Mobility Prediction via Sequential Learning for 5G Mobile Networks

Francesca Meneghello, Davide Cecchinato, Michele Rossi
Department of Information Engineering University of Padova, Padova, Italy
{meneghello, cecchin4, rossi}@dei.unipd.it

***Abstract*—Here, we present a mobility prediction framework for 5G mobile systems. Our work stems from the intuition that mobility in vehicular networks is highly correlated, and such correlation can be captured by advanced neural network designs to anticipate the users' point of attachment. To prove this, we combine Markov chains with recurrent and convolutional neural networks, training them on mobility trajectories estimated by the received radio signal from mobile millimeter-wave devices. The proposed framework is decentralized, i.e., user trajectories are independently learned by each base station. In this paper, various problems are pragmatically tackled and solved, such as dealing with imbalanced datasets, as some trajectories are under represented, and obtaining a mobility classifier whose accuracy increases as new mobility samples are collected.**

The proposed technique is assessed using emulated traces obtained through the SUMO mobility simulator for the city of Cologne. Numerical results show accuracies higher than 88% in the prediction of the next serving base station from 4 seconds before the handover is performed. Mobility (next base station) predictors like the ones presented here are key for network management purposes within 5G networks, e.g., to proactively allocate communication and edge computing resources.

***Index Terms*—5G, mobility prediction, radio-positioning, recurrent neural network, convolutional neural network.**

I. INTRODUCTION

Fifth-generation (5G) mobile networks are expected to provide broadband access in dense areas, high communication capacity and ultra-low latency, for mobile users [1], [2]. These features will be enabled by the introduction of, among others, millimeter-wave (mm-wave) communications and massive multiple-input multiple-output (MIMO) technologies.

Besides other benefits, the high bandwidth available at mm-wave frequencies and the antenna arrays employed for massive-MIMO, will allow 5G networks to provide *network-based positioning* that, with an expected accuracy of less than one meter, will outperform that provided by the global navigation satellite system (GNSS) (five meters) [3]. These positioning capabilities can be exploited to enhance the 5G network performance, through location- and mobility-aware allocation of communications and computing tasks in 5G edge networks [4], [5]. Several proactive resource allocation schemes are investigated in the literature, exploiting *short-term* and *long-term* predictions. Short-term predictions, i.e., from one to a few radio frames ahead, can be utilized at runtime to tune the mobile network parameters. For example, location-based prediction of interference sources can be used within channel access protocols. As an example, in [4], [7] short-term location estimates are utilized to perform beamforming for 5G enabled vehicles and high-speed trains. Longer-term mobility predictions, on the other hand, can be exploited to re-route processes at the network layer [6].

Paper contribution: our work focuses on long-term mobility predictions, i.e., ranging from one to a few seconds into the future. Our idea is that 5G base stations (BSs) could independently track the connected users to estimate their physical trajectories as they move through the radio cells. We target vehicular networks, where mobility is constrained to the physical connectivity structure of road links, as well as to their morphology. Our intuition is that vehicular mobility is a highly correlated process, which can be effectively captured by the BSs by observing the radio signals from their connected mobile users. In the literature, this prediction task has been tackled exploiting Markov chains (MC) and support vector machines (SVMs), using as input the sequence of *previously visited cells* [8]–[10]. Although this approach may be effective in simple scenarios, we remark that the historical information about the sequence of serving BSs does not suffice to provide accurate position estimates as the number of physical trajectories increases. As a solution, we propose to exploit the location information that will be provided by 5G networks, using the sequence of users' positions with a granularity of one second. Our approach is *decentralized*, as each BS tracks its own connected users by estimating the probability distribution of their next serving BS (taken from the set of neighboring ones). Learning is carried out through recurrent and convolutional neural networks (NNs) combined with standard MCs. Once trained, the predictor returns mobility estimates, i.e., a probability vector for the next serving BS, from the moment a user connects to the cell to the moment it leaves it. Such estimates are continuously improved as subsequent radio samples from the user are collected.

To the best of our knowledge, the only other work adopting a similar approach is [11], where a recurrent neural network (RNN) is used to estimate the next BS that the user will be connected to from the received signal strength (RSS) values acquired every 500 ms. In [11], a simple RNN design is considered, training it in a *centralized* fashion on a small-scale scenario featuring 8 BSs and 3 users moving in a grid topology over an area of 6×6 square kilometers. Our present work provides significant advancements over this prior research along several dimensions: *i)* a real-life mobility scenario is considered, using the "simulation of urban mobility" (SUMO) [12] to emulate more than 50 thousands vehicles moving within the city of Cologne; *ii)* a realistic 5G network is emulated, with densely deployed BSs within the monitored area (coverage radius of 40 m, for a total of 196 BSs); *iii)* we combine RNN with Markov chain estimators, retaining the benefits of both; *iv)* the approach is decentralized by design, i.e., a single NN-MC based predictor is trained for each BS, solving issues such as the lack of data for less represented trajectories. Numerical results indicate that the new serving BS can be estimated with an accuracy higher than 88% from four seconds before the handoff, which seems to be appropriate for proactive resource management algorithms.

