Where to Go Next? : A Realistic Evaluation of AI-Assisted Mobility Predictors for HetNets

Marvin Manalastas, Hasan Farooq, Syed Muhammad Asad Zaidi and Ali Imran
University of Oklahoma, Tulsa, USA 74135
Email: {marvin, hasan.farooq, asad, ali.imran}@ou.edu

***Abstract*—5G is considered as the ecosystem to abet the ever growing number of mobile devices and users requiring an unprecedented amount of data and highly demanding Quality of Experience (QoE). To accommodate these demands, 5G requires extreme densification of base station deployment, which will result in a network that requires overwhelming efforts to maintain and manage. User mobility prediction in wireless communications can be exploited to overcome these foregoing challenges. Knowledge of where users will go next enables cellular networks to improve handover management. In addition, it allows networks to engage in advanced resource allocation and reservation, cell load prediction and proactive energy saving. However, anticipating the movement of humans is, in itself, a challenge due to the lack of realistic mobility models and insufficiencies of cellular system models in capturing a real network dynamics. In this paper, we have evaluated Artificial Intelligence (AI)-assisted mobility predictors. We model mobility prediction as a multi-class classification problem to predict the future base station association of the mobile users using Extreme Gradient Boosting Trees (XGBoost) and Deep Neural Networks (DNN). Using a realistic mobility model and a 3GPP-compliant cellular network simulator, results show that, XGBoost outperforms DNN with prediction accuracy reaching up to 95% in a heterogeneous network (HetNet) scenario with shadowing varied from 0dB to 4dB.**

***Index Terms*—Mobility prediction, AI, self- organizing networks (SON), Deep Neural Networks, XGBoost, HetNets.**

I. Introduction

The volume of capacity-hungry devices is expected to rise exponentially with the increase in bandwidth demand from the end-users. To cope with the anticipated challenge of providing a 1000x increase in capacity, network densification has emerged as the primary method future networks will rely on. The deployment of heterogeneous types of base stations (BS) such as macro cells and smalls cells will be necessary to alleviate the issue. However, as dense deployments of heterogeneous types of cells become the norm, intricacy in managing the network mounts. This complexity will affect all aspects of cellular network management from resource allocation, mobility, and energy saving to the associated increase in mobile operators capital expenditures and operational expenditures (CAPEX/OPEX) .

To address these challenges, Self-Organizing Networks (SON) have arisen as the go-to solution. With SON, manually executed operations, such as network configuration, optimization, and maintenance, can now be automated. However, current SON solutions are reactive in nature - that is SON functionalities will only intervene once a problem has occurred. This characteristic contradicts with 5G's ambitious quality of service which requires a proactive mode of operation for SON functions. Achieving this proactivity is possible by anticipating the movements of users and forecasting the future network state using information readily available in the network which referred to as Big Data. Given these proactive predictive capabilities, a more effective and efficient method of network resource allocation can be put in place [1].

To predict where a user will go next is one of the key components of a proactive SON. This can be done by forecasting the future locations of the users in terms of the associated base stations. This mobility prediction relies on the notion that activities and movements of mobile users are predictable to a certain extent as verified by the work [2]. According to this study, user's daily mobility has a regularity with a 93% average predictability despite the randomness of individual trajectories.

The majority of the current studies in mobility prediction leverage analytical-based techniques, particularly Markov chain. Its popularity can be attributed to minimal space/time complexity relative to other techniques. One particularly promising work [3] harnesses semi-Markov model capabilities for spatiotemporal mobility prediction in cellular networks. A maximum prediction accuracy of 90% is achieved in the experimental evaluation utilizing actual network traces. However, considering the computational resources available in the present-day, AI-based mobility predictors are viable alternatives. Works [4] and [5] exploit Machine Learning (ML) for classification of the spatial trajectories through supervised learning using Support Vector Machines (SVMs). Results show that accuracy of the predictions reach more than 90% using regular mobility movements. Using deep learning, authors in [6] propose a mobility prediction in mobile ad-hoc networks. The best performance of their mobility predictor shows a mean square error (MSE) of 5.29e-08 in the validation set.

