Handover Forecasting in Cellular Networks: A Spatio-temporal Graph Neural Network Approach

Gwladys Ornella Djuikom Foka, Razvan Stanica, Diala Naboulsi

Handover Forecasting in Cellular Networks: A Spatio-temporal Graph Neural Network Approach

Gwladys Ornella Djuikom Foka
Software and IT Engineering Department
École de Technologie Supérieure (ÉTS)
Montreal, Canada
gwladys-ornella.djuikom-foka.1@ens.etsmtl.ca

Razvan Stanica
Telecommunications Department
Univ Lyon, INSA Lyon, Inria, CITI
Villeurbanne, France
razvan.stanica@insa-lyon.fr

Diala Naboulsi
Software and IT Engineering Department
École de Technologie Supérieure (ÉTS)
Montreal, Canada
diala.naboulsi@etsmtl.ca

***Abstract*—The proliferation of connected devices, as well as the continuous integration of data demanding applications, have brought significant challenges to mobile network operators. Indeed, mobile networks need to accommodate the resulting exponential growth of traffic, while still ensuring a decent Quality of Service (QoS) and Quality of Experience (QoE) to their users. To do so, adequate management solutions are needed. So far, a variety of problems have been tackled, including resource allocation and network planning. Yet, handovers (HO) management has not received much attention, despite its importance for users QoS and QoE. In this work, we study the problem of mobile devices HO forecasting, over time, among base stations. To solve the problem, we introduce a Handover Graph Convolutional Long Short-Term Memory (HGC-LSTM) neural network forecasting approach. Our approach allows capturing spatio-temporal correlations among neighboring pairs of base stations in the forecasting process. The evaluation on a real-world dataset shows that the proposed HGC-LSTM approach outperforms other methods in the state-of-art in terms of accuracy and execution time.**

***Index Terms*—Handovers Forecasting, 5G and beyond, Time Series Forecasting, Machine Learning**

I. Introduction

By the end of 2028, 5 billion 5G subscriptions are forecasted in the world [1]. With more and more data-hungry applications, mobile network data traffic has been skyrocketing, namely doubling in the past two years. As a result, mobile operators are facing a remarkable surge in data traffic, leading to significant pressure on their infrastructures. Indeed, the proliferation of intelligent devices, coupled with data hungry applications, are pushing for greater network capacity [2], [3]. Operators are thus continuously seeking for appropriate solutions, that can help accommodate for the resulting traffic, while guaranteeing a decent quality of service (QoS) and quality of experience (QoE) to their customers.

In this context, different problems have been investigated, including resource allocation and network planning, for future 5G and beyond networks. However, handover (HO) management has not received as much attention, despite the impact it has on users QoS and QoE. Indeed, a HO refers to the process of transferring an ongoing user communication session from one Base Station (BS) to another, while minimizing disruptions perceived by the user [4]. Hence, HO management represents one of the key aspects to handle in order to preserve users' QoS and QoE and maintain a high level of users satisfaction, as they move. With the growing number of connected devices, relying on very diverse applications and with drastic increase in data traffic, HO management is becoming more and more challenging, especially for the mobile operators, because of the high number of HOs on one hand, and the HO overhead in terms of signaling and resources requirements on the other hand [2].

The HO management process includes itself several components, namely HO prediction [5], HO triggering [6], HO decision [7], HO optimization [8] and HO evaluation [9]. Our work is focused on HO forecasting, a key aspect in the HO management process, allowing for more efficient resource (e.g. bandwidth, power, processing capacity) allocation schemes that can minimize operational costs and lead to improved scalability and load balancing. In addition, it is beneficial for network planning, including network optimization and dynamic configuration of the base stations.

In our work, we are interested in studying the problem of mobile devices HO forecasting, over time, between every pair of neighboring base stations, in a certain geographical area. In this context, most existing contributions focus on the prediction of the next BS (or cell) where a HO will occur [10], except for two previous studies [5], [11] that focus on forecasting the number of HOs. In [5], the authors group cells that have similar HO behavior in clusters, and then they forecast the number of HOs attempts for each cluster. However, this study does not consider the prediction of HOs between BSs or cells, as we do, but rather the prediction of the number of HO attempts per cell. On the other hand, in [11], the authors forecast the number of HOs and traffic between every pair of neighboring BSs, over time, with a multivariate forecasting approach, for resource allocation in Cloud-Radio Access Networks (C-RANs). This work is very similar to ours, yet in our case, we adopt a novel approach, based on Graph Neural Networks (GNN), that can capture spatio-temporal correlations among neighboring pairs of base stations, in the forecasting process.

The contributions of this paper are twofold:

This work was supported by the National Natural Sciences and Engineering Research Council of Canada (NSERC) through research grant RGPIN-2020-06050 and by the Fonds de Recherche du Québec – Nature et Technologies (FRQNT) through research grant 2022-NC-297403.