The rest of the paper is organized as follows. In Section II we detail the proposed mobility prediction framework. In Section III we describe the experimental scenario and the performance of our technique is assessed in Section IV. Concluding remarks are presented in Section V.

II. SEQUENTIAL LEARNING ARCHITECTURE

Next, we detail the proposed mobility estimation framework. The sequence of azimuth (α) and elevation (β) angles that describes the positions of the vehicle from the time it connects to a BS ($t = 1$) to the instant in which it hands over to another BS ($t = T$) is referred to as a *trajectory*. Formally, a T-long trajectory is denoted by $\mathbf{S}_T \triangleq (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_T)$, with $\mathbf{s}_t = (\alpha_t, \beta_t)^T$. Our framework returns a new soft-prediction, in the form of a probability vector over the next serving BS, every time a new trajectory sample is gathered, with an accuracy that increases with the number of samples. Considering that the BS to which the user will hand over must be one among the N_{BS} adjacent BSs, we formulate the problem as a classification task and we design a supervised learning framework to carry it out. The classification is perfomed in parallel by two blocks: *i)* a NN block, consisting of a recurrent neural network (RNN) (see Section II-A) and a convolutional neural network (CNN) (Section II-B), that uses as input the user trajectory samples, and *ii)* a MC block (as proposed by [9]) that exploits the sequence of previously visited cells (Section II-C). The classification results from *i)* and *ii)* are then combined to obtain the final estimates using a parameter that is learned as part of the training (see Section II-D). The blocks are shown in Fig. 1, while the training process is detailed in Section II-F. To complete our framework, we propose a pre-processing algorithm to deal with possible imbalanced datasets (Section II-E), which is key to also learn under represented trajectories, for which there are fewer training examples.

A. GRU-based RNN for mobility feature extraction

RNN is the most used learning tool for sequential data, as it is able to track temporal correlation among input samples and handles inputs with different numbers of elements [13]. An RNN is composed of one or more *memory cells* whose number is indicated by the number of layers, L_{rec}. The temporal correlation between subsequent input samples, from $\mathbf{s}_1$ to $\mathbf{s}_t$, is recorded in the values of the N_{rec} neurons constituting the cell *internal state*, $\mathbf{h}_t$, where t is the time index. To that end, the input sequence $\mathbf{S}_t \triangleq (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_t)$, is fed one sample at a time into the first layer of the RNN. In case of multiple layers, the internal state of the first layer cell becomes the input of the second layer cell, and so on. Every time a new sample arrives, the internal states of the RNN cells are updated based on the values of the previous states and on the new sample. The update procedure is described by the *cell update function*, that defines the memory cell, whose parameters are adjusted to minimize a *loss function*, i.e., a user-defined distance between the expected and the actual network output. Dropout layers can be inserted between recurrent layers, providing network regularization by randomly zeroing some of the elements of the input, following a Bernoulli distribution. In our implementation, we use $L_{\mathrm{rec}} = 2$ gated recurrent units (GRUs) [14] memory cells with $N_{\mathrm{rec}} = 20$ neurons per layer, and a dropout layer with a retain probability of 0.9. For each input sequence, we collect the internal state values of the two stacked RNN cells, each time a new trajectory sample is processed. Note that the RNN internal state is a *feature vector* capturing the most representative traits of the input time series. After applying layer normalization, this feature vector is used by the subsequent CNN that acts as a classifier, see Fig. 1.