Although results are auspicious, the suitability for practical applications is inadequate for two reasons. First, [4], [5] and, [6] all use unrealistic mobility models. [6] uses the random waypoint model (RWP) which clearly fails to capture the mentioned degree of regularity in human movement. Though [4] and [5] incorporate models better than RWP, they design the users to roam around a cellular network following no realistic route. In addition, [4] and [5] both adopt unrealistic cellular system models. In [4], the cellular network used is composed only of omnidirectional base stations, while in [5] base stations are represented by irregular polygons, both insufficiently reflect a modern cellular network setup.

This paper will overcome these limitations by leveraging the core idea of [3] using AI instead of semi-Markov model. Moreover, instead of using unrealistic human movement patterns, realistic traces are generated using a traffic simulator named Simulation of Urban Mobility (SUMO). Combined with these realistic mobility patterns, is a realistic cellular network system created using a Python-based cellular network simulator called AI4Networks Simulator.

The main contributions of this paper can be summarized as follows:

1. A novel set of input feature combination composed of base station camping history, current cell association and sojourn time which refers to the period a user stayed in one cell. Additionally, handover locations are also used to further improve the prediction accuracy.

2. Current papers on mobility prediction use unrealistic mobility patterns of human movements. In this paper, mobility model used, though synthetic, captures a realistic movement of users. In addition, a real road topology is also used for simulation.

3. The cellular network setup is also realistic using a HetNet scenario with base station association based on received signal strength and incorporates shadowing. Current papers on mobility prediction use only one type of base station and do not consider the effect of shadowing.

The rest of the paper is organized as follows: Section II describes the Mobility prediction model; Results are illustrated and evaluated in Section III; and Section IV concludes the paper.

II. REALISTIC MOBILITY PREDICTION FRAMEWORK

The suitability and practical application of the results from this paper to a real-world scenario are achieved in three ways: 1) by using a synthetic but highly realistic mobility pattern, 2) by incorporating a realistic cellular network scenario which captures events like handover and incorporates shadowing and 3) ease in modeling the mobility predictor leveraging the power of AI. The process of future base station association prediction is shown in Figure 1. This part of the paper will discuss the user mobility model and cellular model as well as the applied AI techniques used to capture mobile user mobility.

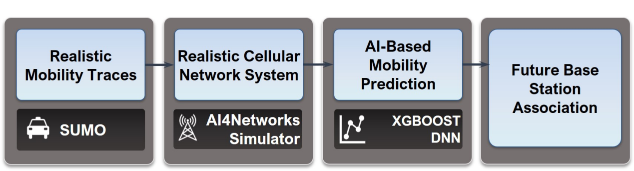

Figure 1: Realistic AI- Assisted Mobility prediction framework

A. *Realistic User Mobility Model*

SUMO is a free, open source traffic simulator which supports network importing and demand modeling [7]. Other traffic simulators are available, but SUMO's ability to create a realistic mobility model using activity-based traffic scenarios sets it apart. Location traces extracted from SUMO exhibit high dynamic characteristics that are intrinsic to fashion a realistic mobility prediction.

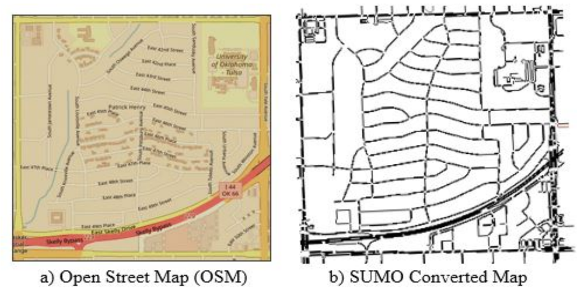

Figure 2: Creating Realistic Road Map from SUMO

To generate mobility traces, SUMO needs two mandatory inputs, a network file and a population definition file. A network file describes roads and intersections where the simulated vehicles will travel. Road topology can either be created manually or automatically by converting an open source map (OSM) network into a SUMO network as shown in Figure 2. The second input needed contains a description of the population inside the network. This general statistical information includes the number of households, the locations of houses, schools and workplaces, free time activity rate, etc. By default, population activities involve travel from home to work or school. However, additional trips can be generated using the 'free time activity rate' attribute. This attribute corresponds to the probability that in a given day, a household will have free time activity. Values can be set from 0 to 1. The higher the value, the more likely that the population will perform free time activities which can be considered as a proxy for increasing randomness in trajectories.

