On Learning Mobility Patterns in Cellular Networks

J. Sánchez-González, J. Pérez-Romero, R. Agustí, O. Sallent

Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{juansanchez,jorperez, ramon, sallent}@tsc.upc.edu

Abstract.- This paper considers the use of clustering techniques to learn the mobility patterns existing in a cellular network. These patterns are materialized in a database of prototype trajectories obtained after having observed multiple trajectories of mobile users. Both K-means and Self-Organizing Maps (SOM) techniques are assessed. Different applicability areas in the context of Self-Organizing Networks (SON) for 5G are discussed and, in particular, a methodology is proposed for predicting the trajectory of a mobile user.

Keywords: Clustering; Cellular Networks; Mobility patterns.

1   Introduction

The new generation of mobile and wireless systems, known as 5th Generation (5G), intends to provide solutions to the continuously increasing demand for mobile broadband services associated with the massive penetration of wireless equipment while at the same time supporting new use cases associated to customers of new market segments and vertical industries (e.g., e-health, automotive, energy). As a result, the vision of the future 5G Radio Access Network (RAN) corresponds to a highly heterogeneous network with unprecedented requirements in terms of capacity, latency or data rates, as identified in different fora [1][2]. To cope with this heterogeneity and complexity, the RAN planning and optimization processes can benefit at a large extent from exploiting cognitive capabilities that embrace knowledge and intelligence.

In this direction, legacy systems already started the automation in the planning and optimization processes through Self-Organizing Network (SON) functionalities [3]. In 5G, considering also the advent of big data technologies [4], it is envisioned that SON can be further evolved towards a more proactive approach able to exploit the huge amount of data available by a Mobile Network Operator (MNO) and to incorporate additional dimensions coming from the characterization of end-user experience and end-user behavior [5]. Then, SON can be enhanced through Artificial Intelligence (AI)-based tools, able to smartly process input data from the environment and come up with knowledge that can be formalized in terms of models and/or structured metrics that represent the network behavior. This will allow gaining in-depth and detailed knowledge about the whole 5G ecosystem, understanding hidden patterns, data structures and relationships, and using them for a more efficient network management [6].

AI-based SON involves three main stages [6]: (i) the acquisition and preprocessing of input data exploiting the wide variety of available data sources; (ii) the knowledge discovery that smartly processes the input data to come up with exploitable knowledge models that represent the network/user behavior; and (iii) the knowledge exploitation stage that applies the obtained models to drive the decision-making of the SON functions. This paper focuses on the knowledge discovery stage and, in particular, on automatically learning the mobility patterns of the mobile users, trying to identify if the traffic across the cells in a scenario follows specific patterns that can be characterized in terms of prototype trajectories followed by many users.

Different works of the literature have addressed the analysis of trajectories in different contexts such as hurricane trajectories, animal movements, public transportation, etc. Various tools have been considered, such as Self-Organizing Maps (SOM) together with visual analysis [7], density-based clustering [8][9] or Principal Component Analysis [10]. In wireless networks, [11] proposed a trajectory prediction strategy to deal with routing in mesh sensor networks. It is based on clustering similar trajectories followed by wireless nodes and using them for making predictions of other nodes. However, the concept of trajectory in [11] is defined by the set of nodes that a mobile node would associate with to send or receive data along a path, but not by the geographical locations. Instead, in our work we intend to derive a deeper knowledge about trajectories based on analyzing the geographical coordinates. In turn, [12] and [13] address the problem of classifying the trajectory followed by a mobile terminal based on a set of reference trajectories in order to optimize the handover process in LTE. However, while [12] and [13] use a simple method for building the set of reference trajectories, based on monitoring certain users with a given probability and adding their trajectories to the set, in our approach we propose the use of clustering techniques, which are more powerful for identifying the most representative trajectories.

In this context, the approach proposed in this paper considers the use of clustering techniques, namely K-means and SOM, to learn the mobility patterns existing in a cellular network. These patterns are materialized in a database of prototype trajectories obtained after having observed multiple trajectories of mobile users. Different applicability areas for these patterns in the context of 5G-SON are discussed and, in particular, a methodology is proposed for predicting the trajectory of a mobile user.