B. CNN for next serving BS prediction

A CNN consists of the cascade of layers (i.e., functions), where all or some of them use the *convolution operator*, i.e., small-size kernels (matrices of weights), which are convolved with the entire input data volume [13]. The result of such convolutions is passed through a non-linearity, called *activation function*, to produce an output *activation map*. The number of kernel filters employed at one layer defines the *depth* of the layer output, i.e., the number of activation maps produced for a single input, each of which captures different aspects of the input. One interesting feature of CNNs is their *parameter sharing* property: the kernel filters are applied to each position of the input, allowing one to capture specific input peculiarities, no matter their specific location. Similarly to RNNs, dropout layers can be interposed between convolutional layers. In this work, we use a one dimensional CNN, where the kernels are vectors, with $L_{\mathrm{conv}} = 2$ convolutional layers with `ReLu` activation functions, and two dropout layers with a retain probability of 0.9. Batch normalization is applied after each convolutional layer to reduce the training time. The CNN input corresponds to the two (stacked) final internal states extracted from the GRU-based RNN. The output of the last convolutional layer is then passed through a fully connected layer with `SoftMax` activation function that produces a N_{BS}-dimensional vector, $\mathbf{p}_t^{\mathrm{conv}} \triangleq (p_{t,1}^{\mathrm{conv}}, p_{t,2}^{\mathrm{conv}}, \ldots, p_{t,N_{\mathrm{BS}}}^{\mathrm{conv}})$, containing the probability that the user will hand over to each of the N_{BS} neighboring cells after leaving the current one.

C. MC for next serving BS prediction

MC is here used to capture the correlation in the sequence of cells that are visited by a user. Indicating with $c_{b+1,i}$ the i-th neighbouring cell of the current (serving) BS,

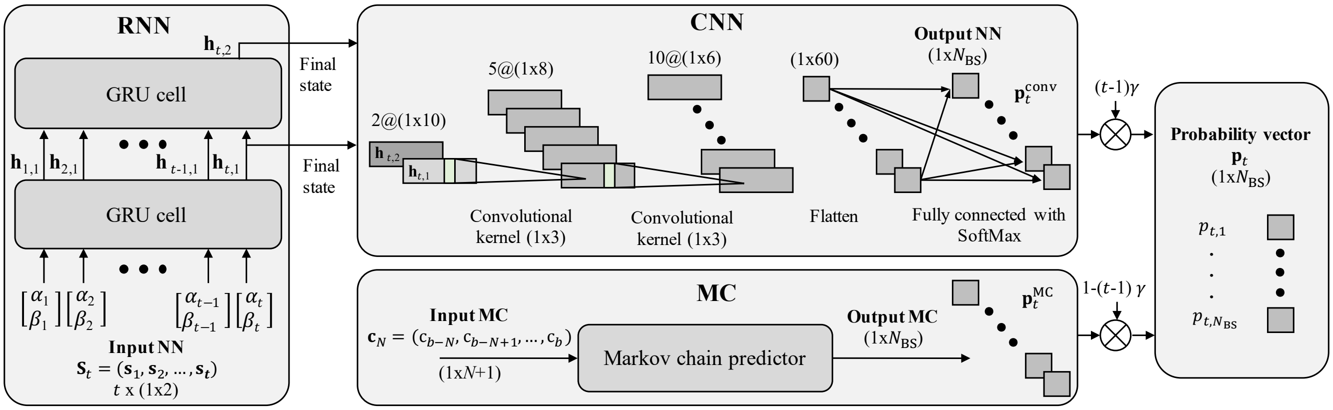
RNN
GRU cell
GRU cell
$\mathbf{h}_{t,2}$
$\mathbf{h}_{1,1}$ $\mathbf{h}_{2,1}$ $\mathbf{h}_{t-1,1}$ $\mathbf{h}_{t,1}$
$\begin{bmatrix}\alpha_1\\ \beta_1\end{bmatrix}\begin{bmatrix}\alpha_2\\ \beta_2\end{bmatrix}\begin{bmatrix}\alpha_{t-1}\\ \beta_{t-1}\end{bmatrix}\begin{bmatrix}\alpha_t\\ \beta_t\end{bmatrix}$
Input NN
$\mathbf{S}_t = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_t)$
t x (1x2)
Final state
Final state
CNN
2@(1x10)
$\mathbf{h}_{t,2}$
$\mathbf{h}_{t,1}$
5@(1x8)
10@(1x6)
(1x60)
Output NN
(1x$N_{\rm BS}$)
$\mathbf{p}_t^{\rm conv}$
Convolutional kernel (1x3)
Convolutional kernel (1x3)
Flatten
Fully connected with SoftMax
$(t$-1$)\gamma$
MC
Input MC
$\mathbf{c}_N = (c_{b-N}, c_{b-N+1}, \ldots, c_b)$
(1xN+1)
Markov chain predictor
Output MC
(1x$N_{\rm BS}$)
$\mathbf{p}_t^{\rm MC}$
1-$(t$-1$)\gamma$
Probability vector
$\mathbf{p}_t$
(1x$N_{\rm BS}$)
$p_{t,1}$
$p_{t,N_{\rm BS}}$

Fig. 1: Processing pipeline. The estimation is performed in parallel by the NN block (GRU-based RNN + CNN), using as input the (α, β) angles describing the trajectories, and the MC block, exploiting the sequence of previous visited cells. The output vector of probabilities for each adjacent BS ($\mathbf{p}_t$) is obtained as a combination of the two classification outputs.