Populations inside the network are set to follow a shortest-path, also called optimal path, model. This means that the simulated vehicles will traverse the route that will take them to the destination the fastest. However, to add randomness, the option exists to assign weights randomly by a factor μ. By doing so, edge weights for routing are dynamically distorted in a random manner. The degree of randomness will depend on the value drawn uniformly from $[1, \mu]$. Every time an edge weight is determined, the randomization is performed so that a vehicle could select a diverse path. This randomness is a good way to simulate the use of alternative routes.

B. *Realistic Cellular Network Model*

AI4Networks Simulator is a cellular network simulator built in Python for 5G and beyond networks in compliance with 3GPP Release 15 [8]. It is a modular, flexible and versatile simulator supporting advanced features like adaptive numerology, handover and futuristic database-aided edge computing to name a few.

A representative cellular network system, as shown in Figure 3, can be created by defining the base station parameters in the site information, such as location, type (macro or small cell), height and transmit power. Aside from that, to capture the dynamics of a real network, shadowing can be incorporated in the site information. AI4Networks Simulator supports mobility models, such as random waypoint, SLAW, and Manhattan, however, traces from external sources like SUMO can easily be converted into a simulator readable format before importing for simulation. Using AI4Networks Simulator, mobility traces from SUMO can be converted into a realistic base station association that is used for mobility prediction.

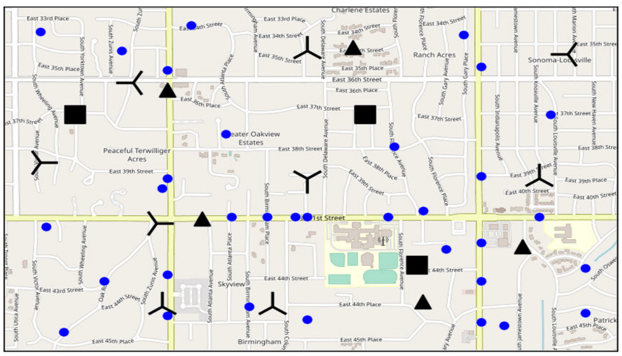

Figure 3: Sample heterogeneous network layout with sectorized BSs, omnidirectional BSs (square), small cells (triangle) and UEs (dots).

C. AI-Based Classification Techniques

In this paper, future base station association is modeled as a multi-class classification problem. With classification, labels of the data points are predicted by mapping input features (X) to discrete labels (y). In our study, the user's trajectory is converted into corresponding cells it camps on during its course. Input features (X) include the history of cell camping, sojourn time (the period a user stays on a cell), and handover location while next BS cell IDs are the discrete classes (y). Two techniques are evaluated to predict the user's next cell association. One is Machine Learning (ML) based called Extreme Gradient Boosting (XGBoost) and the other falls under the Deep Learning (DL) family known as Deep Neural Networks (DNN). These two techniques are chosen due to their promising prediction accuracy as verified by the work in [9].

a) Extreme Gradient Boosting (XGBoost): XGBoost is a popular type of gradient boosting algorithm which belongs to a ML category known as ensemble learning. Techniques under this category train several learners to perform the same task. In XGBoost, multiple regression trees, called weak learners, are trained and then converted into a single superior learner, which is the combination of the decision results of all the weak learners. Mathematically this can be expressed as [10]:

$$\hat{y}_i = \sum_{k=1}^{K} f_k x_i, f_k \in F \tag{1}$$

where F is the set of all possible weak learner and K is the total number of weak learners.

XGBoost searches for the optimal parameters by minimizing the loss function given by:

$$L = \sum_i l(\hat{y}_j, y_i) + \sum_k \Omega(f_k) \tag{2}$$

$$where \quad \Omega(f_k) = \gamma T + \frac{1}{2}\lambda\|\omega\|^2 \tag{3}$$

where l is the loss function that calculates the difference between the target value y_i and the predicted value $\hat{y}_j$, Ω is the regularized term which measures the complexity of the model, T is the total number of leaf nodes with ω representing the weight of the leaf nodes, γ is the learning rate, and λ is the constant coefficient controlling the degree of regularization of f_k.

