Radio Link Failure Prediction in 5G Networks

Karim Boutiba
EURECOM
Sophia Antipolis, France
karim.boutiba@eurecom.fr

Miloud Bagaa
CSC-IT Center for Science Ltd.
Espoo, Finland
miloud.bagaa@csc.fi

Adlen Ksentini
EURECOM
Sophia Antipolis, France
adlen.ksentini@eurecom.fr

***Abstract*—Radio Link Failure (RLF) is a challenging problem in 5G networks as it may decrease communication reliability and increases latency. This is against the objectives of 5G, particularly for the ultra-Reliable Low Latency Communications (uRLLC) traffic class. RLF can be predicted using radio measurements reported by User Equipment (UE)s, such as Reference Signal Receive Power (RSRP), Reference Signal Receive Quality (RSRQ), Channel Quality Indicator (CQI), and Power HeadRoom (PHR). However, it is very challenging to derive a closed-form model that derives RLF from these measurements. To fill this gap, we propose to use Machine Learning (ML) techniques, and specifically, a combination of Long Short Term Memory (LSTM) and Support Vector Machine (SVM), to find the correlation between these measurements and RLF. The RLF prediction model was trained with real data obtained from a 5G testbed. The validation process of the model showed an accuracy of 98% when predicting the connection status (i.e., RLF). Moreover, to illustrate the usage of the RLF prediction model, we introduced two use-cases: handover optimization and UAV trajectory adjustment.**

I. Introduction

Radio Link Failure (RLF) is an important criterion in modern wireless networks, including 5G. RLF is critical for services requiring high reliability, which is one of the key features of 5G, such as those supported by ultra-Reliable Low Latency Communications (uRLLC) class [1]. RLF impacts principally network services that involve high-mobile devices, such as vehicles and flying drones.

Indeed, combining highly mobile users, at high altitude like Unmanned Aerial Vehicle (UAV) [2], with the current limited number of deployed 5G cells and their small coverage due to wider frequency bands' usage [3] (which make the radio link between the User Equipment (UE) and the base station more susceptible to blockage and degradation leading to sudden interruption of the communication link), may yield to frequent RLF. RLF corresponds to a disconnection from the network, which strongly impacts sensitive services, such as UAV safety, where RLF increases the network latency and, in the worst cases, to UAV's losses and collisions. In this context, it is important that 5G tackles the RLF issue to improve reliability by, for instance, predicting when a mobile device has a high probability of seeing a RLF and take the appropriate action, like anticipating the Handover or updating the trajectory of the UAVs.

RLF can be estimated by measuring radio key indicators such as Reference Signal Receive Power (RSRP), Reference Signal Receive Quality (RSRQ), Channel Quality Indicator (CQI), and Power HeadRoom (PHR). By combining this information, it is possible to know the quality of the radio channel, hence the probability of RLF. However, deriving a closed-form of the RLF distribution probability is very challenging. To fill this gap, we propose in this paper to use Machine Learning (ML) to predict the RLF relying on the measurement of RSRP, RSRQ, CQI, and PHR. The output of the ML model will be used as input by third tiers applications to improve the reliability of 5G networks. Therefore, we will propose two applications that rely on the RLF prediction model: (1) an application that optimizes the handover in the context of O-RAN [4]; (2) an application that adapts UAVs trajectory to avoid RLF in the context of ETSI MEC framework [5].

The contributions of this work are manifolds:

- We propose a ML model based on LSTM to predict the future trend of radio channel measurements, taking into account the evolution of past measurements.
- We propose a novel method based on SVM to classify the connectivity status (connectivity/ no connectivity) of UEs according to radio channel measurements and hence predict RLF.
- We trained and tested the framework on a real dataset using a 5G testbed based on OpenAirInterface (OAI) platform [6].
- We present two use cases where the RLF prediction can be used. The first one is handover enforcement, and the second one is UAV trajectory modification. These use cases are aligned with O-RAN and MEC architecture, respectively.

The paper is organized as follows: Section 2 provides needed background to understand our approach and state-of-the art solutions to predict connectivity loss. Our approach is presented in Section 3 and evaluated in Section 4. We conclude the paper in Section 5.