$i \in \{1, \ldots, N_{\rm BS}\}$, and with $\mathbf{c}_N = (c_{b-N}, c_{b-N+1}, \ldots, c_b)$ the ordered sequence of the last $N+1$ cells visited up to and including time t, each element of the $N_{\rm BS}$-dimensional vector $\mathbf{p}_t^{\rm MC} \triangleq (p_{t,1}^{\rm MC}, p_{t,2}^{\rm MC}, \ldots, p_{t,N_{\rm BS}}^{\rm MC})$ is computed as follows:

$$p_{t,i}^{\rm MC} = \frac{\sum_{\rm dataset} \texttt{Number}(c_{b+1,i}, \mathbf{c}_N)}{\sum_{\rm dataset} \texttt{Number}(\mathbf{c}_N)}, \tag{1}$$

where $\texttt{Number}(c_{b+1,i}, \mathbf{c}_N)$ (resp. $\texttt{Number}(\mathbf{c}_N)$) indicates the number of instances for the sequence $(c_{b+1,i}, \mathbf{c}_N)$ (resp. $\mathbf{c}_N$) in the training dataset. The value of N, i.e., the MC order, results from a numerical performance assessment, see Section IV.

D. Combination of NN and MC predictors

The final $N_{\rm BS}$-dimensional probability vector $\mathbf{p}_t \triangleq (p_{t,1}, p_{t,2}, \ldots, p_{t,N_{\rm BS}})$ is obtained by combining the NN and MC estimates through a parameter $\gamma \in \{0, 1\}$, which is fixed for each NN-MC predictor and automatically learned during training. The parameter γ is used together with the variable t, allowing the predictor to differently weigh the two estimates as the number of gathered trajectory samples t changes:

$$k = \min\{(t-1)\gamma, 1\} \ \in \{0, 1\}, \tag{2}$$

$$\mathbf{p}_t = k\mathbf{p}_t^{\rm conv} + (1-k)\mathbf{p}_t^{\rm MC}. \tag{3}$$

E. Dealing with imbalanced datasets

In an urban scenario, the number of examples available for each of the possible trajectories inside a BS's coverage area usually differs. Using such an imbalanced dataset for training would lead to a neural network with poor mobility prediction capabilities for those users that follow under represented trajectories. To cope with this, we propose a method to artificially generate new trajectories, so that they resemble the ones belonging to the under represented classes. This oversampling process is only performed on the training dataset, whose trajectories are labeled with the indication of the next BS towards which the user will be moving after the handover. Based on this label, the input data is split into different *groups*. The groups whose cardinality is smaller than 70% the most populated one undergo an oversampling process, while the others remain unchanged, as they contain enough samples to accurately learn their trajectories. The trajectories belonging to each of these groups are further split into a number of *classes* reflecting a quantization on the trajectory space, i.e., by grouping together those trajectories that lead to the same final BS (same group). To do this, we first obtain fixed-length trajectory representations (*codes*). A simple and common coding method in machine learning is *padding*, which amounts to concatenating each input vector with as many zeros as needed to reach a common (fixed) length. However, in our case, this is not the best choice as the inputs have a specific physical meaning (city roads). Instead, we interpolate the trajectories with a degree 3 polynomial function, and use the interpolated versions, i.e., the codes, as inputs for the (unsupervised) "hierarchical density-based spatial clustering of applications with noise" (HDBSCAN) algorithm [15]. For each trajectory class, we then produce artificial trajectories picking at random trajectories from the class, and adding white Gaussian noise with standard deviation $\sigma = 5 \times 10^{-4}$ radiants to each sample. The new trajectories are assigned to the proper label and are ready to be used for the model training.