Training the ensemble model in an additive manner is more efficient and avoids the complexities of using traditional methods in Euclidean space, therefore, we will need to add f_t in order to optimize which forms a new loss function. At time step t:

$$L(\Theta)^{(t)} = \sum_{i=l}^{n} l(y_i, \hat{y}^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{4}$$

$$L(\Theta)^{(t)} = \sum_{i=l}^{n} [g_i f_t(x_i + \frac{1}{2} h_i f_t^2(x_i)] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2 \tag{5}$$

where $y_L^{(t)}$ represents the prediction of the i-th instance at the t-th iteration, while the first and second order gradient statistics on the loss function are given as g_i and h_i respectively.

In this paper, we used grid-search to find the optimal performance of XGBoost. Based on the results, best prediction accuracy is observed using the minimum child weight value of 5, maximum depth of 3, column sample of 0.8, step size of 50 and shrinkage value of 0.01. Shrinkage η supervises the learning rate and controls over fitting. Other parameters used are set to their default values.

b) Deep Neural Networks (DNN): Deep Neural Networks belong to the family of Artificial Neural Networks (ANN) composed of multiple hidden layers between input and output layers. In this paper, we have implemented a feed-forward deep neural network to predict the next base station mobile users will enter. In a feed-forward network, the flow of information is unidirectional. This means information flow is from input going to the hidden layers and finally to the output layer without any loop and not forming any cycle.

Performance of DNN depends on the choice of activation functions. In our model, we have used the Rectified Linear

Unit (ReLU) activation function on the input and hidden layers and softmax function on the output layer. One particular advantage of ReLU is that it speeds up the training of the neural network by rapidly accelerating the convergence of stochastic gradient descent compared to the other functions like sigmoid and tanh. Mathematically, ReLU is represented as:

$$f(x) = max(0, x) \tag{6}$$

where $f(x)$ is the activation, x is the input data, and the function $max(0, x)$ is a non-linearity that is applied element-wise. Simply say, if $x < 0$, $f(x)$ = 0 and if $x >= 0$, $f(x)$ = x.

On the other hand, softmax is typically the preferred activation function for the output layer especially for classification tasks. For multi-class classifications, decimal probabilities are assigned to each class using softmax. The sum of all decimal probabilities should be equal to one which is not the case in other activation functions such as sigmoid function. This additional restriction helps in a more rapid training convergence period. Mathematically, the softmax function is given by:

$$softmax(z) = \sigma(z)_j = \frac{e_j^z}{\sum_{k-1}^{K}} j = 1, ..., K \tag{7}$$

where input z is defined as:

$$z = w_0x_0 + w_1x_1 + ... + w_mx_m = \sum_{i=0}^{m} w_ix_i = w^Tx \tag{8}$$

where x is the feature vector of a single training sample, w is the weight vector, and w_0 is the bias unit.

For mobility prediction, a DNN with a depth of 4 hidden layers and a width of 60 neurons each is used. Several other combinations of depth and width are evaluated but none exceeds the performance of the final model. The size of the output layer depends on the number of the output classes. Predictive performances of the models are evaluated using k-fold cross-validation with k = 4 and n = 2. This helps in judging how the trained model will perform when feed with unseen data. Models are run with a batch size of 32 and 300 epochs for best test accuracy rate.

D. Mobility Prediction Model

The first step in our mobility prediction is to generate realistic mobility traces. To do so, SUMO is utilized. A population is initially configured with activities involving only trips from home to work and vice versa using the shortest path. This is done by configuring a population all made up of working adults, setting the unemployment rate to 0 and setting car preference equal to 1. With this setting, all individuals in the population will go to work using their cars. This set up is the baseline scenario, tagged as Scenario 1, which reflects the regular human mobility in deterministic trajectories. In this scenario, no randomness is involved in terms of the population's activity or routes traversed. In addition, two more scenarios are created. In Scenario 2, we added an additional trip each day aside from the regular home to work routine by setting the free time activity rate equal to 1. The destinations of these additional trips are chosen randomly. This captures the randomness in real human movement accounting for other activities outside their routines such as shopping, visiting friends or leisure. Thus, we describe Scenario 2 as having medium randomness. However, the route from home to work still follows a shortest-path model. To capture the randomness on the path a real human might take outside its regular route, for example taking a detour due to traffic congestion or road construction, Scenario 3 is created. Here, a factor of 10 is used to update the weights of each edge every time the vehicle passes by. With this, high randomness is expected for Scenario 3.

