IEEE 18th Wireless and Microwave Technology Conference (WAMICON), April 2017

Base Station Prediction and Proactive Mobility Management in Virtual Cells using Recurrent Neural Networks

Dilranjan S. Wickramasuriya, Calvin A. Perumalla, Kemal Davaslioglu, and Richard D. Gitlin
Department of Electrical Engineering
University of South Florida
Tampa, Florida
Email: {dilranjanw, calvin4}@mail.usf.edu, kdavasli@uci.edu, richgitlin@usf.edu

***Abstract*—Multicell cooperation in 5G next-generation wireless networks is essential to increasing multiuser channel capacity. Multiple base stations need to coherently process their transmitted (or received) data streams to mitigate inter-cell interference and achieve significant diversity gains. This is only possible if the correct base stations are selected. As users demand higher data rates at higher mobility, the time required to predict the optimal set of base stations to create a virtual cell is significantly reduced. In this paper, a method based on a Recurrent Neural Network (RNN) is presented to rapidly predict the next base station that a mobile node will associate with. RNNs have been used in machine learning to identify sequential data patterns such as required in protein sequence classification. In this research a RNN, trained using sequences of Received Signal Strength (RSS) values, is used to predict base station association. Simulation results demonstrate that the proposed machine learning method achieves an accuracy of over 98% to predict the optimal virtual cell topology in the time required based on the mobility of users.**

Keywords—user to base station association; mobility prediction; handoff; Recurrent Neural Network; virtual cells.

I. INTRODUCTION

Providing uninterrupted service to mobile users who are streaming multimedia content with strict time constraints is a challenge for next-generation wireless networks. In a 5G scenario, it is anticipated that a single user will be connected to multiple base stations. It is therefore imperative to find the best set of base stations a mobile user is to be associated with in a timely manner in order to increase channel capacity and mitigate inter-cell interference. Efficient handoff schemes are critical to address this task and several techniques have been proposed in the literature to optimize resource allocation [1]. One approach to reducing latency in handoff is to predict the future location of a mobile node based on its past trajectory. This would typically require storing the precise locations of a user over a certain time window. However, as Lv *et al.* [2] have pointed out, mobile users may be unwilling to continuously transmit GPS information due to increased power consumption and privacy issues.

The most common state-of-the-art technique for user to base station association in Long Term Evolution Advanced (LTE-A) networks is known as *range extension* (or *range expansion*). Here, constant bias values are added to the Received Signal Strength Indicator (RSSI) values of picocell base stations to increase their coverage area. Cell range extension addresses the *uplink-downlink imbalance*. Consider a heterogeneous network where macrocell and picocell base stations are deployed. Certain users may be closer to a picocell base station so that the picocell can receive their uplink signals at higher magnitudes while a macrocell association can yield higher throughput values in the downlink. There have been several studies in the literature to dynamically select these range extension bias values based on the network load through decision rules [3] or through solving a network-wide centralized optimization problem for a *frozen* network [4]–[6]. However, most of these models address the cell selection problem of associating a single user with a single base station. These rules and algorithms do not consider the mobility of the users for anticipatory mobility management. To make things even more difficult, next generation wireless networks will be cooperative in nature, where multiple base stations or remote radio heads (RRHs) will make joint decisions in the uplink signal reception and/or downlink signal transmission. Recent research areas that include network multiple-input multiple-output (network-MIMO) [7], [8], coordinated multipoint (CoMP) transmission and reception [9], [10], virtual cells [7] and cell-free networks [11]–[13], all require methods to support base station association in a dynamic virtual cell formation and heretofore no method has been proposed that addresses the user mobility in an anticipatory and proactive way.