F. Training process

Next, we detail how the building blocks composing our solution are combined and jointly trained, as summarized in Algorithm 1. First, the training trajectories are used to generate artificial examples to balance the training dataset. Hence, from each (T-long) trajectory we extract all the possible $(T-1)$ sub-trajectories of size greater than two, i.e., ($[\mathbf{s}_1, \mathbf{s}_2]$, $[\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3]$, $\ldots[\mathbf{s}_1, \mathbf{s}_2, \ldots \mathbf{s}_T]$). Each of these sub-trajectories is forwarded through the NN block to obtain the probability vector $\mathbf{p}_t^{\rm conv}$ for the next BS. The sequence of previously visited cells $\mathbf{c}_N$ is used to compute the probability vector $\mathbf{p}_t^{\rm MC}$, which is then combined with $\mathbf{p}_t^{\rm conv}$ to obtain $\mathbf{p}_t$. $\mathbf{p}_t$ is then compared with the ground truth to determine the prediction loss. (The ground truth is a $N_{\rm BS}$-dimensional "one hot vector" with the element corresponding to the actual next BS being equal to 1, and the other ones to 0). The loss is computed by multiplying the negative log-likelihood by an exponential penalty term that depends on the sub-trajectory length (t) and on the total length (T) of the trajectory as follows:

$$\text{penalty}(t) = 2^{t/T} - 1. \tag{4}$$

The backpropagation of sub-trajectory prediction losses allows the network to adjust its parameters to correctly estimate the next BS even when the complete T-long trajectory is not yet available. Through Eq. (4), higher penalties are assigned as the number of collected samples, t, approaches T. This forces the network to become more accurate as the number of samples increases. Every 20 epochs, the performance of the framework is assessed using the validation data. The RNN+CNN weights and the combination parameter (γ) are extracted and saved if they outperform the previously saved ones in terms of validation accuracy. Otherwise, an ε-greedy approach is adopted, see, e.g., [16]: with probability $\varepsilon = 0.3$ the training continues from the weights computed in the last iteration, while with probability $1 - \varepsilon$ it restarts from the previously saved network parameters. Besides, every 20 epochs the artificial training examples used to balance the dataset are re-computed. These combined actions increase the generalization capacity of the framework as during each training round (20 epochs) the dataset is re-populated maintaining all the real training data, while adding *different* artificial examples.

Algorithm 1 Deep learning framework: training process.

```
- num_epochs ← 100, $\varepsilon \leftarrow 0.3$, max_val_accuracy ← 0;
for $j \leftarrow 1$ to (num_epochs/20) do
    - add training trajectory examples – Section II-E;
    - extract training sub-trajectories;
    for $jj \leftarrow 1$ to 20 do
        - split the set of sub-trajectories in batches with batch_size = 64 examples each;
        for all batches do
            - forward propagate the input – Section II-F;
            - compute the prediction error using penalty in (4);
            - backpropagate the error through the network;
    - assess prediction performance with validation set;
    - val_accuracy ← accuracy on the validation set;
    if val_accuracy > max_val_accuracy then
        - save network parameters (weights, biases, $\gamma$);
        - max_val_accuracy ← val_accuracy;
    else if random(0, 1) > $\varepsilon$ then
        - restore previously saved network parameters;
```

III. EXPERIMENTAL MOBILITY SETUP

We consider an urban 5G scenario in the center of the city of Cologne (900×700 square meters), with 5G enabled vehicles that move around while being continuously connected to the 5G network. The mobility is emulated with SUMO, an open-source traffic simulation suite that allows generating the movement of emulated users around a predefined city road map, and extracting the needed metrics [12]. BSs are deployed on a grid topology, and the mobility area is covered with hexagonal 5G cells with a radius of 40 m, each with a BS in its center. With this topology, each BS has a total of $N_{\text{BS}} = 6$ neighboring BSs, and depending on the specific road links morphology, a vehicle exiting from the coverage area of one BS, can visit a subset of them, i.e., there are n possible next BSs, with $n \in \{1, \ldots, 6\}$. Note that the 5G communication protocol stack is not emulated, as it is not needed for the purpose of position tracking. We only consider the link establishment phase, as detailed in the next section.

A. Channel model for user-BS association

For the wireless link between the vehicles and the BSs, we consider the mm-wave channel model of [17]. At every point in time, each vehicle (user) connects with the BS providing the best communication conditions, measured in terms of path loss (PL). Indicating with d the distance between the transmitter and the receiver, and with $\mathcal{N}(0, \sigma^2)$ a Gaussian random variable with zero mean and variance σ^2, we have:

$$\text{PL}(d) = \rho_1 + \rho_2 10 \log_{10}(d) + \xi \quad [\text{dB}], \tag{5}$$

$$\xi \sim \mathcal{N}(0, \sigma^2) \quad [\text{dB}]. \tag{6}$$

The channel parameters are set to $\rho_1 = 61.4$, $\rho_2 = 2$ and $\sigma = 5.8$ dB, as defined in [17] for the 28 GHz LOS model.

For the user association, a simple hysteresis mechanism, with parameter ε_P, is considered. Specifically, a handover is performed to a new BS only if the power received from this new BS exceeds that received from the serving BS by an amount greater than $\varepsilon_P = 2$ dB.