II. Background

A. Radio Measurements

Different metrics are considered to measure the radio channel quality, among these metrics, the most used are:

Reference signal Receive Power (RSRP): defined as the linear average received power (in Watts) of the signals that carry cell-specific Reference Signals (RS) within the frequency bandwidth. RSRP provides a cell-specific signal strength metric. This measurement is mainly used for handover decisions. It enables the base station to rank different candidate cells according to their signal strength.

Reference Signal Strength Indicator (RSSI): defined as the total received power observed by the UE from all sources, including serving and non-serving cells, adjacent channel interference, and thermal noise within the measurement bandwidth. RSSI is not reported as a measurement in the 3GPP standard since [7], instead it is used to compute the RSRQ measurement.

Reference Signal Receive Quality (RSRQ): provides a cell-specific signal quality metric. Similar to RSRP, this metric is used mainly for handover and cell re-selection decisions. RSRQ enables the base station to rank different candidate cells according to their signal quality. It is used in scenarios, where RSRP measurements do not provide sufficient information to perform reliable mobility decisions. RSQR is computed as follows: $RSRQ = \frac{N*RSRP}{RSSI}$

Where N is the number of Resource Blocks (RBs) of the bandwidth, RSRP indicates the wanted signal strength, RSRQ enables the combined effect of signal strength and interference to be reported efficiently.

Channel Quality Indicator (CQI): It is an indicator carrying the information on how good/bad the communication channel quality is. It is reported by UEs to indicate to the base station the level of modulation and coding the UE could operate (i.e., means choosing the transport block size) [8], which in turn can be directly converted into throughput.

Power headroom (PHR): indicates how much transmission power is left for a UE to use in addition to the power being used by the current transmission. If the PHR value is negative, it indicates that the UE is already transmitting with greater power than it is allowed to use.

B. Related works

In [9], the authors used deep learning to reduce RLF when performing the handover process. They relied on the measured RSRP of the serving cell and the strongest neighbor cell to predict if the future handover (1-2 sec ahead) will succeed or fail. The work in [10] presented a clustering method to group cells with similar handovers' behavior to trigger early handovers without requesting the measurements of each UE. However, these works are based only on computer simulation, limiting the usability of the proposed solutions when they are deployed in real networks.

The authors in [11] proposed a deep learning approach to predict RLF according to RSRP measurements. But, the authors considered only RLF that may occur when a UE switches between a 4G and a 5G cell; hence RLF is not used to optimize network operations.

Besides, these works are focused only on handover, while other 5G use-cases may need information about RLF. Indeed, we can mention the UAV control use case, as the UAV controller needs to be notified whenever the drone may lose connectivity. This will allow for updating the drone trajectory to avoid RLF. Another use case is the smart video streaming use case, where gNB may schedule more resources for UEs to anticipate RLF by buffering more data.

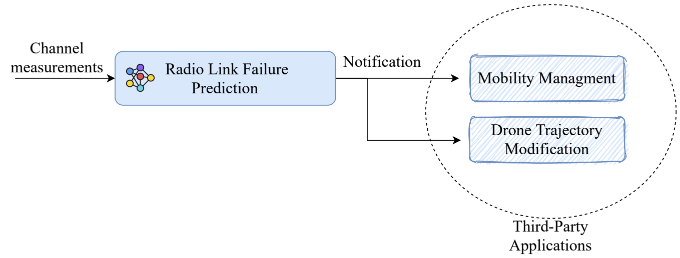

Fig. 1. Framework overview

III. Proposed framework

As indicated earlier, our main objective in this work is the early detection of radio connectivity loss and anticipation of RLF. To achieve this objective, we first conducted a campaign to measure RSRQ, RSRP, PHR, and CQI using a 5G testbed based on OAI. Second, we analyzed radio measurements and found a correlation between some measurements (RSRQ, PHR, and CQI) and connectivity status (i.e., there is connectivity or not). RSRP did not contribute explicitly to the correlation since it is implicitly used to compute the RSRQ metric (section II-A). Nevertheless, applying this correlation directly to the reported measurements leads to derive the connectivity status only when the measurements are collected. Therefore, there is a need to predict how radio measurements will evaluate in the near future. Based on this prediction, we used a classifier to find the correlation with the connectivity status (i.e., RLF).
Once RLF is predicted, third-party components can consume this information (i.e., radio connectivity loss events) for targeted UEs aiming at applying policies to avoid RLF, such as:

- Mobility Management application that takes handover decisions before link failure.
- UAV Trajectory Modification application that updates the pre-flight drone trajectory whenever the radio connectivity might be lost.