The rest of the paper is organized as follows. Section 2 describes the proposed methodology based on clustering tools for learning mobility patterns. Section 3 discusses the applicability areas and describes the approach for identifying the trajectory of a mobile user. Proposed approach is evaluated in Section 4, while Section 5 summarizes the concluding remarks.

2   Mobility pattern knowledge discovery

Current cellular networks like 4G already include the capability that the User Equipments (UEs) provide geolocation information, including both geographical coordinates and altitude, as part of the radio measurement reporting processes [14]. Location information can be obtained from UEs in connected mode, who periodically transmit

measurement reports to the network. Furthermore, thanks to the use of Minimization of Drive Tests (MDT) feature [15], UEs in idle mode can log measurements and transmit them later on when the UE enters in connected mode. These capabilities enable MNOs to collect large amounts of data that include valuable knowledge about the spatio-temporal traffic distribution across the cells. This paper proposes a methodology to analyze this data and identify the existing mobility patterns of the UEs.

The approach for learning mobility patterns is graphically illustrated in Fig. 1. It operates on a long-term basis after having observed a large amount of connected and idle mode UEs in different time periods of a certain geographical area and analyzes the collected location information from these UEs to identify the existence of prototype trajectories. As shown in Fig. 1 the first step is the pre-processing, which analyzes consecutive reports for each UE and extracts the geolocation information in order to build a trajectory for this UE. A trajectory is defined here as the concatenation of N coordinates at consecutive time instants $t_1,...,t_N$. Then, assuming for simplicity two-dimensional (2D) coordinates (x,y), the trajectory for the j-th UE is given by the vector of dimension B=2N denoted as $\mathbf{r_j}=[x_j(t_1),y_j(t_1),..., x_j(t_N),y_j(t_N)]$. The result of the pre-processing task will be a total of J trajectories $\mathbf{r_j}$, j=1,...,J.
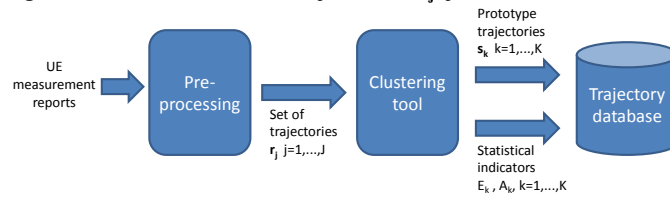
Fig. 1. - Procedure for learning mobility patterns

The second step is the clustering, which processes the set of J trajectories by grouping them in K clusters in a way that trajectories of the same cluster are similar among them and different from the trajectories of the rest of the clusters. Two alternative clustering techniques are considered in this work:

- K-means: This strategy belongs to the family of partitioning methods. It groups the J input trajectories in K clusters by trying to maximize that the similarity between trajectories of the same cluster and to minimize the similarity between trajectories of different clusters, using the Euclidean distance as a metric of similarity. The process can be summarized as follows (see [16] for further details): (a) The algorithm starts by selecting randomly K out of the J input trajectories. Each of these K trajectories represents an initial cluster. For each cluster k, the algorithm computes the centroid $\mathbf{s_k}$. At this initial stage, where each cluster contains only one trajectory, the centroid $\mathbf{s_k}$ equals the selected trajectory for the k-th cluster. (b) Each of the remaining J-K trajectories is assigned to the cluster to which it is the most similar, based on Euclidean distance between the trajectory and the centroid of each cluster $|\mathbf{r_j}-\mathbf{s_k}|$. Once all the J trajectories have been clustered, the new values of the centroids $\mathbf{s_k}$ are recomputed. In particular, the i-th component of $\mathbf{s_k}$ is the average of the i-th components of all the trajectories belonging to the k-th cluster. (c) Using the new values of the centroids $\mathbf{s_k}$, each of the J trajectories $\mathbf{r_j}$ is reassigned to the

cluster with lowest distance $|\mathbf{r_j}\text{-}\mathbf{s_k}|$. The new centroids are recomputed and this step is iteratively repeated until convergence (i.e. until there are no changes in the obtained clusters after two consecutive iterations). (d) At the end of the process, each cluster k=1,...,K will contain a number of input trajectories N_k and its centroid $\mathbf{s_k}$ will be the so-called prototype trajectory that is taken as a representative of all the trajectories belonging to this cluster.