The contribution of this paper is to propose a novel algorithm for proactive and anticipatory mobility management that dynamically selects base stations to form virtual cell topologies. The proposed algorithm is based on Recurrent Neural Networks (RNNs) [14] with long and short term memory. Virtual cell formation and cell-free networks not only provide higher channel capacities for multiple users due to enhanced interference mitigation, but also alleviate the problem of frequent handovers for high mobility users – an important issue in current state-of-the-art wireless networks. Furthermore, an emerging application is that of airborne networks where the time required to form virtual cells is severely time constrained. Our proposed algorithm can be useful in such situations. Based on simulation results, we

show that virtual cell topologies can be rapidly predicted and formed with a very high accuracy based on user mobility.

II. Literature Survey on Machine Learning Approaches for User-Base Station Association Problems

A number of different approaches have been proposed in the literature for mobility prediction in order to make optimal handoff decisions and conserve resources in wireless networks. Here, we review some of the literature where machine learning has been utilized. Michaelis *et al.* [15] used Support Vector Machines (SVMs) to predict the next cell that a mobile node would connect to. Based on actual data recorded in the Lake Geneva region, they extracted short sequences indicating previously associated cell IDs as features and obtained a classification accuracy of almost 78% when a global model was trained. Accuracy was approximately the same when they incorporated location information and trained user-specific generative models with Markov Random Fields. Tkacik and Kordik [16] developed a human mobility predictor based on a Neural Turing Machine. This is a type of RNN capable of identifying temporal patterns in sequences. Given a sequence of where a person had visited each hour, they attempted to predict the next location. Using actual data for six people recorded over a three-month period, their model achieved an accuracy of just over 80% in predicting the next place that the person would be at. Luo *et al.* [17] proposed the use of a Nonlinear Autoregressive Exogenous Model (NARX) based Neural Network (NN) for optimal handover prediction. They considered a scenario with two Access Points (APs) where a person moves from the vicinity of the first to the second one. RSS values and delays from the two APs were fed into the NN having 12 hidden layers. Prediction was based on the history of the past four inputs. The NN was trained with the optimal handover decision being determined by integer optimization. On unseen data, the NN was able to choose a point at which handover was close to the optimal handover location having minimal cost. Javed *et al.* [18] used four different features and AdaBoost with Decision Stumps to predict the occurrence of a handoff in the near future. The features included the past handoff rate, number of cells with RSS above a certain threshold (active set), active set update rate, and signal strength variation. They obtained an accuracy of 80.3% in predicting handoff over a future time window of 60 sec. Liou and Huang [19] used a Neural Location Predictor to predict a mobile node's trajectory and infer handoff. They fed the past three location coordinates of a user into a NN and obtained an accuracy of 78% for a simulation on a mountain road, 86% on a circular road and 90% on a freeway.

Laasonen [20] used incremental clustering to build routes created from sequences of visited cell IDs. These routes comprised of a set of edges in a graph. Based on the node a mobile user was at, the time of day, the day of week and other context variables, a prediction was made for the next major location that the user would visit. Results were presented for three people with accuracy varying between 80-90% and were

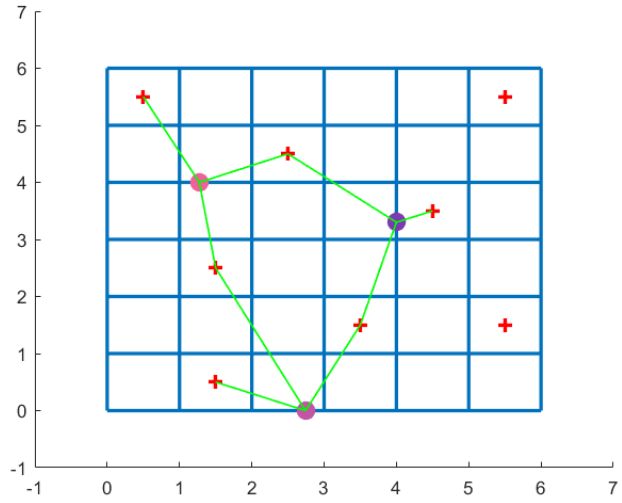

Fig. 1. A road network with eight base stations (crosses) and three mobile nodes (circles), where each node is connected to the three closest base stations.

