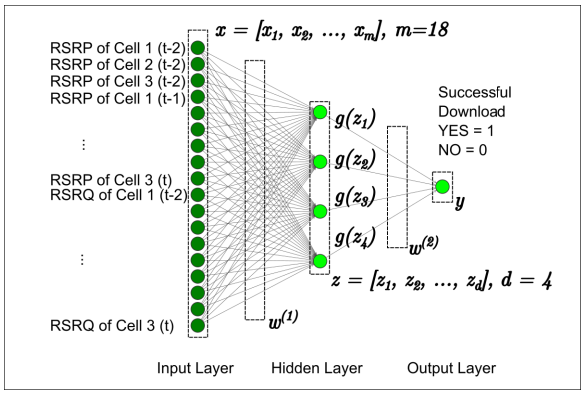


**Fig. 6** MLP neural network design.

In our second method, we decided to view the problem as a classification problem, without necessarily predicting the future RSRP/RSRQ data. We directly collect the RSRP/RSRQ data reported by the UE to determine whether it is good or not to perform a handover in a certain neighbor cell.

The MLP neural network in our method contains fully interconnected 18 input nodes, 4 hidden nodes, and 1 output node (Fig. 6). The inputs are the last 3 samples of RSRP and RSRQ measurements from all the 3 cells. The output is whether the data transmission (i.e. file download) is successful or not, represented by the number 0 (failed download) or 1 (successful download). The result of the output node is a floating point continuous number between 0 and 1 that can be used as the prediction score. The score will then be sent to Traffic Steering xApp and the cell with the highest score will be determined as the target cell. In this method, we do not need throughput data and perform any prediction to determine the target cell.

We use Tensor Flow Keras API for the implementation. Currently, the training process is done with 150 times iterations through the whole training data (*epoch = 150*), and the model is updated every 10 training data (*batch size = 10*).

The MLP neural network method has a simpler implementation than the VAR method since it uses only RSRP/RSRQ measurement without a throughput measurement. This MLP neural network method can be faster as the neural network model can be saved and reused without necessarily querying the training data on each prediction.

As a summary, our two previous methods can be compared in Table 1.

**Table 1** Comparison of the previous methods: VAR and MLP-NN.

| | VAR | MLP-NN |
|---|---|---|
| Approach | Prediction | Classification |
| Training Data Components | RSRP/RSRQ, Throughput | RSRP/RSRQ, Download Status |
| Training Data Usage | Queried from the database on each simulation | Only during the NN model creation (first simulation) |
| Method for Inference | Query the time series throughput data (filtered with RSRP/RSRQ) from the Training Data and predict the next throughput for each cell | Provide RSRP/RSRQ as input to the NN model and perform classification to obtain successful download probability (score) from each cell |

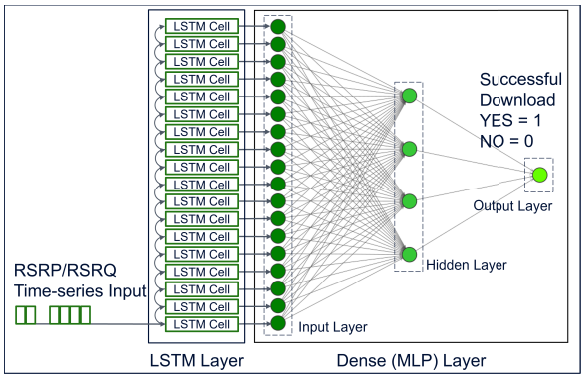


**Fig. 7** LSTM neural network design.

## 6. Proposed Method: LSTM Neural Network

The two previous methods can outperform the traditional handover algorithm. However, the MLP neural network still underperforms the VAR method. We view the neural network as more promising since it has a less complex and less demanding implementation. It requires only RSRP/RSRQ measurements with no throughput measurement and the model can be created once and used repeatedly in every case. Therefore, we seek ways to improve this neural network.

MLP has several limitations and weaknesses: it is stateless, unaware of temporal structure, has messy scaling, and requires fixed-sized inputs and outputs [48]. In our scenario, the input is RSRP/RSRQ measurements which are time-series data, not static fixed-sized data. The length of the input is actually not fixed, depending on the cases. Using MLP, we have to fixate the input to only 3 samples per cell and thus limit the amount of information. Our MLP neural network also has an issue with scaling. When we add or remove the cell amount in the network we have to completely change the model architecture.

In machine learning, time-series data can be handled in various ways. Time-series prediction involves predicting the next value for a given input sequence, time-series classification involves predicting a class label for a given input sequence, and time-series generation involves generating a new output sequence that has the same general characteristics [48].

