CS230

Deep learning approach for optimizing mobile handovers and reducing measurement report in LTE/5G Self-Organizing Networks

Sourav Bandyopadhyay
AI Graduate student
souravb@stanford.edu

Abstract

Self-organizing networks are used in a wireless network system to assign channels and other radio parameters, monitor coverage holes, improve mobile mobility. The motivation of this project is to explore ways to make mobility handovers quicker so that the mobile can save on battery for cell searching and help in assigning the best radio channel for a mobile to maintain a good quality of service in a private network for mobile robots that are continuously moving and needs to remain connected to a network

1 Introduction

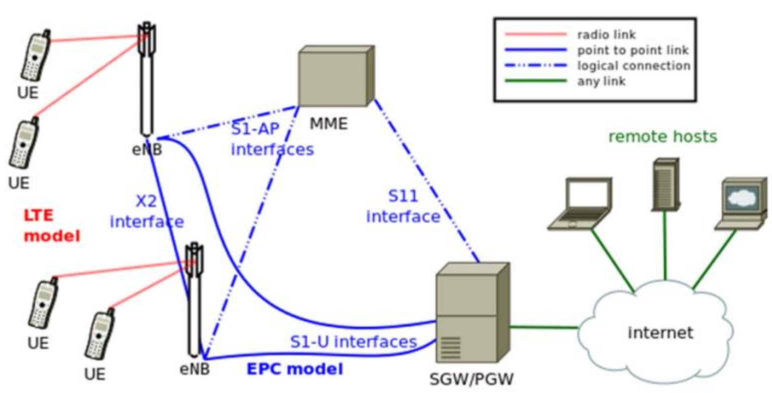


Figure 1: Example of NS3 mobility simulator for LTE networks, Adapted from [1]

The current Self-Organizing networks are mostly rule based in the industry, and a lot of the network needs manual interruption to setup and tune based on the performance observed, this caused a very slow feedback loop in the network and network performing at a suboptimal level thus not meeting the Service Level Agreements (SLA's) and Quality of performance. To allow the network make decisions in an optimal manner and be able to take actions ahead of time this project defines a method to predict if there is a scenario where a mobile handover can be triggered based on mobile parameters reported and try to use the node base (eNB) interfaces to share the context of the mobile preemptively.

2 Related work

There are several approaches that has been researched when it comes to handover and mobility optimization and load balancing mobiles a network intelligently, a basic example in a dense network can be each eNB load balances based on the Σ (mobiles) attached. In a handover scenario most of the handover triggers are threshold based, that means the handover mechanism gets triggered when certain thresholds are met, although most of these thresholds have hysteresis to avoid hard switching and certain rule-based trigger criteria's need to be met for a mobile to try to switch the network, these methods are suboptimal and can cause the mobile to camp in to an eNB that may cause reduction in QoS.

There are research work happening on handover optimization and making radio-performance measurement and fault management more preemptive, some of the methodologies I have seen are taking the temporal characteristics of the data into account as most of the data are sequential and can be thought of start transitions. Other methodologies that I have read are using reinforcement learning techniques like Deep Q-learning to predict the next state of a cell in a network, other methods are mostly trying to learn the states of the network using LSTM/RNN to predict certain parameter. Essentially all these techniques are still in research phase, and some may be useful in a macro network, some may be good for an indoor warehouse, robots, and drone control scenarios. In this project i am trying to define a deep-learning method which based on the signals capture from the mobile predicts if there will be a handover or not and that the probable handover candidate for that mobile.

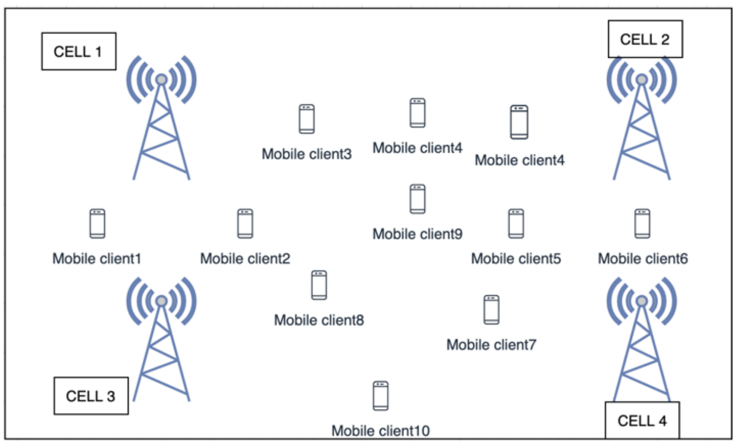
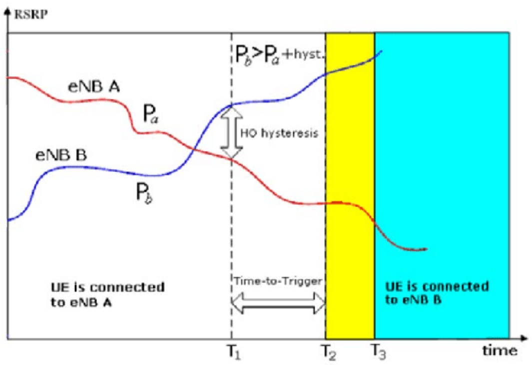