- Self-Organizing Map (SOM): This clustering strategy relies on a neural network model with a total of K neurons and where each neuron is characterized by a B-dimensional weight vector $\mathbf{s_k}$. The process can be summarized as follows (see [17] for details): (a) The weight vectors $\mathbf{s_k}$ are initialized. This can be done randomly or through the linear initialization method described in [17]. (b) An iterative unsupervised learning process is used to update the values of the weight vectors $\mathbf{s_k}$ of the different neurons according to the Kohonen's algorithm [17] based on the input trajectories $\mathbf{r_j}$. In essence, at iteration t the algorithm identifies, for each trajectory $\mathbf{r_j}$ the winning neuron as the one with the lowest Euclidean distance $|\mathbf{r_j} - \mathbf{s_k}|$. Then, the algorithm updates the weight vector of this winning neuron k as $\mathbf{s_k}(t+1)=\mathbf{s_k}(t)+\alpha(t)(\mathbf{r_j} - \mathbf{s_k}(t))$ where $\alpha(t)$ is a scalar-valued adaptation gain that decreases with successive iterations. A similar update is performed for the weight vectors of the rest of neurons k'≠k but in this case the adaptation gain $\alpha(t)$ is multiplied by a neighborhood function that decreases with the distance between neurons k' and k. The process is repeated for a certain number of iterations. (c) At the end of the process, all the input trajectories that have neuron k as winning neuron form the k-th cluster. The number of trajectories in the k-th cluster is N_k, and the prototype trajectory of this cluster is the weight vector $\mathbf{s_k}$.

As shown in Fig. 1, the prototype trajectories obtained as a result of the clustering will be stored in the database. In addition, two statistical indicators are also included for each cluster to assess how representative this cluster is:

- Percentage of hits ($A_k=N_k/J$): It is the percentage of input trajectories that belong to the cluster k. The prototype trajectories of clusters with a high value of A_k will be more frequent and representative of the scenario.
- Average squared Euclidean distance of the trajectories in k-th cluster (E_k): It is a metric that captures the degree of similarity between trajectories of the same cluster with respect to the prototype trajectory $\mathbf{s_k}$ of the cluster. A high value of E_k reflects a higher dispersion in the cluster, meaning that the prototype trajectory is less representative of the clustered trajectories. It is defined as:

$$E_k = \sum_{j \in \text{ Cluster k}} \left| \mathbf{r_j} - \mathbf{s_k} \right|^2 \tag{1}$$

3 Exploitation of mobility patterns

It is envisaged that the identification of prototype trajectories as explained in previous section can have applicability for different 5G-SON functions.

For example, prototype trajectories can be used in the context of self-planning to decide appropriate cell locations and antenna settings. For example, if there is a well identified representative trajectory, a sector of a cell site can be pointed in the direction of this trajectory. Typically, this can be the case of a cell site providing coverage over a main street. Despite one could argue that a radio engineer could easily identify such a situation and take such a common sense decision, the interest of the proposed use case remains in the fact that SON involves automatization. That is, self-planning and self-configuration means the capability for the system to automatically identify the trajectories and propose the adequate values for the parameters of a new cell.

Similarly, the learnt mobility patterns can also have applicability in the self-optimization of several functions such as handover, load balancing or admission control. For example, by identifying the trajectory of a UE or group of UEs in relation to a known prototype trajectory it is possible to anticipate the cell that the UEs are heading to and configure these functions so as to avoid call droppings and overload situations. In the following, we focus on proposing a methodology to predict the future positions of a certain UE based on analyzing the actual locations reported by the UE in relation to the learnt prototype trajectories.

3.1 Mobility prediction

The proposed approach is illustrated in Fig. 2 and is executed on an individual UE basis. The criterion to decide which specific UEs are analyzed is out of the scope of this paper and it will depend on the specific self-optimization function under consideration. For example, the optimization of load balancing may predict the trajectory of UEs that demand a high bit rate in order to anticipate the arrival of these UEs to a cell and take the appropriate actions to ensure there are sufficient resources for these UEs in the cell. Similarly, it is also possible to predict the trajectory of high priority UEs to ensure that they will not experience problems in handovers, etc.