We assess and identify our scenario as a time-series classification problem because our input data has a temporal structure and we solve the problem by classifying if the file download is successful or not. There are so many methods to solve this time-series classification problem [49], including the deep learning approach using neural network [50]. Instead of using a simple MLP neural network for time-series data, it is recommended to use a Recurrent Neural Network (RNN) that better considers the temporal feature of the input. There are several methods based on RNN, for example, SimpleRNN, Gated Recurrent Unit (GRU), and Long Short-term Memory (LSTM).

Long Short-term Memory (LSTM) [51] is an artificial neural network that has a feedback connection and thus can be classified as RNN. LSTM has been shown to outperform other RNN methods on numerous temporal processing tasks [52]. These temporal processing tasks include the processing of multivariate time-series data to perform predictions on future values. Several applications employ LSTM due to this capability, for example, handwriting recognition, speech recognition, and machine translation.

LSTM employs the "Long-term memory" and "Short-term memory" that occurs in the RNN architecture that processes time-series data. The connection weights and biases in the network change once per episode of training, analogous to how physiological changes in synaptic strengths store long-term memories; the activation patterns in the network change once per time-step, analogous to how the moment-to-moment change in electric firing patterns in the brain store short-term memories [53]. The LSTM architecture aims to provide a short-term memory for RNN that can last thousands of timesteps, thus "Long Short-Term memory". LSTM networks are well-suited to classifying, processing, and making predictions based on time series data since there can be lags of unknown duration between important events in a time series.

We modify our neural network by adding an LSTM layer before the MLP layer. This will enable the neural network to process the time-series input data before feeding them to the MLP to perform classification (Fig. 7).

The LSTM layer contains 18 serially-connected LSTM cells. The input is the RSRP/RSRQ data fed into the first LSTM cell and processed serially to the next LSTM cells. Besides providing input to the next LSTM cell, all 18 LSTM cells also provide input to the MLP's 18 input layers that will further perform the classification function. Using this architecture, any arbitrary length of input can be processed, unlike the MLP neural network that requires fixed-length input. The dropout rate is chosen 50% and the MLP layer architecture is the same with the previous method 18 input nodes, 4 hidden nodes, and 1 output node (Fig. 6). The activation function is *relu* in the hidden layer and *sigmoid* in the output layer. The training process is done with 150 times iterations through the whole training data (*epoch* = *150*), and the model is updated every 10 training data (*batch size* = *10*).

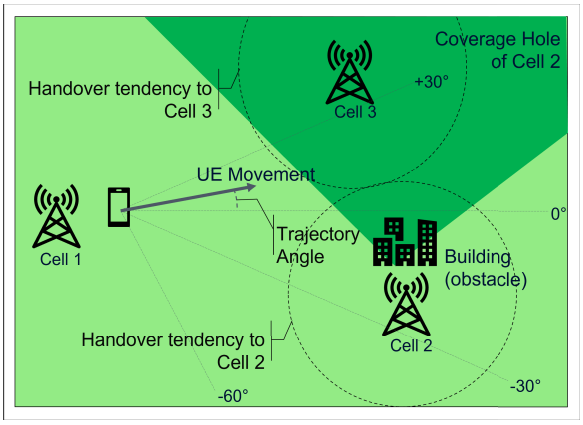


**Fig. 8** Environment for simulation with 3 cells and 1 coverage hole.

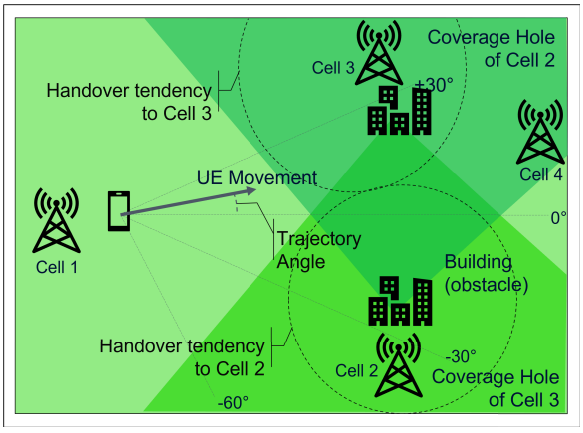


**Fig. 9** Environment for simulation with 4 cells and 2 coverage holes.

## 7. Simulation Design and Data Collection

In this research, we created two network environments for two experiments. The first one contains three cells, one moving UE, and a building creating a coverage hole (Fig. 8). The second one contains four cells, one moving UE, and two buildings creating two coverage holes (Fig. 9). This environment is built using NS3 LTE network simulator [54] based on previous studies[33], [36]. The simulation parameters are reusing the previous work as described in Table 2.