B. Vehicle location information acquisition

High positioning accuracy is a key requirement for 5G systems, and cellular network based positioning in the three dimensional space should be supported with accuracy from 10 m to less than 1 m, depending on the application [18], [19]. Network based positioning was considered as an optional feature in previous generation cellular systems and was implemented mainly for emergency calls, through different techniques [20]. Among them, the ones that provide the highest accuracy are based on trilateration and triangulation. The former relies on time of arrival, time difference of arrival or received signal strength metrics, while the latter exploits the angle of arrival (AoA) of the received signals. These techniques entail the combination of the measurements from three different BSs. Instead, in 5G networks, the use of two dimensional antenna arrays with the introduction of massive MIMO enable two dimensional terminal positioning by just using a single AoA measurement [21].

We emulate such single BS AoA positioning approach by processing the x-y traces extracted from the SUMO simulator to obtain the angular information. We add white Gaussian noise to the extracted data to account for estimation errors. Specifically, indicating with h_{BS} the height of the BS, the azimuth (α) and elevation (β) components of the AoA are obtained as follows, see Fig. 2,

$$\Delta x = x_{\text{user}} - x_{\text{BS}} + w, \quad w \sim \mathcal{N}(0, \sigma_w^2), \tag{7}$$

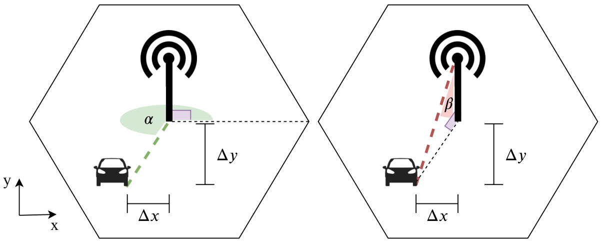

Fig. 2: Azimuth (α) and elevation (β) angles that identify the position of each vehicle inside the (serving) cell. The relative coordinates of the vehicle with respect to the BS are indicated with Δx and Δy. 90-degree angles are drawn in violet.

$$\Delta y = y_{\text{user}} - y_{\text{BS}} + w, \quad w \sim \mathcal{N}(0, \sigma_w^2), \tag{8}$$

$$\alpha = \begin{cases} \arctan\left(\frac{\Delta y}{\Delta x}\right), & \text{if } \Delta x > 0 \\ \arctan\left(\frac{\Delta y}{\Delta x}\right) + \pi, & \text{otherwise} \end{cases}, \tag{9}$$

$$\beta = \arctan\left(\frac{\sqrt{\Delta x^2 + \Delta y^2}}{h_{\text{BS}}}\right). \tag{10}$$

We set $h_{\text{BS}} = 8$ m and $\sigma_w = 5/3$ m in our simulations, accounting for positioning errors of up to 5 meters ($\pm 3\sigma_w$).

IV. Prediction performance

We trained the proposed mobility prediction framework using 24 hour long SUMO mobility traces, acquired with a granularity of one second. Trajectories are grouped based on the current serving BS and labeled with the next serving BS identifier, using the user-BS association procedure described in Section III-A. Each of the 196 BS in the deployment obtains a trajectory dataset from the radio measurements gathered within the radio cell, that is used to train and evaluate the NN-MC mobility predictor for that BS. Trajectory data is split as follows: 80% for training, and 20% evenly divided between validation and test sets. Note that although the architecture is the same for all the BSs, the NN weights and the combination parameter γ are independently adjusted during the training processes, and are in turn BS specific.

The proper MC order is selected by evaluating the next radio cell prediction accuracy obtained using the MC block only, with different N values. As can be seen from Fig. 3, the best choice is $N = 2$, as using higher MC orders does not lead to any noticeable accuracy increase. Note that if the number of previously visited cells M at time t is smaller than N, the prediction is performed by training a MC with order $M < N$. The case when a single BS can be reached form the current BS is also shown in Fig. 3 for completeness: for it, as expected, the prediction accuracy is always 100%.

In Fig. 4, we show the hard-decision accuracy obtained by averaging the results achieved by the 196 NN-MC classifiers, grouped based on the number of BSs in their handover set ($n = 2, 3, 4$). For comparison, we also report the accuracies achieved separately by the MC classifiers and NN classifiers (i.e., RNN and CNN jointly trained without the MC contribution). We do not report the cases when the handover set contains more than four BSs because they rarely occurred, and hence we could not collect enough data for reliable training and assessment. As expected, the NN and the NN-MC approaches outperform the state of the art MC classifiers, proving the importance of collecting users' positions within the serving cell to estimate the next serving BS, rather than only relying on the sequence of previously visited cells. The NN-MC approach, based on the combination of these two types of information, achieves the highest accuracies. In all the cases, the average accuracy is greater than 88% starting from four seconds before the handover and remains above 82% until six seconds before, which represents the average time the users remain inside a cell.