shown to gradually increase when the model had more trips to learn from. Anagnostopoulos *et al.* [21] explored several different machine-learning techniques to predict a user's next cell ID. They extracted four-element feature vectors indicating the previous cell IDs a user had visited to train their models. They generated artificial data and used a parameter δ to control the degree of randomness of a moving person. A value of δ close to zero would generate a mostly deterministic trajectory, while a value close to one would give rise to a highly random path. They evaluated the performance of Naïve Bayes, k-Nearest Neighbors (kNN), J48 Decision Trees, a rule-based learner (JRip), AdaBoost, Bagging and a vote of kNN and J48. They obtained the best accuracy with the voting scheme which was over 90% with $\delta = 0$ and about 70% when $\delta = 0.25$. Anagnostopoulos *et al.* [22] also incorporated temporal context into the prediction problem by including transition time slots into the feature vectors. Using an approach similar to the one above, they evaluated Naïve Bayes, J48 Decision Trees and JRip. The accuracy they obtained on next cell predictions was similar to [21]. However, they also considered next cell cluster prediction and reported slightly higher accuracies. For instance, they obtained a classification accuracy of over 80% when the degree of randomness of user movement was 0.25.

Despite several machine-learning techniques being proposed in the literature, none of the authors have considered learning on variable-length sequences of RSS values. Additionally, the methods highlighted above only consider a mobile node to be associated with a single base station at a time. In a 5G network, a node might be receiving data from multiple base stations concurrently. As highlighted above, the use of RNNs is proposed in this paper to predict which base station a mobile node will connect to when handoff occurs.

III. Methodology

A. Model

A 6 km × 6 km road network with regularly spaced intersections 1 km apart is assumed, as shown in Fig. 1. A pedestrian or vehicle is randomly generated on a road and begins to move to along the road. A pedestrian has an initial speed drawn from a uniform distribution between 8-12 km/h and a vehicle has an initial speed between 55-65 km/h. We label the pedestrian or vehicle as a mobile node. A mobile node continues to maintain a constant speed until leaving the road network. When a node arrives at an intersection, it decides to move straight ahead with a probability of 0.5 or decides to turn either left or right with an equal probability of 0.25. No U-turns are permitted. Eight base stations were selected for this network topology and placed at arbitrary locations not too far apart.

When a node is within the road network, it maintains its connectivity to the three closest base stations. This was the ground rule we established in this simulation that the RNN had to learn. While more complicated, realistic assignments are possible, we chose this method to demonstrate the capability of RNNs to learn sequences in order to predict base station assignment. We leave the task of learning more complex scenarios to future research.

The following path loss model is adopted, which is suggested by 3GPP [3], with an additional term to account for large scale shadow fading. Here, σ is normally distributed with mean zero and variance 9 dB and d is measured in kilometers.

$$PL(d) = 128.1 + 37.6\log(d) + \sigma. \quad (1)$$

B. Reccurent Neural Networks

The RNN architecture is able to detect patterns in sequential information by learning how present inputs depend on previous ones [14].

The top portion of Fig. 2 shows a typical RNN architecture. It has the typical input (denoted by X_t), hidden layer (H) and output layers (denoted by Y). Here 'h_t' denotes the output of the hidden layer and 't' denotes the time step. Unlike a typical NN, the outputs of the hidden layers do not depend simply on the input layer connections. They also depend on the previous output of the hidden layer. Hence decisions are made based on present inputs and previous hidden layer outputs.

The bottom portion of Fig. 2 shows the 'unrolled' RNN structure. By the term 'unroll' we mean that the RNN structure is now envisioned as a network of sequential relay units in time. In this format, it is easier to envision input time dependencies.

The RNN, while being a powerful intuitive concept, is not very effective in implementation. The problem is that the architecture tends to prioritize the most recent inputs and neglects the effects of those inputs that are further away in the past. This is called the '*exponential decay of error*' or the '*vanishing or exploding error problem*' [23].