On each simulation, the UE moves to the right side of the network with a random trajectory angle. Due to this movement, the UE needs to perform a handover from Cell 1 to either Cell 2 or Cell 3 (or also Cell 4 for 4 cells simulation), depending on the trajectory angle. The UE also downloads files during the movement and in the end, the download may be successful or may not. For every simulation, we noted down the target cell, the download success status, and the RSRP/RSRQ measured by UE.

The simulation activity can be described in Fig. 10. Our simulation contains three activities: the training data generation (1), handover simulations using the traditional algorithm (2), the target cell determination using Near-RT RIC (3), and the handover result verification of the target cell determined by Near-RT RIC (4).

**Table 2** NS3 simulation parameters.

| Parameter | Value |
|---|---|
| System bandwidth | 5 MHz |
| Inter-site distance | 500 m |
| Adaptive Modulation and Coding Scheme | MiErrorModel |
| Simulation area | $2000 \times 2000$ m$^2$ |
| Number of base stations | 3 and 4 |
| Transmit Power | 46 dBm |
| Number of UEs | 3 and 4 (1 moving) |
| Velocity of UE1 | 16.6667 m/s |
| Path Loss Model | Cost 231 |
| Antenna Height | 30 m |
| Obstacle Height | 35 m |
| File Size | 15 MB |

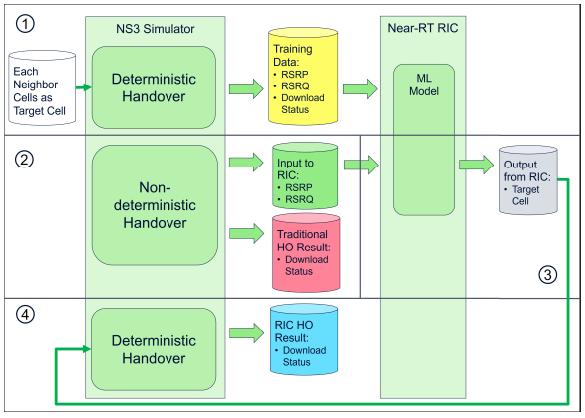


**Fig. 10** Simulation activities done for this research.

To create the training data, we ran 100 simulations of deterministic handover to each neighbor cell. The first 100 simulations are deterministic handover cases where the UE is forced to perform handover to Cell 2 regardless of the trajectory angle. The next 100 simulations are also deterministic handover cases but this time to Cell 3. There are also the next 100 simulations to Cell 4 for 4 cells simulation. This activity is described as the first activity in Fig. 10.

To compare the performance of the handover algorithms (the traditional algorithm and our RIC-based proposed algorithms), we ran other simulations of non-deterministic handover. In these simulations, the UE may perform a handover to any neighbor cell, using a traditional handover algorithm, based on the RSRP/RSRQ measurements.The result of these simulations (download success status and RSRP/RSRQ measurement) is the result of the traditional handover algorithm and is used as a baseline to be compared with machine-learning-based algorithms run in Near-RT RIC. The RSRP/RSRQ measurement for these simulations is also used as input for the RIC-based handover algorithm. This activity is described as the second activity in Fig. 10.

Next, we performed RIC-based handover simulations. For each simulation run, we performed target cell determination using the machine-learning-based algorithm in Near-

RT RIC by providing RSRP and RSRQ measurements of the same simulations that we ran in the traditional handover algorithm process. The algorithm in Near-RT RIC would then get the score of each existing neighbor cell in the network. The cell with the highest score is then chosen as the target cell. This activity is described as the third activity in Fig. 10.

From Near-RT RIC we only obtained the target cell, but not yet the download success status. Therefore, we need to perform verification using NS3 to check if the download is successful or not, given the target cell from Near-RT RIC. Next, we performed deterministic handover again using NS3 but using the target cell obtained by Near-RT RIC. From here we get the download success status if the handover is controlled by Near-RT RIC. This activity is described as the fourth activity in Fig. 10.

We choose download success rate as the main performance metric. As stated in Eq. (1) in Sect. 2, the handover process is best improved by reducing the probability of handover failure, thus avoiding unnecessary handover and handover to a wrong cell. Based on this statement, we focus on the target cell determination process. We decide the performance metric as download success rate if we use a certain method to select the target cell.

## 8. Simulation Result and Discussion

As described in Sect. 7, we already performed three sets of simulations in our previous papers: the traditional handover, Near-RT RIC handover using VAR, and Near-RT RIC using MLP neural network. In this paper, we propose an additional method which is Near-RT RIC using LSTM neural network and we also performed another set of simulations. The traditional handover was done using the NS3 simulator and we noted down the RSRP/RSRQ measurement and the handover results (target cell and download success status). The RSRP/RSRQ measurement of those simulations was used as input in Near-RT RIC handover simulations. After that, we compared the download success rate of all simulations among all methods (Fig. 11 and Fig. 12). We performed all those simulations in two network scenarios: 3 cells with 1 coverage hole and 4 cells with 2 coverage holes.