In the following sections, we will start by describing the RLF prediction model (Figure 1). We will show how this model can be used to optimize handover and update UAVs' trajectories.

A. Radio Link Failure Prediction model

1) Design: The RLF prediction model is structured into five stacked layers (Figure 2):

- **Input layer**: Composed of three-time windows; a window for each radio information (i.e., CQI, RSRQ, PHR). A time window is a vector that keeps the N last radio information collected from the RAN at the current time-step t.
- **LSTM layer**: LSTM is an improved version of RNN, designed to forecast time series data. The advantage of LSTM compared to RNNs is that the former deals with vanishing and exploding gradient problems. In RNNs, the gradient problem becomes smaller due to long data sequences. Hence, the network parameter updates become insignificant, which means no real learning is achieved.

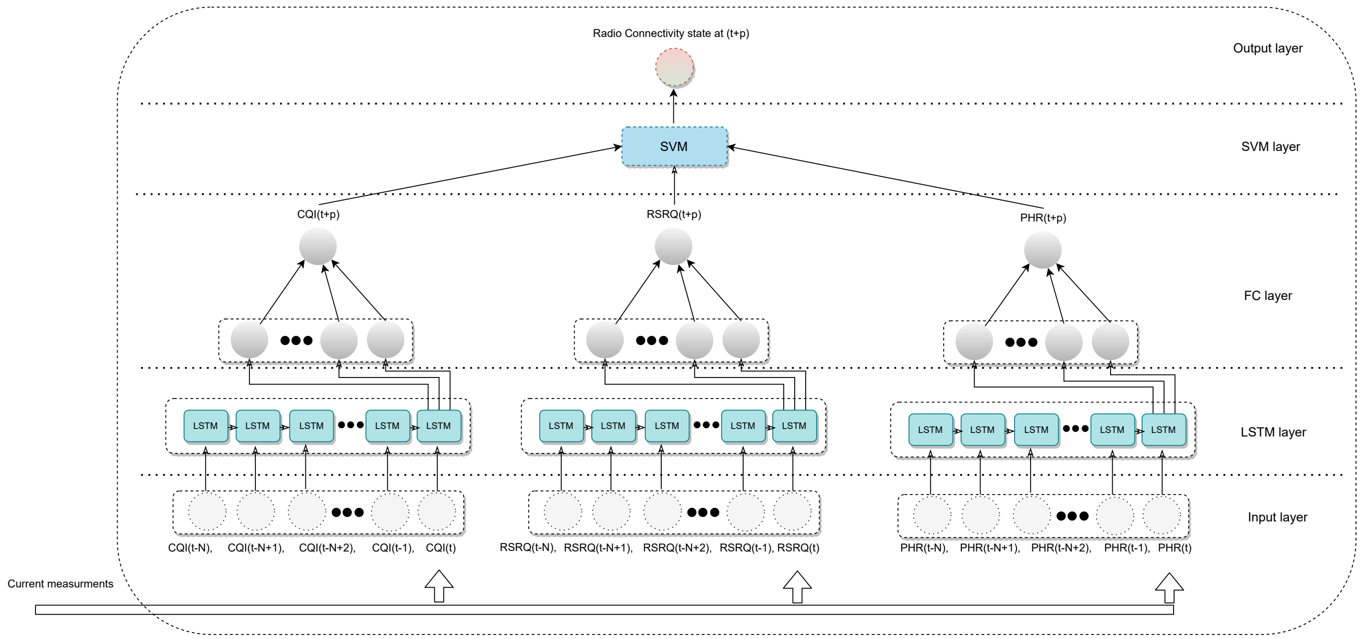

Fig. 2. RLF Prediction Model design

Besides the existing hidden state of RNNs, LSTM introduces the cell state to make the gradient computation more stable (avoid exploding or vanishing).
LSTM can learn the order dependence between items in a sequence and the context required to make predictions in time series forecasting problems.
The key function of LSTM is its ability to use a local memory (i.e., learn when it needs to memorize or forget the information) to extract temporal information from time sequences. Then, it uses this information to predict the future value at time $t+p$, $p > 0$.