The process of Fig. 2 starts from the measurement reports provided by the UE whose trajectory is being predicted. First, pre-processing stage is carried out to extract the geolocation information and build the trajectory $\mathbf{u}$ that is currently being observed for this UE. The trajectory $\mathbf{u}$ is a vector of dimension $C=2M$ composed by the concatenation of M pairs of coordinates followed by the UE at consecutive time instants $\mathbf{u}=[x(t_1),y(t_1),\ldots, x(t_M),y(t_M)]$. Without loss of generality, let us consider that the dimension of $\mathbf{u}$ is lower than the number of elements of the prototype trajectories $\mathbf{s_k}$ (i.e. $C \leq B$). This reflects that, in case that the UE was following a prototype trajectory, the actual location of the UE is somewhere within the prototype trajectory.

The mobility prediction process of Fig. 2 intends to determine the likelihood that the UE is following one of the learnt prototype trajectories. This is done by assessing the similarity between the trajectory $\mathbf{u}$ followed by the UE and the prototype trajectories $\mathbf{s_k}$ according to the Euclidean distance. Given that $C \leq B$, all the possible portions of C consecutive elements of the vectors $\mathbf{s_k}$ ($k=1,\ldots,K$) need to be considered when assessing this similarity. The α-th portion of $\mathbf{s_k}$ is then defined as the vector $[s_k(1+\alpha),\ldots, s_k(C+\alpha)]$ with $\alpha=0,\ldots,B-C$, where $s_k(i)$ denotes the i-th component of $\mathbf{s_k}$.

Then, the squared Euclidean distance between the α-th portion of s_k and trajectory u is computed as:

$$d_{u,k}(\alpha) = \sum_{c=1}^{C}[u(c) - s_k(c + \alpha)]^2 \qquad \text{with } \alpha=0,\dots,\text{B-C} \qquad (2)$$

Then, the similarity between u and s_k is computed as the minimum Euclidean distance between u and the possible portions of the prototype trajectory s_k, that is:

$$m_k = \min_{\alpha} d_{u,k}(\alpha) \qquad (3)$$

A low value of m_k indicates that the trajectory u is very similar to some portion of vector s_k. Then, the likelihood L_k that the UE is following the prototype trajectory s_k is defined here as:

$$L_k = \frac{1/m_k}{\sum_{k=1}^{K}(1/m_k)} \qquad (4)$$

A high value of L_k reflects that the UE is following a trajectory very similar to a portion of s_k. Therefore, s_k provides information about the positions that the UE may likely follow in the future.
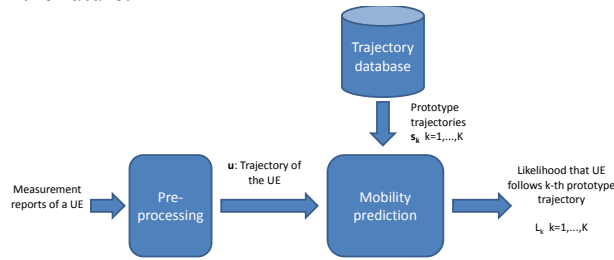

Fig. 2. - Exploitation of learnt patterns for predicting the trajectory of a UE

4   Results

This section provides some results to illustrate the performance of the proposed approach. The considered scenario is shown in Fig. 3 and represents an urban area in the intersection between two main streets. The mobility of multiple UEs has been considered including a wide variety of situations as shown Fig 3a. For example, some UEs move straight along a street, others move straight and turn right, left or move back. For each kind of trajectory, 100 realizations have been generated by considering UE trajectories that are not perfectly straight but they have lateral movements simulating e.g. cars changing the lane in the road. It is assumed that the distance between two consecutive positions of the trajectory is a random value (simulating that the user speed may be variable). Moreover, 100 realizations of users that move a short distance and stop at a particular position (represented by black arrows in Fig. 3a) have been also generated. Finally, a group of 100 static users (represented by black dots in Fig 3a) have also been placed randomly in each of the four corners of the scenario.

After the preprocessing of the UE measurements, there are a total of J=2100 trajectories. Each trajectory $\mathbf{r_j}$ consists on N=40 positions.