The successful download rate for the traditional handover algorithm, in 3 cells 1 coverage hole environment, is 86.2%, not 100% due to the presence of the coverage hole. All of the simulations with failed downloads happened when the UE was handed over to Cell 2 (based on the best RSRP/RSRQ measurement) but it entered the coverage hole behind the building after the handover. If it was handed over to Cell 3 instead of Cell 2, the download may be successful because Cell 3 was not obstructed by the building. This result shows that sometimes the traditional handover algorithm is not reliable in a non-ideal condition.

This successful download rate for the traditional handover algorithm is getting worse in the 4 cells 2 coverage holes environment, which reached only 29%. The two coverage holes created a blank spot in the network that caused download failure if the UE performed a handover to either

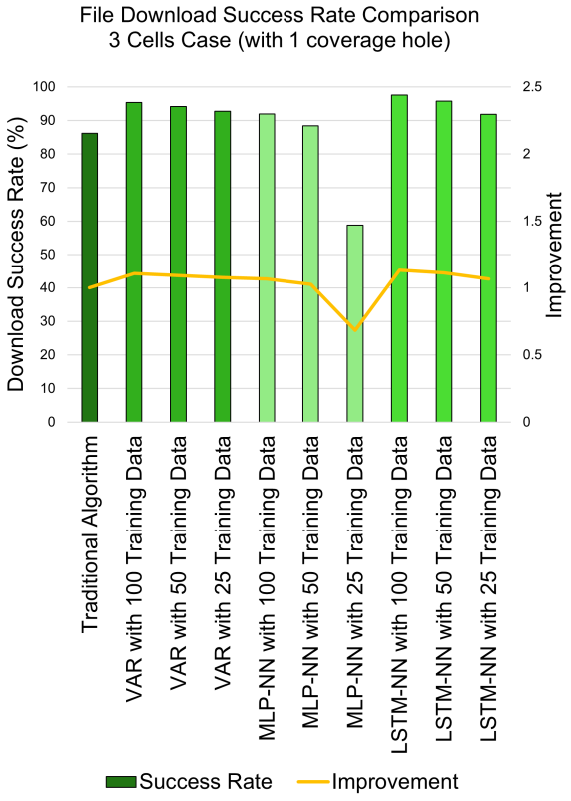


**Fig. 11** Simulation result comparison for a network with 3 cells and 1 coverage hole.

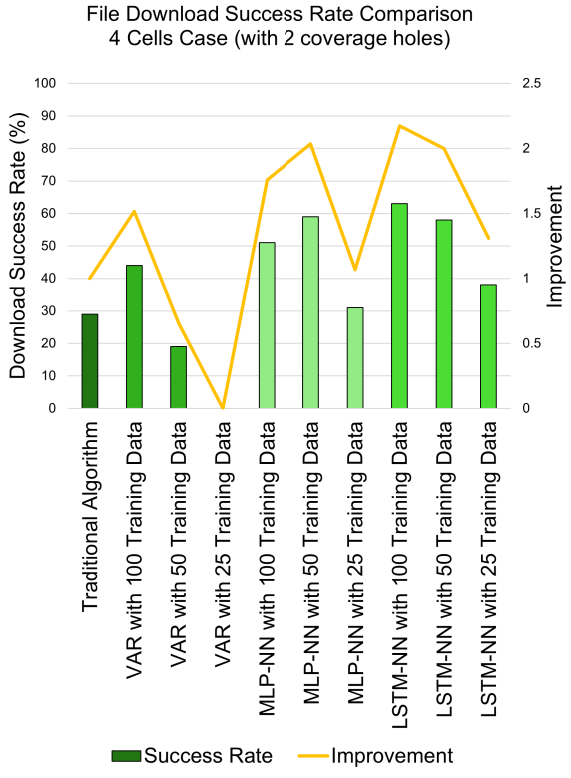


**Fig. 12** Simulation result comparison for a network with 4 cells and 2 coverage holes.

**Table 3** Comparison of the handover methods.

| Method | Traditional | Related Work[6] | Modified VAR | MLP-NN | LSTM-NN |
|---|---|---|---|---|---|
| Approach | Selection | Classification | Prediction | Classification | Classification |
| Brief Description | Select the target cell from the neighbor cell (as the candidate target cell) that has the best RSRP and/or RSRQ. | Provide the RSRP and RSRQ measurement to an MLP neural network model to get the score of a candidate target cell. The score of all candidate cells is then compared and the highest score is chosen as the target cell. | Predict the future data rate of a candidate cell from the pool of training data. The candidate cell with the highest predicted data rate is chosen as the target cell. | Provide the RSRP and RSRQ measurement to an MLP neural network model to get the score of a candidate target cell. The score of all candidate cells is then compared and the highest score is chosen as the target cell | Provide the RSRP and RSRQ measurement to the LSTM network and continue to the MLP model to get the score of a candidate target cell. The score of all candidate cells is then compared and the highest score is chosen as the target cell |
| Best Performance (download success rate in 1-coverage-hole network) | 86.2% | 95.37% | 95.3% | 91.9% | 97.6% |

Cell 2 or Cell 3. The UE may experience a successful download if it performed a handover to Cell 4 but Cell 4 is never an option in the traditional handover algorithm as the RSRP/RSRQ are too low at the time of handover.