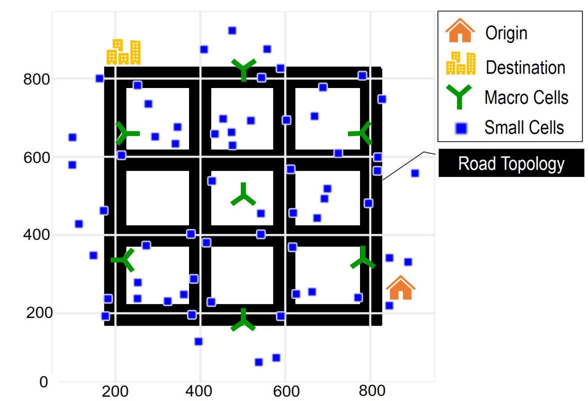

Figure 4: System Model of the Mobility Prediction.

A network or road map, 800m x 800m in size, is initially used. This grid-type map which the vehicle can traverse is our baseline network. An example of one user's origin and destination location is illustrated in Figure 4. This base network is used for simplified extraction of traces and model training. However, realistic roads are not always grid-like. For this reason, we have also used a real map of size 1.6km x 1.6km from Figure 2 including the University of Oklahoma - Tulsa campus. This setting will test how effective our models will perform in a real road network.

SUMO is run initially to generate the populations mobility traces equivalent to 10 days with one-second granularity. In addition, to test the effect of increasing the size of input data, we run simulations which provide the equivalent of 30-days and 60-days of population movement.

Using AI4Networks Simulator, two sets of cellular network models are created. The first is composed of 7 omnidirectional macro cells and the second set is a HetNet consisting of 7 macro cells, each with 3 sectors and 3 uniformly distributed small cells per sector. This results in a total of 85 base stations, 21 macro cells and 64 small cells as shown in the system model in Figure 4. The first setup is used to identify the best input feature combination and determine additional input features to help further improve the prediction accuracy. The second setup is used to test the mobility prediction model in a more realistic cellular system. To capture the dynamics of a real cellular network, values of shadowing is varied using values of 0dB, 2dB and 4dB.

Mobility traces from SUMO are fed into the AI4Networks Simulator to get the base station association at every time step. The output cell association of the AI4Networks Simulator is then processed and is used for mobility prediction. Approximately 90% of the data are used for the training set, and the remaining 10% are utilized to test the prediction accuracy. The training data set is further split into two for training (75% of 90%) and validation (25% of 90%) by using four-fold cross-validation. The cross-validation uses one-fold for the testing set and the union of the rest of the folds for the training set. For both prediction techniques, we have simulated different input features combinations of current location, sojourn time and previous cells to determine the best one. Then, handover location is incorporated to determine if it will increase the prediction accuracy before finally testing the models in a HetNet setup and using a real road network.

III. Results and Discussion

Using the base setup composed of 7 omnidirectional macro cells with no randomness in the population's mobility, Scenario 1, we identified the best input feature combination that will yield the highest prediction accuracy. As seen from the results in Figure 5, using the current cell alone to predict the next cell will not produce desirable results. Adding sojourn time as an input feature will result in a prediction of higher accuracy. However, knowledge of the previous cell the mobile user camped provides the biggest leap in accuracy. Adding more previously visited cells does not affect the prediction accuracy substantially. Based on the results, one previous cell, the current cell and the sojourn time are the best input features combination in predicting future cell association. It is notable that the performance of the prediction models is greatly affected by shadowing. From more than 98% accuracy in 0dB shadowing, it drops to 88% and 82% if 2dB and 4dB of shadowing are incorporated respectively. It is also apparent that performance of the two classification algorithms are almost equal once we added previous cell/cells as an input feature.

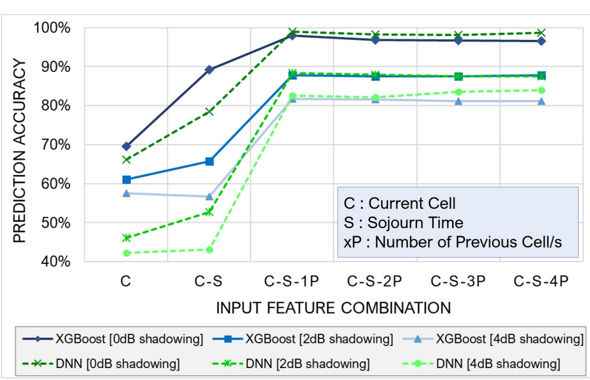

Figure 5: Determination of best input feature combination.