(a)   (b)
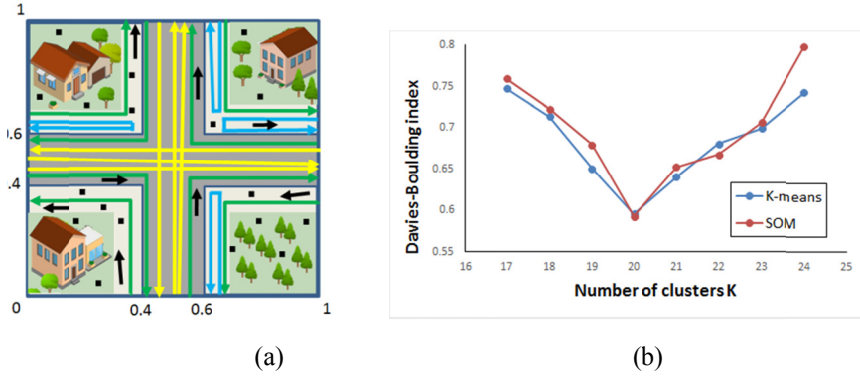
Fig. 3. (a) Illustration of the considered scenario. Distances are normalized between 0 and 1. (b) Davies-Bouldin index for different numbers of clusters.

4.1 Clustering process

The K-means and SOM clustering techniques have been implemented by means of RapidMiner Studio [18]. The K-means algorithm is configured with 1000 runs and a maximum of 100 iterations for each run (i.e. the process explained in section 2 is repeated 1000 times with different initial random selections, and the best result among all runs is kept). In turn, a SOM with one dimension is configured with 10000 iterations and initial adaptation rate equal to 0.1 and final adaptation rate 0.01. The neighborhood function is defined by an initial adaptation radius of 2 and a final adaptation radius equal to 0.01.

First, the impact of the number of clusters K has been analyzed for both K-means and SOM techniques. The Davies-Bouldin index [19] is considered as a relevant metric to assess the quality of the clustering process. This index takes into account how similar are all the trajectories that belong to the same cluster and how different are the prototype trajectories of the different clusters. Low values of the Davies-Bouldin index reflect a better quality of the clustering process. Fig. 3b presents the Davies-Bouldin index as a function of the number of clusters for both K-means and SOM methodologies. As shown, for the considered use case, the minimum value of the Davies-Bouldin index is observed with K=20 clusters for both methodologies. For this case, Fig. 4 illustrates the prototype trajectories $\mathbf{s_k}$ obtained by the K-means methodology. The same prototypes are obtained by the SOM methodology with K=20. The red point marked in each prototype trajectory in Fig. 4 indicates the initial position of the trajectory while the black point indicates its final position (e.g. the prototype trajectory 1 represents a user moving from the left to the right while the prototype trajectory 2 represents a user moving from the right to the left). Note that some shorter prototype trajectories represent users who move on a specific direction and then go back (e.g. prototype trajectory 13 represents to users who move from the left to the right in the scenario, and then go back from the right to the left). Other

prototype trajectories, such as prototype 17, represent the centroid of some static users located around this area.
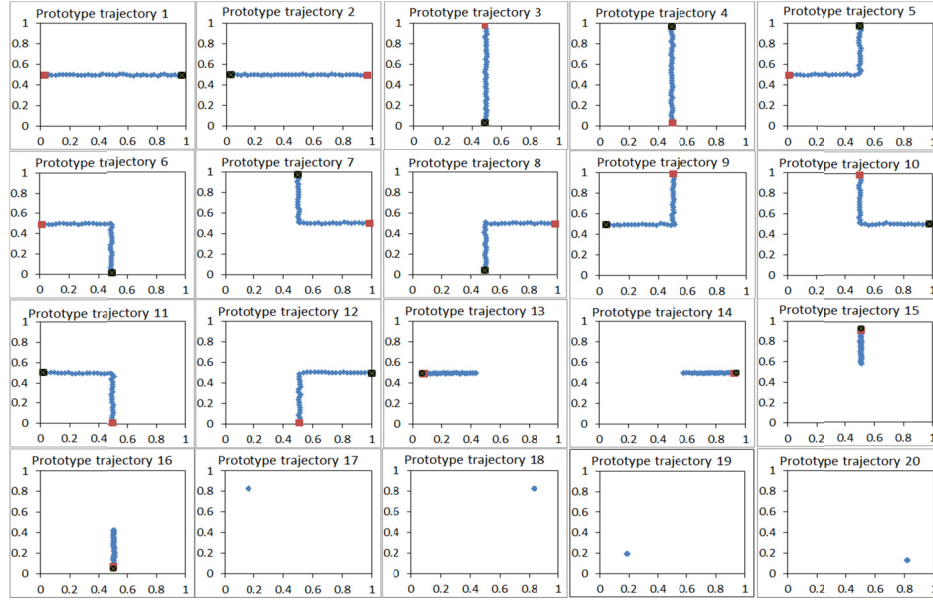
Fig. 4. Prototype trajectories obtained with K-means (K=20). Horizontal and vertical axes represent normalized distances between 0 and 1.