When we determined the target cell using Near-RT RIC in the 3 cells 1 coverage hole network environment, the successful download rates are mostly increasing, depending on the method and the amount of training data (Fig. 11). If the QoE Predictor xApp uses vector autoregression (VAR), the success rate can reach 95.3% using all 100 available training data, 94.1% with 50 training data, and 92.7% with only 25 training data. If we use MLP-NN, the success rate is slightly lower but still higher than the traditional algorithm in most cases, the download success rate can reach 91.9% using all 100 available training data and 88.4% using only 50 training data. However, the performance plummeted to only 58.8% if we only use 25 training data (even lower than the traditional handover algorithm). Using our newly-proposed LSTM-NN, the success rate is superior to other methods, reaching 97.6% using 100 training data, 95.7% using 50 training data, and 91.8% using only 25 training data.

If we use Near-RT RIC to determine the target cell in the 4 cells 2 coverage holes network environment, some methods significantly can increase the success rate, given enough training data. As shown in Fig. 12, VAR didn't perform much in this network environment. Our previous MLP-NN method performed better but ultimately our newly-proposed LSTM-NN worked best for this environment.

The related work [6] reported that their proposed method achieved a download success rate of 95.37% compared to the state-of-the-art method that yields only 54.45% in their simulation. However, they did not perform the implementation in the Near-RT RIC.

It is shown that machine-learning algorithms can provide better handover performance by determining the correct target cell in a non-ideal network condition, given enough training data. The more training data, the better the performance. From the simulation result, our newly proposed LSTM-NN works better than all other methods and we proved it in both network environment cases.

As a summary, all of the handover methods discussed in this paper can be compared in Table 3. The Modified VAR and MLP-NN are our previous proposed methods [13], [14] while LSTM-NN is the current proposed method.

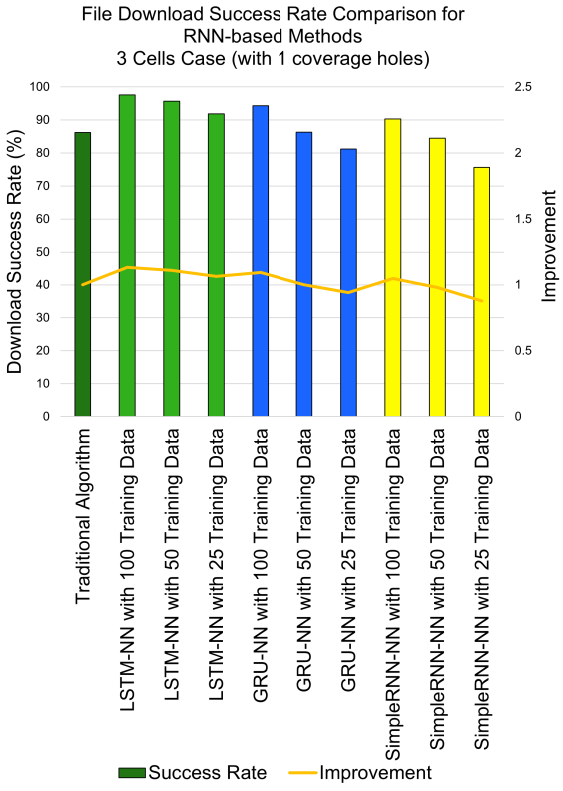


**Fig. 13** Comparison of RNN-based methods for a network with 3 cells and 1 coverage hole.

In addition, we also compared other RNN-based methods that are designed for time-series classification. We replaced the LSTM layer with the GRU and SimpleRNN layer. No changes were performed in the MLP layer and the hyperparameters. The result is similar to the LSTM method but LSTM is still superior to those methods (Fig. 13). However, those methods are very promising for future exploration.