A study [11] uses location coordinates as one of the input features to model taxi drivers' behavior to predict their future destinations. Inspired by the usefulness of this idea, we have added the handover location, expressed as coordinates, as an input feature to determine its effect. Using Scenario 1 and the best input features identified, the effect of incorporating handover location on prediction accuracy is evaluated. Based on the results in Figure 6, a 2% to 4% increase in the accuracy can be achieved with 2dB and 4dB shadowing by adding handover location. Accuracy with 0dB shadowing is already very high but even higher results are obtained.

Best input features plus handover location are the input features used in mobility prediction in a HetNet cellular network. With a HetNet, more base stations are involved resulting to more categories to choose from, and thus it is more challenging to predict the next base station. Figure 7 shows the mobility prediction accuracy results of the two algorithms in 3 different randomness scenarios. Similar to the previous results, the effect of shadowing in the accuracy is pertinent. As randomness in trips made is added (scenario 2), accuracy of prediction decreases to 58.54%-84.92% and further reduces to 54.07%-80.07% when randomness in route used is incorporated (scenario 3). However, the most interesting result is XGBoost outperforming DNN for all shadowing values and all scenarios simulated. This observation conforms with results in [9] where the authors conclude that XGBoost is the current technique of choice for mobility prediction. Also from [9], authors conclude that DNN's performance stabilizes by adding more input features. In our case, as the number of classes increases, i.e., a HetNet with 84 classes, DNN will require more input features to perform better.

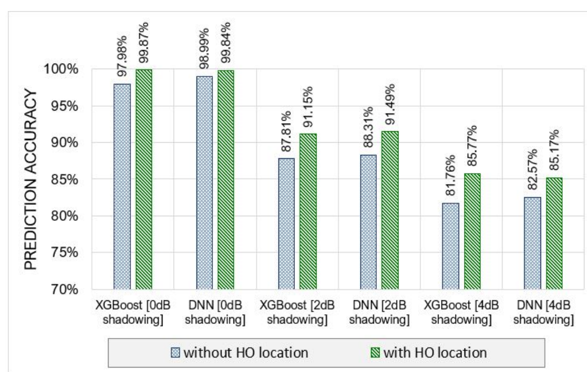

Figure 6: Effect of incorporating HO location in the mobility prediction.

The effect of increasing the training data can be reflected in the results shown in Figure 8a. Results of a 10-day simulation with 4dB shadowing are compared with the results of 30-days and 60-days of mobility traces. Results show that increasing the training data will increase the prediction accuracy. Improvement is greater in XGBoost as neural networks tend to require more input features with the increase in the input data.

Figure 8b shows the result of prediction accuracy using a real road topology. Using Scenario 3 with high randomness,

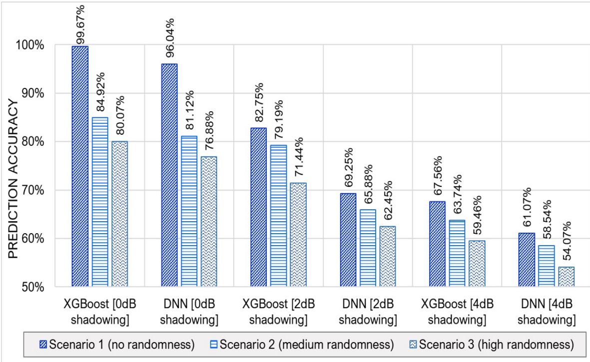

Figure 7: Mobility prediction model performance in HetNets.

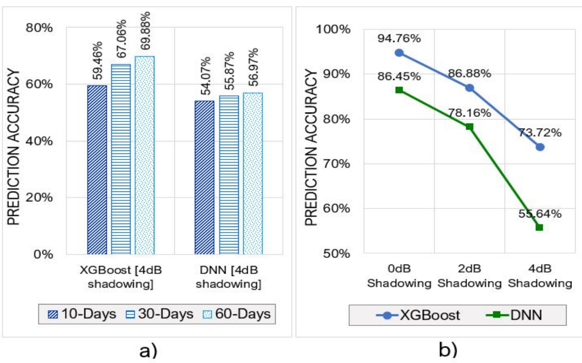

Figure 8: a) Effect of increasing the data set b) Prediction performance in a real road network

and mobility traces run for 60 days, XGBoost performs better than DNN. It can also be noticed that 73.72% accuracy can be achieved with 4dB shadowing, higher than 69.88% using a synthetic grid-type map. This result is because the shortest-path from origin to destination using the realistic road map has fewer edges. This means a fewer number of turns is required than in the grid type map making it more predictable.