- **Fully Connected layer**: This layer will sum the vector from the output of the LSTM layer to produce one predicted value.
- **SVM layer**: This layer takes the predicted values of (CQI, RSRQ, PHR) and classifies them into two classes (connectivity/ no connectivity).
- **Output layer**: This layer represents the final connectivity state at time $t+p$.

2) Data Collection: Although many recent works used deep learning algorithms to predict the channel quality, they mainly rely on simulation to generate data for the learning step. However, it is well-accepted that simulation has limitations to reproduce the behavior of real networks. In our work, we trained the RLF prediction model using data collected from a testbed based on OAI. We collected (CQI, PHR, and RSRQ) values when both the connectivity is available and non-available (i.e., RLF). We labeled the data (i.e., the tuple CQI, PHR, and RSRQ) by 1 if there is connectivity, 0 otherwise.

3) Data Pre-processing: After collecting the data and creating the data set, we cleaned the latter by removing the static values as we are more interested in the data trend over time (up/down). This manipulation made the prediction more accurate. We also needed to format the data before feeding the model. The LSTM's input data is structured into a 3D vector (samples, time-steps, features), while the SVM's input data is structured into a 2D vector (samples, features). This step is mandatory to keep the integrity between different layers and improve the overall model accuracy.

4) Training process: During this step, we trained the LSTMs and the SVM models separately using the Keras framework [12]. We used the Adam optimizer [13] to have an adaptive learning rate. It is important to note that for this kind of forecasting model, two main problems arise. The first one is over-fitting. It means that the model will learn too much about the particularities of the training data and would not be able to generalize the model to new data. In this case, the predicted signal matches exactly the real signal in the same data set, even if the latter is very complex. The second problem is that the model generates a signal that follows the real signal instead of predicting it. To overcome the first problem, we followed many techniques. First, we used two separate data sets, one for training and one for testing. Moreover, we used a Dropout layer after the LSTM hidden layer. In neural networks, some units may fix up the mistakes of other units leading to complex coordination, which, in turn, yields to overfitting as these coordinations do not generalize to unseen data.
Meanwhile, the Dropout layer randomly ignores neurons during a particular forward or backward pass. Hence, it will break up situations where network layers coordinate to correct mistakes from prior layers, making the model more robust and reducing the overfitting effect. We also used L2 regularization to push the model weights to 0 to have a less complex model, and hence less over-fitting.

To overcome the second-mentioned problem, we used the Teacher Forcing [14] method. It is a method for training RNNs that uses the output from a previous time step as an input. When the RNN is trained, it can generate a sequence using the previous output as current input, contrary to the standard method that uses real data as an input during the training phase.

B. Examples of application using the RLF prediction model

In this section, we describe how the RLF prediction model can be used via two representatives applications or services: Improving the handover using O-RAN architecture and optimizing UAVs trajectory based on ETSI MEC architecture.

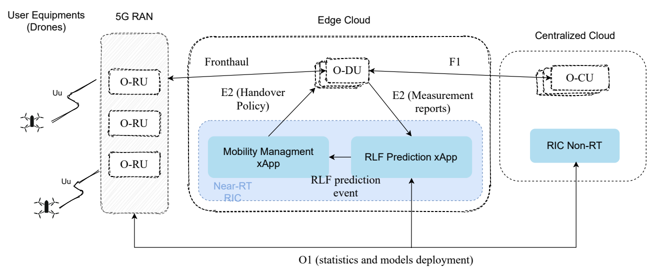

Fig. 3. RLF Prediction in O-RAN scenario

1) RLF prediction as an O-RAN xApp: According to O-RAN terminology, we assume that the RLF prediction modules and mobility management applications are run as xAPP. The mobility management xApp uses the input of the RLF xApp to optimize handover, which means, whenever an RLF is predicted, the network will trigger an early handover to avoid disconnection from the network.