## 9. Conclusion and Future Work

The handover process may cause an interruption in the data transmission, moreover in a high-mobility condition where the radio condition may worsen because of the user speed. Increasing the probability of successful handovers, such as making sure to perform handover to the correct target cell, can minimize this interruption. Therefore, target cell determination is very important in the handover process.

In this paper, we presented the result of our newly proposed method, the LSTM neural network, using O-RAN Near-RT RIC to determine the target cell in the handover process. This new method is an improvement of our previous machine-learning-based methods that is better at handling the time-series nature of the input data. From the simulation result, it can be concluded that this method can be used and is proven better to determine the target cell compared to other methods, the traditional handover algorithm, and our previous machine-learning-based methods. The performance of the algorithms depends on the method and the amount of training data.

In the future, we will further improve the neural network to get better performance. We plan to test this method in another non-ideal network environment other than the coverage hole case. We also plan to explore the possibilities of using another RNN-based methods such as GRU and SimpleRNN.

### References

[1] M.S. Obaidat and P. Nicopolitidis, Smart Cities and Homes: Key Enabling Technologies, Morgan Kaufmann, 2016.
[2] H.S. Park, Y. Lee, T.J. Kim, B.C. Kim, and J.Y. Lee, "Handover mechanism in NR for ultra-reliable low-latency communications," IEEE Netw., vol.32, no.2, pp.41–47, 2018.
[3] 3GPP, "NR; User Equipment (UE) radio transmission and reception; part 2: Range 2 standalone-Rel. 16," 2020.
[4] P.V. Klaine, M.A. Imran, O. Onireti, and R.D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," IEEE Commun. Surveys Tuts., vol.19, no.4, pp.2392–2431, 2017.
[5] C. Lee, H. Cho, S. Song, and J.M. Chung, "Prediction-based conditional handover for 5G mm-Wave networks: A deep-learning approach," IEEE Veh. Technol. Mag., vol.15, no.1, pp.54–62, 2020.
[6] Z. Ali, N. Baldo, J. Mangues-Bafalluy, and L. Giupponi, "Machine learning based handover management for improved QoE in LTE," Proc. NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, no.5, pp.794–798, 2016.
[7] N. Sinclair, D. Harle, I.A. Glover, J. Irvine, and R.C. Atkinson, "An advanced SOM algorithm applied to handover management within LTE," IEEE Trans. Veh. Technol., vol.62, no.5, pp.1883–1894, 2013.
[8] N.M. Alotaibi and S.S. Alwakeel, "A neural network based handover management strategy for heterogeneous networks," Proc. 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA 2015, pp.1210–1214, 2016.
[9] M.A.F. Rihani, M. Mroue, J.C. Prevotct, F. Nouvel, and Y. Mohanna, "A neural network based handover for multi-RAT heterogeneous networks with learning agent," Proc. 13th International Symposium on Reconfigurable Communication-Centric Systems-on-Chip, ReCoSoC 2018, 2018.
[10] B. Shubyn, N. Lutsiv, O. Syrotynskyi, and R. Kolodii, "Deep learning based adaptive handover optimization for ultra-dense 5G mobile networks," Proc. 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2020, pp.869–872, 2020.
[11] G. Masini, Y. Gao, and S. Sirotkin, "Artificial intelligence and machine learning in NG-RAN: New study in RAN3," 2021.
[12] O-RAN, "O-RAN: Towards an Open and Smart RAN," White Paper, Oct. 2018.
[13] B.H. Prananto, Iskandar, and A. Kurniawan, "Handover improvement using O-RAN near-RT RIC for network with coverage hole," IEICE Commun. Express, vol.11, no.1, pp.19–23, 2023.
[14] B.H. Prananto, Iskandar, and A. Kurniawan, "A new method to improve frequent-handover problem in high-mobility communications using RIC and machine learning," IEEE Access, vol.11, pp.72281–72294, 2023.
[15] J. Agrawal, P. Mor, J. Keller, R. Patel, and P. Dubey, "Introduction to the basic LTE handover procedures," 2015 International Conference on Communication Networks (ICCN), pp.197–201, 2015.
[16] 3GPP TS 36.300, "3GPP TS 36.300 V16.0.0 (2019-12) evolved universal terrestrial radio access (E-UTRA) and evolved universal terrestrial radio access network (E-UTRAN) overall description Stage 2 (Release 10)," 2013.
[17] 3GPP TR 38.913, "Study on scenarios and requirements for next generation access technologies (Release 16)," 2020.
[18] 3GPP TR 36.881, "Evolved universal terrestrial radio access (E-UTRA) study on latency reduction techniques for LTE (Release 14)," 2016.
[19] A. Chincholi, M. Menon, and L. Hsu, "Measurement gap enhancements for BL/CE UEs," World Intellectual Property Organization WO 2019/094977 Al, May 2019.
[20] M. Polese, M. Giordani, M. Mezzavilla, S. Rangan, and M. Zorzi, "Improved handover through dual connectivity in 5G mmWave mobile networks," IEEE J. Sel. Areas Commun., vol.35, no.9, pp.2069–2084, 2017.
[21] H.L. Wang, S.J. Kao, C.Y. Hsiao, and F.M. Chang, "A moving direction prediction-assisted handover scheme in LTE networks," J. Wireless Com. Network., vol.2014, no.1, 190, 2014.
[22] H. Martikainen, I. Viering, A. Lobinger, and T. Jokela, "On the basics of conditional handover for 5G mobility," IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC, vol.2018-Septe, no.2, 2018.
[23] I.L. Da Silva, C. Eklöf, J. Muller, and R. Zhohov, "This is the key to mobility robustness in 5G networks," Ericsson.com, https://www.ericsson.com/en/blog/2020/5/the-key-to-mobility-robustness-5g-networks, accessed March 13 2024.
[24] Y. Koda, K. Yamamoto, T. Nishio, and M. Morikura, "Reinforcement learning based predictive handover for pedestrian-aware mmWave networks," INFOCOM 2018 - IEEE Conference on Computer Communications Workshops, pp.692–697, 2018.
[25] C. Wang, L. Ma, R. Li, T.S. Durrani, and H. Zhang, "Exploring trajectory prediction through machine learning methods," IEEE Access, vol.7, pp.101441–101452, 2019.
[26] R.V. Akhpashev and V.G. Drozdova, "Spatial interpolation of LTE measurements for minimization of drive tests," International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices, EDM, vol.2018-July, pp.136–138, 2018.
[27] A. Suresh Kumar, S. Vanmathi, B. Praveen Sanjay, S. Ramya Bharathi, and M. Sakthi Meena, "Handover forecasting in 5G using machine learning," International Journal of Engineering and Technology (UAE), vol.7, no.2, pp.76–79, 2018.
[28] A. Masri, T. Veijalainen, H. Martikainen, S. Mwanje, and J. Ali-Tolppa, "Machine-learning-based predictive handover," IEEE IM 2021- France, no.Im, pp.648–652, 2021.
[29] Expert.AI Team, "What is machine learning?," Expert.AI, https://www.expert.ai/blog/machine-learning-definition/, accessed March 13 2024.
[30] T. Mitchell, Machine Learning, McGraw Hill, 1997.