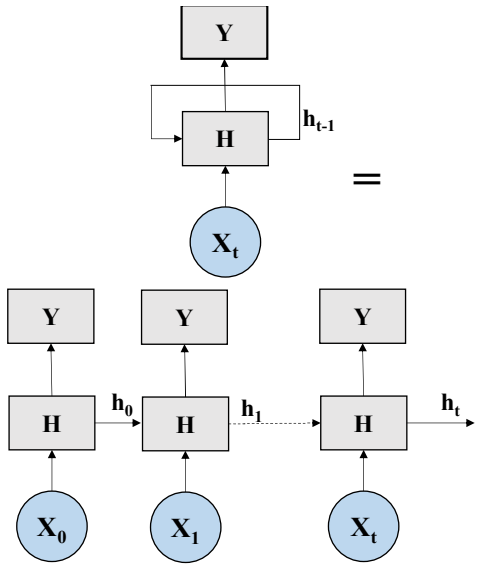

Fig. 2. Top figure shows a typical RNN structure and bottom figure show shows the same structure unrolled in time.

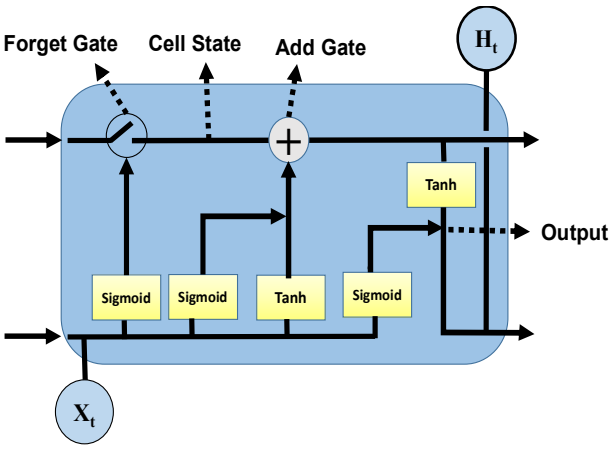

Fig. 3. Long Short Term Memory (LSTM) unit

The solution to this problem was proposed by Hochreiter and Schmidhuber in 1997 [23]. It is called the Long Short Term Memory (LSTM). This architecture and method were applied to the research problem addressed in this paper.

The LSTM unit and its underlying architecture is shown in Fig. 3. All of the hidden layers (H) shown in Fig. 2 are replaced with an LSTM unit. The main difference in the LSTM is the '*error carousal*' or '*cell state*', which is a buffer that holds relevant information of past inputs and affects present and future inputs. In the training phase, based on the present input X_t, the information to be retained or forgotten (forget gate) is learned by two sigmoid neurons and one tanh neuron (i.e., the activation function is a sigmoid or tanh). This is shown in the Fig. 3 as a box labeled 'Sigmoid' and 'Tanh'. Similarly, the add gate adds new information to the cell state based on X_t. The other neurons