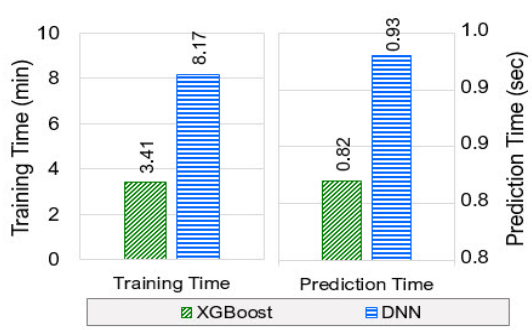

Figure 9: Training and Prediction Time Comparison

Comparison of the training and prediction time of the two algorithms shown in Figure 9 shows that XGBoost is faster in both training and prediction duration. This happens as training a DNN needs a repeated scanning of the entire training data set before reaching the asymptote.

IV. Conclusion

In this paper, we evaluated two mobility prediction models leveraging the power of AI in the form of XGBoost and DNN. Experimental results show that using one previously camped-on BS, current BS association and sojourn time as input features yield the best prediction accuracy. Moreover, incorporating the handover location and increasing umber of training sample further improve the performance of the models. Comparison of the performances of the two algorithms show that XGBoost and DNN perform similarly when using a smaller number of base stations, however, XGBoost triumphs against DNN in a HetNet scenario where more base stations are involved. XGBoost also showed dominance when using an actual road topology with accuracy of 74% to 95% in a scenario with high randomness.

For future works, we will analyze how the models will perform on a much larger scale together with testing different variations of train-test data split like 70%-30% and 80%-20% partition. We will also evaluate our models using real data from a live network. In due course, we will apply the findings in this paper to some practical applications, such as proactive handover management, load-balancing and energy saving.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant Numbers 1619346, 1559483, 1718956 and 1730650. For further information about these projects please visit www.AI4networks.com.

References

[1] A. Imran, A. Zoha, and A. Abu-Dayya, "Challenges in 5g: how to empower son with big data for enabling 5g," *IEEE Network*, vol. 28, no. 6, pp. 27–33, Nov 2014.

[2] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010. [Online]. Available: https://science.sciencemag.org/content/327/5968/1018

[3] H. Farooq and A. Imran, "Spatiotemporal mobility prediction in proactive self-organizing cellular networks," *IEEE Communications Letters*, vol. 21, no. 2, pp. 370–373, Feb 2017.

[4] T. Anagnostopoulos, C. Anagnostopoulos, and S. Hadjiefthymiades, "Mobility prediction based on machine learning," in *2011 IEEE 12th International Conference on Mobile Data Management*, vol. 2, June 2011, pp. 27–30.

[5] L. M. J. Chen and Y. Xu, "Support vector machine based mobility prediction scheme in heterogeneous wireless networks," *Mathematical Problems in Engineering*, vol. 5, 2015.

[6] Y. Yayeh, H. Lin, G. Berie, A. B. Adege, L. Yen, and S. Jeng, "Mobility prediction in mobile ad-hoc network using deep learning," in *2018 IEEE International Conference on Applied System Invention (ICASI)*, April 2018, pp. 1203–1206.

[7] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[8] A. Zaidi and A. Imran, "Ai4networks simulator for enabling 5g and beyond," *Submitted to IEEE Acess*, 2019.

[9] H. Gebrie, H. Farooq, and A. Imran, "What machine learning predictor performs best for mobility prediction in cellular networks?" *IEEE ICC 2019 First International Workshop on Data Driven Intelligence for Networks and Systems (DDINS)*, 2019.

[10] X. Developers, "Xgboost documentation," 2016. [Online]. Available: https://xgboost.readthedocs.io/en/latest/

[11] A. Rossi, G. Barlacchi, M. Bianchini, and B. Lepri, "Modeling taxi drivers' behaviour for the next destination prediction," 07 2018.