[31] X. Chen, F. Meriaux, and S. Valentin, "Predicting a user's next cell with supervised learning based on channel states," IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC, pp.36–40, 2013.
[32] L. Yan, H. DIng, L. Zhang, J. Liu, X. Fang, Y. Fang, M. Xiao, and X. Huang, "Machine learning-based handovers for sub-6 GHz and mmWave integrated vehicular networks," IEEE Trans. Wireless Commun., vol.18, no.10, pp.4873–4885, 2019.
[33] T. Cabral De Brito, G. Advisor, and V.A. De Sousa, "Machine learning based handover management for LTE networks with coverage holes," Ph.D. Thesis, Universidade Federal do Rio Grande do Norte, 2018.
[34] L.L. Vy, L.P. Tung, and B.S.P. Lin, "Big data and machine learning driven handover management and forecasting," 2017 IEEE Conference on Standards for Communications and Networking, CSCN 2017, no.September 2019, pp.214–219, 2017.
[35] V. Yajnanarayana, H. Ryden, and L. Hevizi, "5G handover using reinforcement learning," 2020 IEEE 3rd 5G World Forum, 5GWF 2020 - Conference Proceedings, pp.349–354, 2020.
[36] Z. Ali, N. Baldo, J. Mangues-Bafalluy, and L. Giupponi, "Simulating LTE mobility management in presence of coverage holes with ns-3," SIMUTOOLS 2015 - 8th EAI International Conference on Simulation Tools and Techniques, 2015.
[37] J. Nakazato, M. Kuchitsu, A. Pawar, S. Masuko, K. Tokugawa, K. Kubota, K. Maruta, and K. Sakaguchi, "Proof-of-concept of distributed optimization of micro-services on edge computing for beyond 5G," IEEE Vehicular Technology Conference, vol.2022-June, pp.1–6, 2022.
[38] A. Akman, "O-RAN Working Group 1 Use Cases Detailed Specification," O-RAN Alliance e.V. Technical Specification, 2020.
[39] Nokia, "Nokia launches first commercial Service Enablement Platform to drive Open RAN innovation," Nokia Press Release, https://www.nokia.com/about-us/news/releases/2021/03/11/nokia-launches-first-commercial-service-enablement-platform-to-drive-open-ran-innovation/, accessed March 13 2024.
[40] O. Orhan, V.N. Swamy, T. Tetzlaff, M. Nassar, H. Nikopour, and S. Talwar, "Connection management xAPP for O-RAN RIC: A graph neural network and reinforcement learning approach," 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), pp.936–941, 2021.
[41] Y. Li, E. Datta, J. Ding, N.B. Shroff, and X. Liu, "Can online learning increase the reliability of extreme mobility management?," 2021 IEEE/ACM 29th International Symposium on Quality of Service, IWQOS 2021, pp.0–5, 2021.
[42] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in O-RAN for data-driven NextG cellular networks," IEEE Commun. Mag., vol.59, no.10, pp.21–27, 2021.
[43] H. Lee, Y. Jang, J. Song, and H. Yeon, "O-RAN AI/ML workflow implementation of personalized network optimization via reinforcement learning," 2021 IEEE Globecom Workshops (GC Wkshps), pp.1–6, 2021.
[44] F. Cefalu, "O-RAN SC Wiki: Getting started," O-RAN SC Confluence, https://wiki.o-ran-sc.org/display/GS/Getting+Started, accessed March 13 2024.
[45] D. Karnwal, "Anomaly detection use case," O-RAN SC Confluence, https://wiki.o-ran-sc.org/display/RICP/Anomaly+Detection+Use+Case, accessed March 13 2024.
[46] B.H. Prananto, Iskandar, and A. Kurniawan, "Study on neural network for cellular mobility management: NS3 simulation with coverage hole case," Proc. 15th International Conference on Telecommunication Systems, Services, and Applications, TSSA 2021, pp.1–6, 2021.
[47] B.H. Prananto, Iskandar, and A. Kurniawan, "O-RAN intelligent application for cellular mobility management," 9th International Conference on ICT for Smart Society: Recover Together, Recover Stronger and Smarter Smartization, Governance and Collaboration, ICISS 2022 - Proceeding, pp.1–6, 2022.
[48] J. Brownlee, "Long short-term memory networks with Python: Develop sequence prediction models with deep learning," 2017.
[49] J. Faouzi, "Time series classification: A review of algorithms and implementations," Machine Learning (Emerging Trends and Applications), hal-03558165, in press.
[50] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.A. Muller, "Deep learning for time series classification: A review," Data Min. Knowl. Disc., vol.33, no.4, pp.917–963, 2019.
[51] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol.9, no.8, pp.1735–1780, 1997.
[52] F.A. Gers, D. Eck, and J. Schmidhuber, "Applying LSTM to time series predictable through time-window approaches," Artificial Neural Networks — ICANN 2001, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol.2130, pp.669–676, 2001.
[53] J.L. Elman, "Finding structure in time," Cognitive Science, vol.14, no.2, pp.179–211, 1990.
[54] NS3, "Design documentation — Model library."