V. Concluding remarks

In this article, we have proposed a decentralized technique for the estimation of mobility trajectories in 5G vehicular networks. Our objective was to predict, with high accuracy, the next serving BS a few seconds before the handover occurs. Such estimates are computed by each BS by processing the radio signals from its connected users. A prediction framework combining neural networks and Markov chains is put forward, detailing its design and proposing a novel training approach that effectively copes with under represented trajectories. To assess the suitability of our algorithm for an urban environment, mobility trajectories of vehicles were collected through the SUMO mobility simulator for the city of Cologne, considering a deployment with 196 base stations and more than 50 thousands mobile users. The final results confirm the effectiveness of the proposed technique, showing accuracies greater than 88% for predictions performed four seconds before the handover. Our approach can be easily adapted to differing scenarios in terms of 5G network topology and vehicles' speed. The vehicle position acquisition rate can be changed based on user/application needs, without any modifications to the algorithm, and is only constrained to the available computing power at the network edge. In any setup, a hyper-parameter selection phase should be performed to adapt the prediction framework to the specific scenario. Our technique can be utilized in conjunction with resource allocation algorithms, e.g., to proactively allocate communication and computing tasks prior to the actual handover events. Proactive strategies are expected to enhance the quality of service experienced by the users, while improving the way in which network resources are assigned. Investigation on this front is left for future work.

Acknowledgment

This work has received funding from the Italian Ministry of Education, University and Research (MIUR) through the PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges". This work has also been supported, in part, by MIUR (Italian Ministry of Education, University and Research) through the initiative

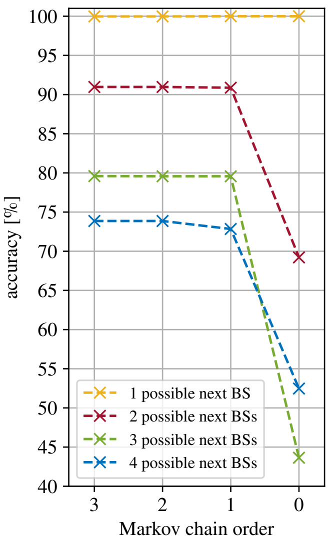

Fig. 3: Next cell estimation accuracy using different order MCs.

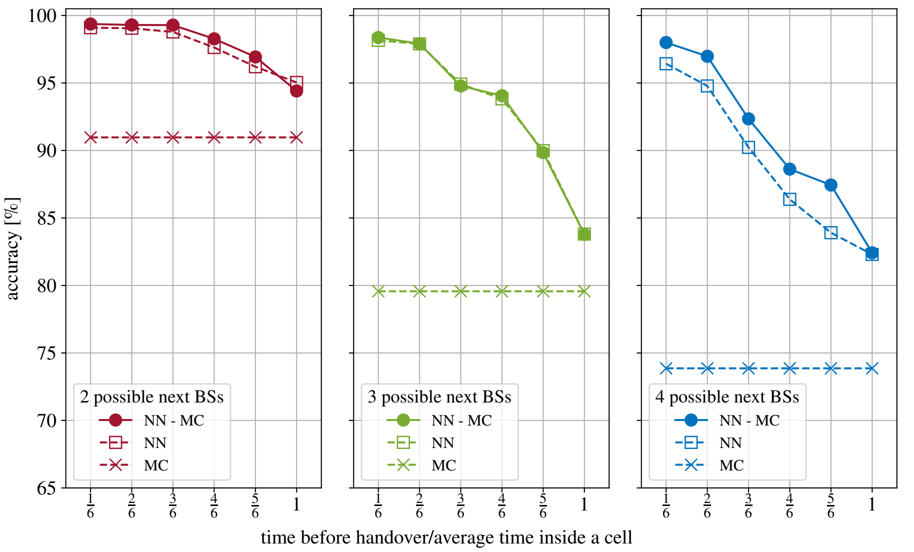

Fig. 4: Prediction accuracy: the x-axis shows the time before the handover occurs, normalized with respect to the average permanence time inside the serving cell.

“Departments of Excellence” (Law 232/2016). The views and opinions expressed in this work are those of the authors and do not necessarily reflect those of the funding institutions.