The considered use-case is described in Figure 3. The RLF prediction is located at O-RAN Near Real-Time (Near-RT) RAN Intelligent Controller (RIC). It fetches the latest radio information (CQI, RSRQ, and PHR) available for a specific/set of UEs from the E2 interface. It keeps a history of past information organized as a time window for each UE and predicts connectivity loss. On the other hand, the Mobility Management service is located at O-RAN Near-RT as a xApp. It consumes RLF prediction events from the RLF prediction xApp to trigger the handover procedure whenever the connectivity is lost. It should be noted that in O-RAN, RIC Non-RT is responsible for training ML models and pushing them to RIC Near-RT modules. Therefore, we assume that RLF prediction xApp receives the trained models from the RIC Non-RT and updates the local models significantly when the inference error increases.

2) RLF prediction as a MEC application: In this use case scenario, we assume that the RLF prediction module is deployed as a MEC service available to other MEC applications. In this scenario, the MEC application is a UAV Trajectory Modification Application. The latter is deployed at the edge as it has to send commands to update the trajectory of UAVs requiring low latency communications. Figure 4 illustrates the relationship between all the components. The RLF prediction

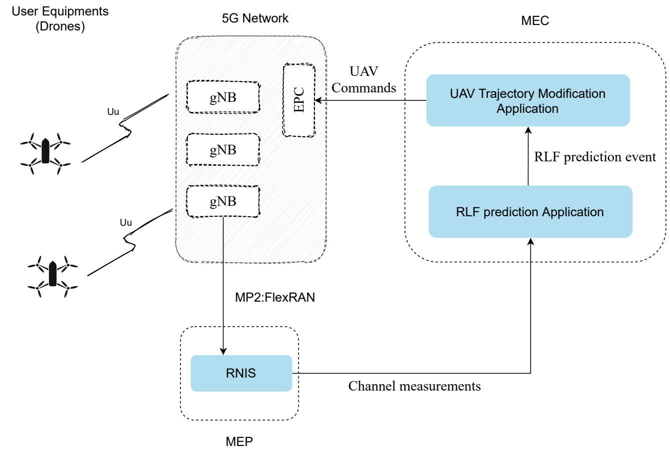

Fig. 4. RLF Prediction in MEC scenario

module collects the CQI, RSRQ, and PHR via the MEC Radio Network Information Service (RNIS) [15] API. This API is provided by the MEC Platform (MEP) through MP2 interface. To recall, the RNIS enables third-party applications to fetch radio measurements for a specific/set of UEs to accomplish data analytic or optimization jobs. The RLF prediction module uses the measurements from the RNIS and predicts whenever the connectivity will be lost for a set of UEs. If a connectivity loss is predicted, the RLF prediction application sends a notification to the UAV Trajectory modification application via the MP1 interface, informing the connectivity status of the near future of the concerned list of UEs. The UAV Trajectory Modification Application uses the prediction events from the RLF Prediction Application and modifies the UAV trajectory in order to avoid link failure.

IV. Performance Evaluation

This section is divided into two parts. The first one is dedicated to the LSTM model evaluation of CQI, RSRQ, and PHR measurements, while the second part focuses on the classification model. We compared two algorithms: Neural Networks (NN) and SVM. We tested the classifier within two cases; the first is when the algorithm gets its input directly from the test data-set, while the second one gets its input from the LSTM models.

We used a different data-set for testing to avoid over-fitting and ensure that our results are not dependant on a particular data set. Two main metrics are used to evaluate the LSTM model:

Accuracy: the ratio of the number of correct predictions to the total number of input samples.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ prediction\ made}$$

Precision: the number of correct positive results divided by the number of positive results predicted by the classifier.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Recall: the number of correct positive results divided by the number of all relevant samples (i.e., all samples that should have been identified as positive).

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

F1 score: the balance between precision and recall.

$$F1\ Score = 2 * \frac{1}{\frac{1}{Recall} + \frac{1}{Precision}}$$

It tells how precise the classifier is (how many instances it classifies correctly), as well as how robust it is.

Root Mean Square Error (RMSE): the distance between the predictions and the real observations.

$$RMSE = \sqrt{\frac{1}{T} * \sum_{t=0}^{T-1} (real_t - predicted_t)^2}$$

A. LSTM models Evaluation

We varied the LSTM parameters and tuned them to achieve the best performances: 1 LSTM hidden layer with 20 perceptrons, trained for 100 epochs with a batch size of 50 samples. We used a dropout layer with a rate of 0.2.