**Baud Haryo Prananto** received a B.S. degree from Bandung Institute and Technology, Indonesia in 2004 and an M.S. degree from Korea Institute of Science and Technology, South Korea in 2008. Currently, he is studying at Bandung Institute of Technology as a Doctoral student. He is currently working as an Expert Technical Trainer in Nokia Solutions and Networks since 2008, delivering training related to 4G and 5G RAN Nokia equipment.

**Iskandar** received B.S. and M.S. degrees from Bandung Institute and Technology, Indonesia in 1995 and 2000 respectively, and a Doctoral degree from Waseda University Tokyo, Japan in 2007. Currently, he is an Associate Professor in the School of Electrical Engineering and Informatics, Bandung Institute of Technology with research interests mainly in wireless telecommunications.

**Hendrawan** received a B.S. degree from Bandung Institute and Technology, Indonesia in 1985 and an M.S. and Ph.D. degree from the University of Essex, UK in 1990 and 1995 respectively. Currently, he is an Associate Professor in the School of Electrical Engineering and Informatics, Bandung Institute of Technology with research interests mainly in queuing theory, data communication, multimedia communication, telecommunication network, and network management.

**Adit Kurniawan** received a B.S. degree from Bandung Institute and Technology, Indonesia in 1986, an M.S. degree from Royal Melbourne Institute of Technology, Australia in 1996, and a Ph.D. degree from the University of South Australia in 2003. He was a Professor in the School of Electrical Engineering and Informatics, Bandung Institute of Technology with research interests in antenna propagation, microwave propagation, wireless spread spectrum, and CDMA.