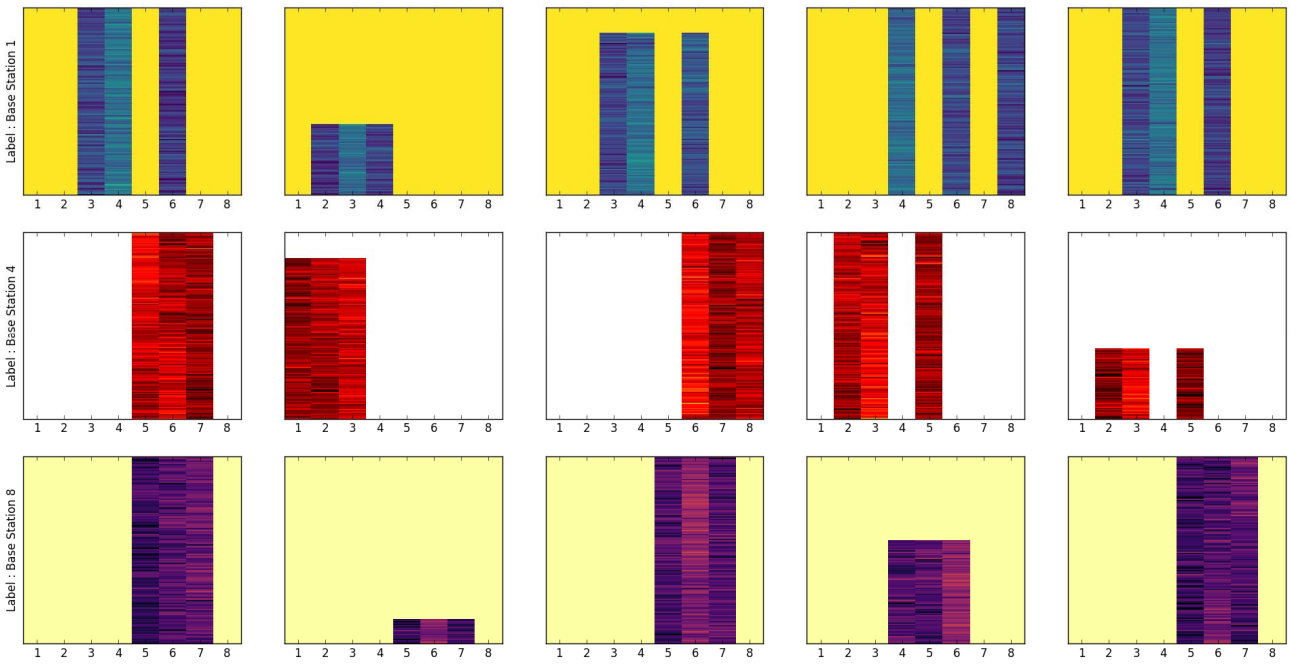

Fig. 4. Sequences of RSS values leading up to a handoff to a particular base station. At each instance in time, the mobile node is connected to three base stations. The RSS value appears as a color intensity. The new base station that the mobile node connects to corresponds to the class label for the sequence.

in the unit then learn the output, H_t, by processing the present input and the cell state information combined. In this way, multiple neurons are used and the cell state is propagated through all these units and gets updated. Relevant information from inputs far away in the past can be retained in the cell state and improve the present decision. Hence, the problem of exponential decay of error is solved.

While multiple neurons per unit leads to a higher computational complexity due to the presence of more neurons to be trained, the resulting training algorithm proposed by Hochreiter and Schmidhuber in [23] improves the computational complexity as compared to traditional RNNs.

C. Sequence Extraction

The RSS values are recorded from each of the three base stations a node is connected to every 500 ms. A vector with eight elements is created and the positions corresponding to the connected base stations are populated with their RSS values. The other elements are zero. These consecutive RSS feature vectors are then stored in a queue data structure. Since RSS values from a long time in the past do not have predictive value regarding where a node maybe headed, the oldest elements are dropped from the queue when its size exceeds 150. Whenever a handoff occurs and the node is closer to a new base station than a previous one, we extract the sequence, label it with a number corresponding to the new base station, and reset the queue. The very last RSS feature vector is discarded from the queue prior to labeling, as this would have an RSS value corresponding to the position of the newly associated base station. Very short sequences are also discarded, as the RNN would need sufficient data in order to make a reliable prediction. We simulated mobile nodes in this manner until we collected a total of 100,000 labeled sequences. A few RSS sequences have been shown in Fig. 4.

D. Classification

Unequal-length sequence classification is a well-known problem that RNNs have been applied to and Google's Python-based Tensorflow library was used to train and test the model. In this case the model needs to predict which of the eight base stations a node would connect to at the point of handoff, based on a sequence of RSS values recorded in its immediate history. The model itself, based on the code in [24], had 640 neurons and a learning rate set to 0.0003. These values were selected after experimenting with different combinations in order to maximize accuracy. The RNN takes about 35 min. to train on a CUDA-enabled laptop with an i7 processor, 16 GB RAM and a GeForce GTX 950M graphics card for the specified number of epochs. In reality, training would only be performed infrequently and in an offline environment. Once the coefficients have been obtained, predictions can be provided much more rapidly.