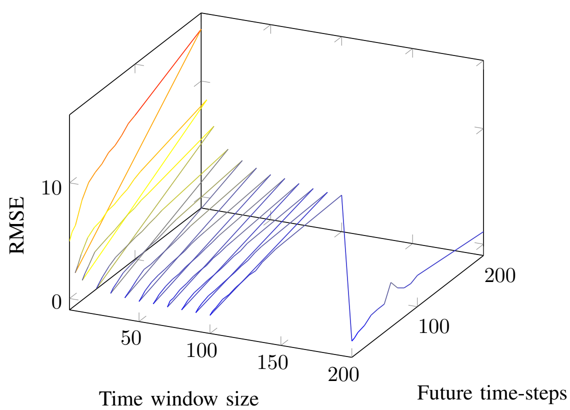

Fig. 5. RMSE variation over past and future timesteps

Figure 5 illustrates the impact of changing the time window size (how many past samples the LSTM takes as input) and the future time-step (the prediction's time-step) on the RMSE. We remark that if the LSTM does not consider enough past samples, the error is high. The model is not able to predict the data. However, when more than 20 samples are used, we observe that the error starts to converge to a smaller value, 0.1 approximately. It should be noted that LSTM is able to forget information; then, even if we continue to increase the window size, the model keeps using only the needed number of past samples to decrease the prediction error. Besides, we noticed that when the future time-step increases, the error increases. This indicates that the model is not able to predict the measurements correctly for the long-term future.

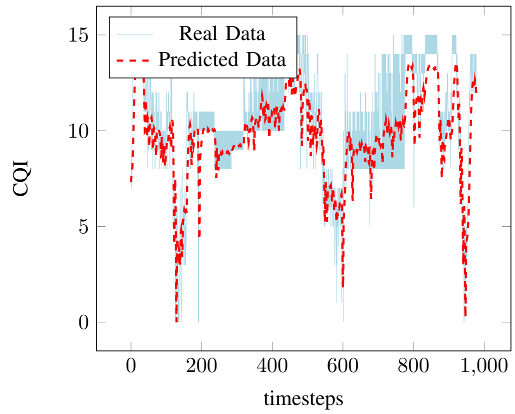

Fig. 6. CQI prediction over time

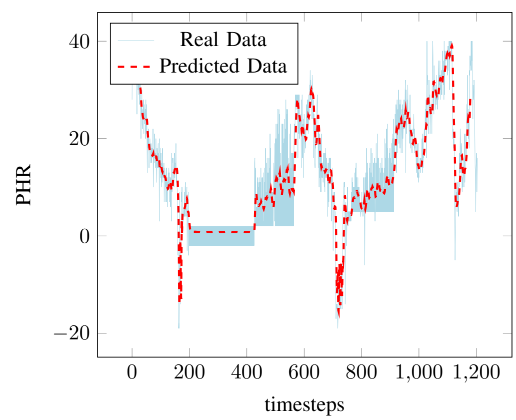

Fig. 7. PHR prediction over time

Figures 6, 7, 8 illustrate, respectively, a part of the data-set plotted with the predicted values of CQI, PHR, and RSRQ. The test data-set is shifted in time to compare the real values and the predicted ones easier. We used 50 samples as input, and we predict five time-steps in the future. The generated results show the accuracy of our LSTM model to estimate the measurement, i.e., CQI, PHR, and RSRQ.

B. Classification model Evaluation

TABLE I
CLASSIFIER MODEL METRICS

metric	SVM (%)	NN (%)
Accuracy	98,44	99,25
Precision	97,76	96,14
Recall	96,56	99,59
F1 score	97,15	97,78

For the sake of comparison, we tested two algorithms: SVM and a Neuron Network (NN) classifier. Table I compares the performance of the algorithms according to performance metrics. We notice that NN performs slightly better than SVM when the real measurements are used as input directly from the dataset. However, when we consider the LSTM's prediction

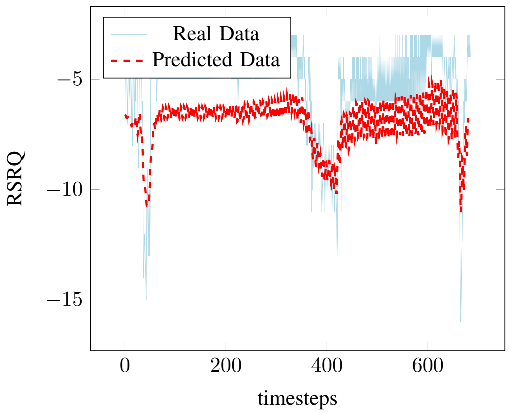

Fig. 8. RSRQ prediction over time

output as input, we can see that SVM is slightly better for the same metrics (Table II).