Figure 2: Simulation scenario being used to generate indoor propagation data using RandomWalk2dMobilityModel

3 Dataset and Features

The data collected is over NS3 simulator, in this project I have built a realistic indoor propagation scenario with 100 mobiles connected to a LTE network and have 21 eNBs Figure 2 showed the simulation scenario on NS3, the relevant data is where the QoS is decreasing for a network, each mobile reports the RSRP which the power level at which it sees the eNB it is attached to, the SINR which is the ratio if signal to (noise + interference) and the connected cell at that time. Table 1 shows a sample form of data being used for this project, the CELL_ID is a unique identifier to identify an eNB, in this example the UE moves from eNB with CELL_ID 7 to CELL_ID 18 and then again to a CELL_ID 16, as we track this movement it can be seen that the range of previous RSRP values that trigger a handover, and this can be sequential time series data based on NS3 RandomWalk2dMobilityModel and NS3 indoor rf propagation model to simulate walk on every direction. I have performed data cleaning to make sure if a mobile is static in a same location and there is no change in RSRP or SINR then that data need not be used as all the data points from those scenarios can be averaged into one data point to take that state into account. The data set is broken down into train and validation sets, 98% is used for training and 2% is used for validation.



Time	SINR	RSRP	CELL_ID
46.25	-145.95	0.95	7
47.25	-146.1	0.99	7
48.25	-144.93	0.87	18
49.25	-144.77	0.9	18
50.25	-111.1	57.84	18
51.25	-111.95	59.59	18
52.25	-113.33	59.33	18
53.25	-123.23	7.11	18
54.25	-129.92	0.06	18
55.25	-121.26	3	16
56.25	-122.72	3.03	16
57.25	-120.41	2.18	16

Figure 3: Time to trigger region where the data is collected at Source: Adapted from [2]

Table 1 Shows the handover of a UE from one cell to another based on the RSRP and SINR measurements

4 Methods

The motivation of this project is this project is to find a deep learning approach to make this process automated and help in tuning the network dynamically rather than rule based, this has a two-prong benefit, this will help networks to meet much higher channel quality as the network learns from handover failure areas and coverage gaps, so the handover thresholds (A1,A2,..,A5) can be optimized and reference signal power of an operating network can be too, so make sure the maximize the coverage. Thus, will also have a cost benefit for networks as this will help design networks with lower number of radios.

In this project I have used a many to one LSTM framework with dropout and batch normalization to find out the temporal characteristics of the data and decide if a handover is about to happen. I am using a multi-layer many-to-one LSTM architecture.