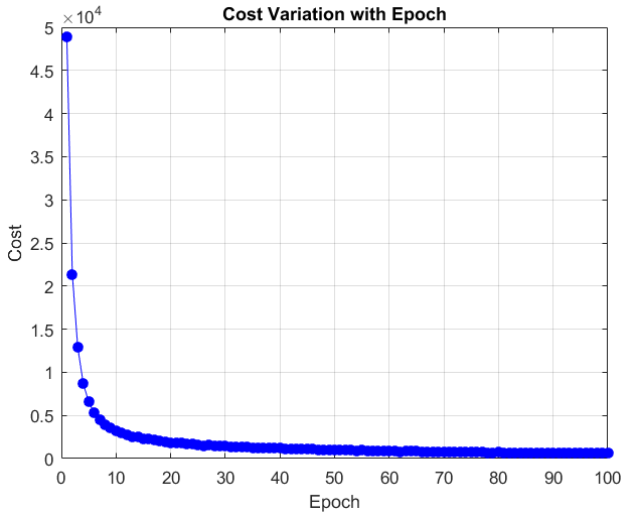

Fig. 5. Convergence of model training cost calculated based on cross entropy

IV. Simulation Results

A total of 70,000 sequences were selected for training and 30,000 sequences were used for testing the RNN. With 100 training epochs, the testing error gradually converges and a maximum accuracy of 98.3% was obtained in this 8-class sequence classification problem. The RNN model is gradually built up by penalizing a cost measured by cross entropy. Fig. 5 shows the gradual decrease of this cost with training epoch number. At each epoch, the model predicts classes and evaluates error on 30,000 sequences in approximately 2.15 s.

V. Conclusions and Future Work

Proactive mobility management for 5G wireless networks has been addressed in this paper. Focus was directed towards base station prediction and the user-to-base station association problem for CoMP transmissions and virtual cell selection. A 98.3% accuracy was achieved by a RNN classifier indicating that it can learn patterns in arbitrary-length sequences to make accurate predictions about which base station a mobile node is going to be associated with in the future. Although the base station assignment considered in this simulation is rather simplistic in that a mobile node only connects to the closest three base stations, the proposed algorithm will be extended in future work to include load balancing and different levels of quality of service constraints per user. Incorporating these constraints into the problem will alleviate congestion due to heterogeneous user distribution and different QoS level demands that would yield an even more effective resource allocation. Considering the 1 ms low-latency requirements of next generation 5G networks, the proposed algorithm helps mobility management function facilitate virtual cell formations.

References

[1] Eunsoo Shim, Hung-yu Wei, Yusun Chang and R. D. Gitlin, "Low latency handoff for wireless IP QoS with NeighborCasting," in *Proc. IEEE International Conference on Communications*, 2002, pp. 3245-3249.

[2] Q. Lv, Y. Qiao, N. Ansari, J. Liu, and J. Yang, "Big data driven hidden Markov model based individual mobility prediction at points of interest," *IEEE Trans. Veh. Technol.,* to be published.

[3] K. Davaslioglu and E. Ayanoglu, "Interference-based cell selection in heterogeneous networks," in *Proc. ITA Workshop*, San Diego, 2013, pp. 1-6.

[4] R. Madan, J. Borran, A. Sampath, N. Bhushan, A. Khandekar, and T. Ji, "Cell association and interference coordination in heterogeneous LTE-A cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 9, pp. 1479-1489, 2010.

[5] K. Son, H. Kim, Y. Yi, and B. Krishnamachari, "Base station operation and user association mechanisms for energy-delay tradeoffs in green cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 8, pp. 1525-1536, 2011.