REFERENCES

[1] A. Osseiran, F. Boccardi, V. Braun, K. Kusume, P. Marsch, M. Maternia, O. Queseth, M. Schellmann, H. Schotten, H. Taoka, H. Tullberg, M. A. Uusitalo, B. Timus, and M. Fallgren, “Scenarios for 5G mobile and wireless communications: the vision of the METIS project,” *IEEE Communications Magazine*, vol. 52, no. 5, pp. 26–35, May 2014.

[2] A. Ghosh, A. Maeder, M. Baker, and D. Chandramouli, “5G evolution: A view on 5G cellular technology beyond 3GPP Release 15,” *IEEE Access*, vol. 7, pp. 127 639–127 651, Sep. 2019.

[3] M. Surridge, “5G innovations for new business opportunities,” Feb. 2017.

[4] M. Koivisto, A. Hakkarainen, M. Costa, P. Kela, K. Leppanen, and M. Valkama, “High-efficiency device positioning and location-aware communications in dense 5G networks,” *IEEE Communications Magazine*, vol. 55, no. 8, pp. 188–195, Aug. 2017.

[5] J. Plachy, Z. Becvar, and E. C. Strinati, “Dynamic resource allocation exploiting mobility prediction in mobile edge computing,” in *Proceedings of the IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sep. 2016.

[6] R. Di Taranto, S. Muppirisetty, R. Raulefs, D. Slock, T. Svensson, and H. Wymeersch, “Location-aware communications for 5G networks: How location information can improve scalability, latency, and robustness of 5G,” *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 102–112, Nov. 2014.

[7] J. Talvitie, T. Levanen, M. Koivisto, T. Ihalainen, K. Pajukoski, M. Renfors, and M. Valkama, “Positioning and location-based beamforming for high speed trains in 5G NR networks,” in *Proceedings of the IEEE Globecom Workshops*, Dec. 2018.

[8] S. Michaelis, N. Piatkowski, and K. Morik, “Predicting next network cell IDs for moving users with discriminative and generative models.” in *Proceedings of the Mobile Data Challenge by Nokia Workshop in Conjunction with International Conference on Pervasive Computing*, Newcastle, UK, Jun. 2012.

[9] A. Hadachi, O. Batrashev, A. Lind, G. Singer, and E. Vainikko, “Cell phone subscribers mobility prediction using enhanced Markov chain algorithm,” in *Proceedings of the IEEE Intelligent Vehicles Symposium*, Jun. 2014.

[10] A. A. Hasbollah, S. H. Ariffin, N. E. Ghazali, K. M. Yusuf, and H. Morino, “Handover algorithm based VLP using mobility prediction database for vehicular network,” *International Journal of Electrical and Computer Engineering*, vol. 8, no. 4, p. 2477, 2018.

[11] D. S. Wickramasuriya, C. A. Perumalla, K. Davaslioglu, and R. D. Gitlin, “Base station prediction and proactive mobility management in virtual cells using recurrent neural networks,” in *Proceedings of the 18th IEEE Wireless and Microwave Technology Conference (WAMICON)*, Cocoa Beach, FL, USA, Apr. 2017.

[12] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, “Recent development and applications of SUMO - Simulation of Urban MObility,” *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[14] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014.

[15] M. F. Rahman, W. Liu, S. B. Suhaim, S. Thirumuruganathan, N. Zhang, and G. Das, “Density based clustering over location based services,” in *Proceedings of the 33rd IEEE International Conference on Data Engineering (ICDE)*, San Diego, CA, USA, Apr. 2017.

[16] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 2011.

[17] M. R. Akdeniz, Y. Liu, M. K. Samimi, S. Sun, S. Rangan, T. S. Rappaport, and E. Erkip, “Millimeter wave channel modeling and cellular capacity evaluation,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1164–1179, Jun. 2014.

[18] *Feasibility study on new services and markets technology enablers (Release 14)*, 3GPP Std., Sep. 2016, tR 22.891 V14.1.0.

[19] *Study on positioning use cases (Release 16)*, 3GPP Std., Sep. 2018, tR 22.872 V16.1.0.

[20] J. A. del Peral-Rosado, R. Raulefs, J. A. López-Salcedo, and G. Seco-Granados, “Survey of cellular mobile radio localization methods: From 1G to 5G,” *IEEE Communications Surveys Tutorials*, vol. 20, no. 2, pp. 1124–1148, Dec. 2018.

[21] E. Björnson, L. Sanguinetti, H. Wymeersch, J. Hoydis, and T. L. Marzetta, “Massive MIMO is a reality - What is next?: Five promising research directions for antenna arrays,” *Digital Signal Processing*, vol. 94, pp. 3–20, Nov. 2019.