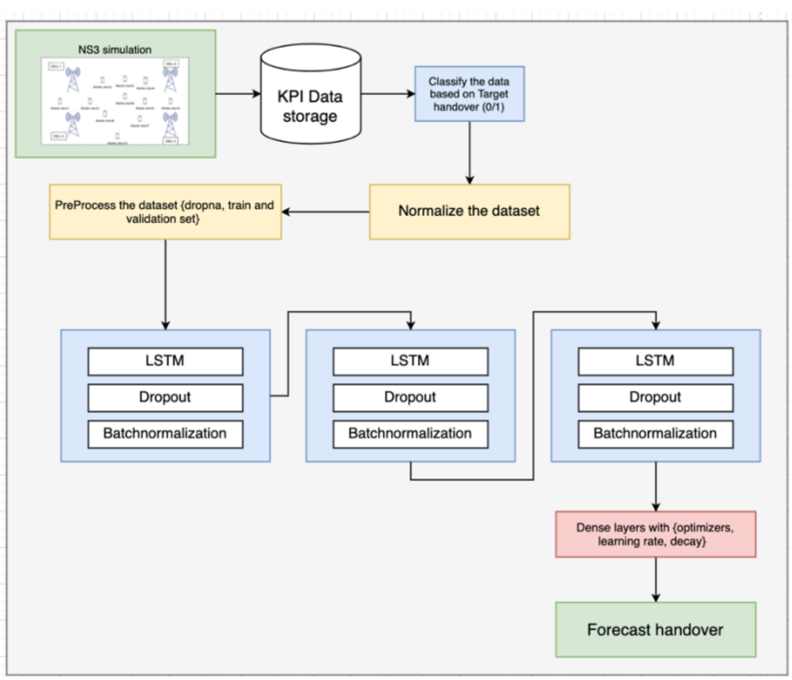


Figure 4: Deep learning framework used in the project

5 Experiments/Results/Discussion

I implemented of the proposed models in Python, using Keras and Tensorflow, as backend. To speed up the training Nvidia CUDA Deep Neural Network (CuDNN) library for GPUs was used inside Nvidia GPU docker container for tensorflow. To converge at the hyperparameters of the final model, number of layers and the number of LSTM units in each layer, i tested five different combinations. Finally, the hyperparameters resulted in a lowest average Mean Square Error (MSE) (over 400 epochs) were selected. I have trined for 50,100,150,200,250,400,500 epochs, after 250 epochs I have seen decreased of the data, I observe that, after 200 epochs, this model can achieve and maintain very high validation accuracy independently from the number of layers and cells.

The performance evaluation of these models is performed in an offline fashion, i.e., by comparing the real time to download for each UE, obtained after selecting the target cell providing the lowest predicted time to download, to the one achieved by using a benchmark approach.

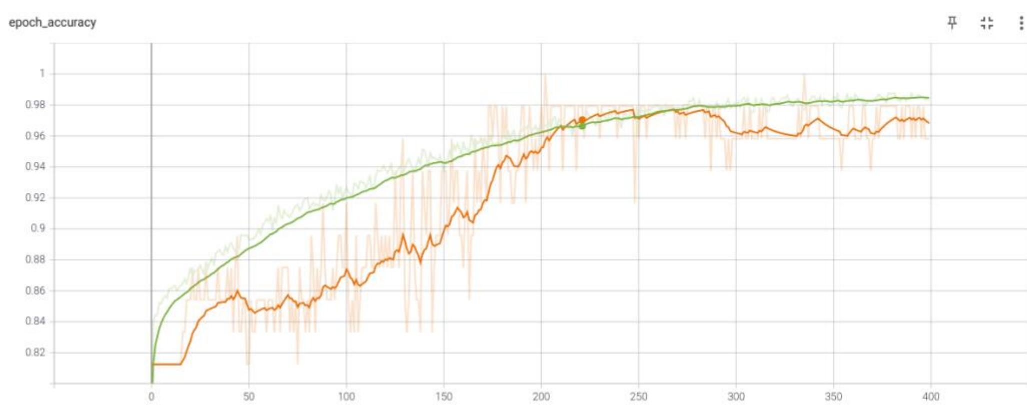


Figure 5: Epoch accuracy for model training (green) validation (orange) y_axis = accuracy, x_axis = num_epoch

6 Conclusion/Future Work

I want to take this work further and train the model with more real-world data with, a part of this implementation is going to be productized in the upcoming months and be used in handover validation and optimization for dense indoor networks.

7 Contributions

This project was done alone by only one team member.

Reference:

[1] nsnam.org/docs/models/html/lte-design.html

[2] https://www.researchgate.net/figure/LTE-Handover-Process_fig3_283734681

[3]https://ieeexplore.ieee.org/abstract/document/8954892?casa_token=jCWAC-lMiIoAAAAA:yvSAwD0Mw-NuITLAYktBvJDBpFKiOifk3OI6kqgAt44PONabK7dvW_dRGbRQguU5WMqLbLCfb9s

[4] https://ieeexplore.ieee.org/document/5680001

[5] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8301842

[6] https://arxiv.org/pdf/1707.02329.pdf

[7] https://www.mdpi.com/2076-3417/9/15/2975/htm

[8] https://www.nsnam.org/docs/models/html/lte-user.html

[9] https://www.nsnam.org/tutorials/consortium14/ns-3-training-session-6.pdf