[6] S. M. Perlaza, E. V. Belmega, S. Lasaulce, and M. Debbah, "On the base station selection and base station sharing in self-configuring networks," in *Proc. 4th International ICST Conference on Performance Evaluation Methodologies and Tools*, 2009, pp. 71:1-71:10.

[7] J. Kim, H. W. Lee, and S. Chong, "Virtual cell beamforming in cooperative networks," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1126-1138, 2014.

[8] D. Gesbert, S. Hanly, H. Huang, S. Shamai Shitz, O. Simeone, and W. Yu, "Multi-Cell MIMO cooperative networks: a new look at interference," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 9, pp. 1380-1408, 2010.

[9] H. Dahrouj and Wei Yu, "Coordinated beamforming for the multi-cell multi-antenna wireless system," in *Proc. 42nd Annual Conference on Information Sciences and Systems*, 2008, pp. 429-434.

[10] P. Marsch and G. P. Fettweis, *Coordinated Multi-Point in Mobile Communications: From Theory to Practice*, New York, Cambridge University Press, 2011.

[11] E. Nayebi, A. Ashikhmin, T. L. Marzetta and H. Yang, "Cell-free massive MIMO systems," in *Proc. 49th Asilomar Conference on Signals, Systems and Computers*, 2015, pp. 695-699.

[12] H. Ngo, A. Ashikhmin, H. Yang, E. Larsson; T. Marzetta, "Cell-free massive MIMO versus small cells," *IEEE Trans. Wireless Commun.*, to be published.

[13] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson and T. L. Marzetta, "Cell-free massive MIMO: uniformly great service for everyone," in *Proc. IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications*, 2015, pp. 201-205.

[14] R. J. Williams and D. Zipser, "Gradient-Based Learning Algorithms for Recurrent Networks and Their Computational Complexity" in *Backpropagaion*, Y. Chauvin and D. E. Rumelhart, Eds. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1995, pp. 433–486.

[15] S. Michaelis, N. Piatkowski, and K. Morik, "Predicting next network cell IDs for moving users with discriminative and generative models, " in *Proc. Mobile Data Challenge Workshop (Nokia) in conjunction with International Conference on Pervasive Computing,* 2012.

[16] J. Tkacik and P. Kordik, "Neural turing machine for sequential learning of human mobility patterns," in *Proc. International Joint Conference on Neural Networks*, 2016, pp. 2790-2797.

[17] Y. Luo, P. N. Tran, C. An, J. Eymann, L. Kreft, and A. Timm-Giel, "A novel handover prediction scheme in content centric networking using nonlinear autoregressive exogenous model," in *Proc. IEEE Vehicular Technology Conference*, 2013, pp. 1-5.

[18] U. Javed, D. Han, R. Caceres, J. Pang, S. Seshan, and A. Varshavsky, "Predicting handoffs in 3G networks," *SIGOPS Oper. Syst. Rev.*, vol. 45, no. 3, pp. 65-70, 2011.

[19] S. Liou and Y. Huang, "Trajectory predictions in mobile networks," *International Journal of Information Technology*, vol. 11, no. 11, pp. 109-122, 2005.

[20] K. Laasonen, "Clustering and prediction of mobile user routes from cellular data," in *Proc. 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2005, pp. 569-576.

[21] T. Anagnostopoulos, C. Anagnostopoulos, S. Haadjiefthymiades, M. Kyriakakos, and A. Kalousis, "Predicting the location of mobile users: a machine learning approach," in *Proc. International Conference on Pervasive Services*, 2009, pp. 65-72.

[22] T. Anagnostopoulos, C. B. Anagnostopoulos, S. Haadjiefthymiades, A. Kalousis , and M. Kyriakakos, "Path prediction through data mining," in *Proc. IEEE International Conference on Pervasive Services,* 2007, pp. 128-135.

[23] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," in *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, Nov. 15 1997.

[24] D. Hafner. (2016). *Introduction to recurrent neural networks in Tensorflow* [Online]. Available: https://danijar.com/introduction-to-recurrent-networks-in-tensorflow/