Fig. 5a illustrates the percentage of hits A_k for each cluster with both K-means and SOM, while Fig. 5b represents the average squared Euclidean distance E_k, All the clusters corresponding to long trajectories (i.e. clusters 1 to 12 of Fig. 4) exhibit low E_k. This indicates that these trajectories are well-clustered and their corresponding prototype trajectories are good representatives of the cluster. In turn, clusters 13, 14, 15 and 16 of Fig 4 include users that move straight and go back, users that move short distances and even some static users. As a consequence, higher percentage of hits A_k and higher values of E_k are observed. Finally, clusters 17, 18, 19 and 20 of Fig. 4 are formed by static users scattered around the four corners of the scenario. These are characterized by high values of E_k, meaning that some static users of these clusters may be located at a relatively high distance of the centroid. A very similar clustering is done by both K-means and SOM methodologies as shown in Fig. 5a and Fig. 5b.

4.2 Mobility prediction

This section presents several examples to illustrate the behavior of proposed mobility prediction approach. Fig. 6 presents the trajectories followed by four different UEs. UE A has a trajectory that consists of 20 positions representing a movement from the left of the scenario to the right. UE B has a trajectory of 20 positions moving straight and then turning in the intersection. UE C has a shorter trajectory of 10 positions while UE D is static and contains 10 samples of the same position.
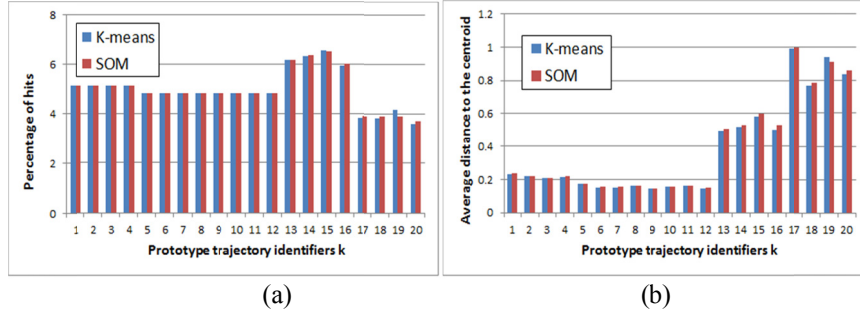
Fig. 5. (a) Percentage of hits A_k for the different clusters; (b) Average square Euclidean distance to the centroid E_k for the different clusters.

Fig. 7 shows the likelihood L_k that each of the four UEs is following each prototype trajectory. As shown, the likelihood that UE A is following prototype trajectory 1 is almost 100%. As seen in Fig. 5, this prototype trajectory corresponds to the users that move from the left to the right. Similarly, for UE B there also is a very high likelihood that it is following prototype trajectory 5. For UE C, the likelihood L_1, L_5 and L_6 are similar, because, with the trajectory followed by UE C so far, it may correspond to either trajectories 1, 5 or 6. Finally, trajectory D is not similar to any of the prototypes obtained in the clustering process (see Fig. 4). For this reason, the prediction process provides a very low likelihood for all the clusters.
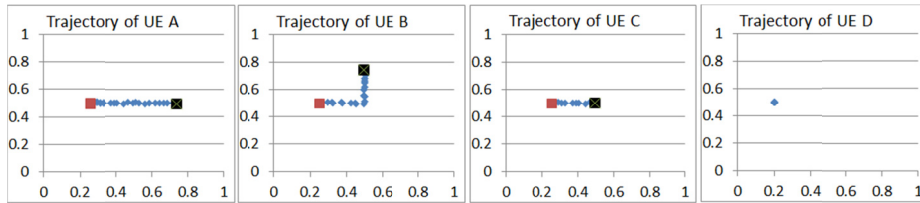