TABLE II
OVERALL MODEL METRICS

metric	SVM (%)	NN (%)
Accuracy	98,03	97,95
Precision	91,42	91,21
Recall	97,81	97,53
F1 score	94,32	94,08

CONCLUSION

In this paper, we addressed the challenge of predicting radio link failure (RLF) in 5G networks. To predict RLF, we used a machine learning model that combines both LSTM and SVM using the radio measurement, composed of RSRQ, CQI, and PHR. To train the RLF prediction model, we generated a data set that matches the tuple RSRQ, CQI, and PHR with connectivity state (available or not) using a 5G testbed based on OAI. The RLF prediction model was able to predict 98,03 % of connectivity status using the collected data set. Besides, we presented two use-case scenarios that exploit the RLF prediction module to optimize UE mobility (i.e., Handover Optimization) and UAV trajectory. Both scenarios were aligned with two well-accepted architectures in 5G, O-RAN, and MEC.

ACKNOWLEDGMENT

This work was partially supported by the European Union's Horizon 2020 Research and Innovation Program under the 5G!Drones project (Grant No. 857031).

REFERENCES

[1] A. Ksentini, P. A. Frangoudis, A. PC, and N. Nikaein, "Providing low latency guarantees for slicing-ready 5g systems via two-level MAC scheduling," *IEEE Netw.*, vol. 32, no. 6, pp. 116–123, 2018.

[2] B. Brik, A. Ksentini, and M. Bouaziz, "Federated learning for uavs-enabled wireless networks: Use cases, challenges, and open problems," *IEEE Access*, vol. 8, pp. 53 841–53 849, 2020.

[3] S. Vahid, R. Tafazolli, and M. Filo, *Small Cells for 5G Mobile Networks*, 05 2015, pp. 63–104.

[4] ORAN Alliance, "O-ran use cases and deployment scenarios: Towards open and smart ran," *White Paper, February 2020*, 2020.

[5] A. Ksentini and P. A. Frangoudis, "Toward slicing-enabled multi-access edge computing in 5g," *IEEE Netw.*, vol. 34, no. 2, pp. 99–105, 2020.

[6] F. Kaltenberger, G. De Souza, R. Knopp, and H. Wang, "The OpenAir-Interface 5G new radio implementation: Current status and roadmap," in *WSA 2019, 23rd ITG Workshop on Smart Antennas, Demo Session, 24-26 April 2019, Vienna, Austria*, 2019.

[7] ETSI, "Evolved universal terrestrial radio access (e-utra); physical layer - measurements," *3GPP TS 36.214 version 8.4.0*, 2010.

[8] A. Amokrane, A. Ksentini, Y. H. Aoul, and T. Taleb, "Congestion control for machine type communications," in *Proceedings of IEEE International Conference on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012*. IEEE, 2012, pp. 778–782.

[9] S. Khunteta and A. K. R. Chavva, "Deep learning based link failure mitigation," *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017.

[10] L. V. Le, L.-P. Tung, and B.-S. Lin, "Big data and machine learning driven handover management and forecasting," *2017 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2017.

[11] S. M. Asad Zaidi, M. Manalastas, A. Abu-Dayya, and A. Imran, "Ai-assisted rlf avoidance for smart en-dc activation," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020.

[12] F. Chollet. (2021) Keras: the python deep learning api. [Online]. Available: https://keras.io/

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[14] A. Lamb, A. Goyal, Y. Zhang, S. Zhang, A. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," *NIPS 2016*, 2016.

[15] S. Arora, P. A. Frangoudis, and A. Ksentini, "Exposing radio network information in a mec-in-nfv environment: the rnisaas concept," in *5th IEEE Conference on Network Softwarization, NetSoft 2019, Paris, France, June 24-28, 2019*. IEEE, 2019, pp. 306–310.