Fig. 6. Example of UEs' trajectories. Horizontal and vertical axes represent normalized distances between 0 and 1.
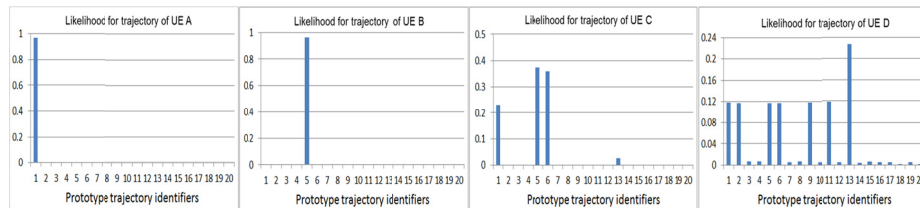

Fig. 7. Likelihood L_k that the UEs are following the learnt prototype trajectories.

5   Concluding remarks

The paper has proposed a methodology for learning mobility patterns in wireless networks based on clustering techniques such as K-means and SOM. Learnt trajectories

present applicability in different areas, such as self-planning and self-optimization. In this respect, the paper has proposed a strategy for predicting the mobility of specific users based on the obtained prototype trajectories. Results reflect that both K-means and SOM techniques are able to properly identify the different trajectories existing in the considered scenario.

References

1. METIS 2020 project, http://www.metis2020.com
2. R. El Hattachi, J. Erfanian (ed.) "NGMN 5G White Paper", NGMN Alliance, Feb., 2015
3. J. Ramiro, K. Hamied, *Self-Organizing Networks. Self-planning, self-optimization and self-healing for GSM, UMTS and LTE*, John Wiley & Sons, 2012
4. C-L I., Y. Liu, S. Han, S. Wang, G. Liu, "On Big Data Analytics for Greener and Softer RAN", IEEE Access, August, 2015.
5. A. Imran, A. Zoha, A. Abu-Dayya, "Challenges in 5G: How to Empower SON with Big Data for Enabling 5G", IEEE Network, November/December, 2014, pp. 27-33.
6. J. Pérez-Romero, O. Sallent, R. Ferrús, R. Agustí "Artificial Intelligence-based 5G Network Capacity Planning and Operation", ISWCS conference, 2015.
7. T. Schreck et al. "Visual Cluster Analysis of Trajectory Data With Interactive Kohonen Maps", IEEE Symp. on Visual Anal. Science and Techn., October, 2008, Columbus, USA.
8. J-G. Lee, J. Han, K-Y. Whang, "Trajectory Clustering: A Partition-and-Group Framework", SIGMOD, 2007, China.
9. G. Andrienko, et al. "Interactive Visual Clustering of Large Collections of Trajectories", IEEE Symp. on Visual Anal. Science and Techn., October, 2009, Atlantic City, USA,
10. E. Masciari, "A Complete Framework for Clustering Trajectories", 21st IEEE International Conference on Tools with Artificial Intelligence, 2009.
11. H. J. Lee et al, "Data Stashing: Energy-efficient Information Delivery to Mobile Sinks through Trajectory Prediction", ACM/IEEE IPSN conference, 2010.
12. B. Sas, K. Spaey, C. Blondia, "Classifying Users based on their Mobility Behavior in LTE networks", 10th Int. Conf. on Wireless and Mob. Comms. (ICWMC), 2014.
13. B. Sas, K. Spaey, C. Blondia, "A SON function for Steering Users in Multi-Layer LTE Networks based on their mobility behavior", VTC spring conference, 2015.
14. 3GPP TS 36.331 v12.7.0, "Radio Resource Control (RRC); Protocol Specification (Release 12)", September, 2015.
15. W. A. Hapsari, A. Umesh, M. Iwamura, M. Tomala, B. Gyula, B. Sébire, "Minimization of Drive Tests Solution in 3GPP", IEEE Communications Magazine, June, 2012.
16. J. Han, M. Kamber, *Data Mining Concepts and Techniques*, 2nd edition, Elsevier, 2006.
17. T. Kohonen, "Essentials of the Self-Organizing Map", Neural Networks, Elsevier, Vol. 37, 2013, pp. 52-65.
18. RapidMiner Studio, http://www.rapidminer.com
19. Davies, D. L.; Bouldin, D. W. "A Cluster Separation Measure". *IEEE Transactions on Pattern Analysis and Machine Intelligence.* Vol: PAMI-1, issue 2: pp: 224–227, April 1979. doi:10.1109/TPAMI.1979.4766909.