- We propose a Handover Graph Convolutional Long Short-Term Memory (HGC-LSTM) neural network forecasting approach, that relies on a GNN architecture, combined with an LSTM model, to forecast the number of HOs, between every pair of neighboring BSs, over time. The GNN architecture allows extracting the spatio-temporal dependencies between the different pairs of base stations, and to achieve better forecasting performance. It further allows modeling complex relationships and captures non-linear dependencies, which are useful for the forecasting task. The LSTM model, well-known for its excellent results on time series data [12], further helps achieve a more accurate forecasting.
- We evaluate the performance of our HGC-LSTM solution on a real-world HO dataset, and compare its performance to other existing forecasting solutions. The results show that our solution outperforms the baseline solutions, while maintaining a decent ratio between the execution time and the accuracy metric. We also shed light on the impact of LSTM parameters on the performance of our solution, and further investigate its performance at times with peaks of HOs.

The rest of the paper is organized as follows. Section II examines previous studies on forecasting in networks in general, and handover forecasting in particular, while Section III presents the problem statement and the corresponding system model. In Section IV, we elaborate on the design of our proposed solution, which covers the methodology and the framework architecture. Section V focuses on the evaluation of our approach. Finally, Section VI concludes this work.

II. Related Works

Existing works on HO forecasting follow one of two axes: the prediction of the next HO target cell (or BS) for a given mobile user, and the prediction of the number of HOs between cells (or BSs), over time.

Regarding the first axis, authors in [13] focused on the HO decision, in order to forecast the best next cell for a given user equipment (UE) even before the moment when a HO needs to be executed. To do so, they proposed a hybrid HO forecasting mechanism, which contains long-term and short-term forecasting models, as well as a fuzzy forecasting model to deal with imprecise data. Their results showed the proposed method decreases the rate of HO failure. Going on the same lane, researchers in [14] also focused on the HO decision and used the HO information to predict the next (target) cell to avoid lag during HOs. There, the authors predict the next target BS based on the history of serving and target BSs during the HO process. For the prediction itself, they used the frequency of previous HOs between pairs of BSs. The simulation showed that the number of neighboring BSs influences the prediction. The authors in [10] target the same problem as in the previously discussed studies, but argue that sophisticated techniques are not necessary for HO prediction in multi-access edge computing (MEC) systems. Therefore, they proposed probabilistic approaches to predict the future cell. Thus, they used forecasting strategies based on the existence of global knowledge and individual profiling and showed that they lead to a high prediction accuracy.

The second axis, in which our contribution fits, includes a lower number of studies. Researchers in [5] propose a method to perform the forecasting of the number of HOs aggregated per cell. They unveil common HO patterns by analyzing cells behavior and propose a clustering approach (K-means) to group cells accordingly. They then used linear regression (LR), neural networks (NN), polynomial regression (PR) and Gaussian processes (GP) to forecast hourly HOs per cell. The evaluation shows that the four algorithms can accurately forecast HOs, with GP presenting the best performance. Speaking of clustering-based frameworks, the authors in [15] propose a three-phased data clustering framework (using a generative adversarial network (GAN) for clustering, cluster calibration, and cluster division) to improve the performance of HO forecasting. With that, LSTM is used to forecast the number of hourly HOs occurrences, per cell, in a specific city. The results show that, not only does this method allow preserving the privacy of the users, but also leads to improvement in HO forecasting. As we can see, the previous papers are only focused on the forecasting at a per-cell level. This is not the case for [11], where authors perform the forecasting of the number of HOs between every pair of neighboring Radio Remote Heads (RRHs) connected to the same Baseband Units (BBUs), as part of a CRAN architecture, based on historical data. For this aim, they propose a Multivariate LSTM model (MuLSTM) to forecast the number of HOs for a future period of time. The results show that MuLSTM is better than Auto-Regressive Integrated Moving Average (ARIMA) and Windowed Artificial Neural Network (WANN) methods.

Besides the studies discussed above, some contributions exploit the number of HOs for mobile traffic forecasting, such as in [16]. There, the authors proposed a spatio-temporal dynamic graph network (SDGNet) solution, based on a GNN approach, to perform mobile traffic forecasting. In their work, the authors model the cellular network as a graph where nodes are the BSs, with edges linking them. The edges are weighted by HO frequency between every pair of BSs. Their results show excellent performance in terms of long-term prediction and time complexity.

In a nutshell, we observed that most of the works on the HO forecasting topic focus on the prediction of the next cell or BS rather than the forecasting of the number of HOs. Considering the latter aspect, the majority of the contributions are focused on the forecasting of the number of HOs per cell, which is different from our problem, in which we target the forecasting of the number of HOs between pairs of neighboring BSs. The closest to our work remains the study in [11], aiming to forecast HOs among pairs of neighboring RRHs. However, in our case, we propose a more advanced technique, relying on a GNN architecture, capable of accurately capturing spatio-temporal dependencies between the different pairs and leading to a better performance, on regular and complex HO patterns.

III. System Model and Problem Statement

In this section, we present the problem statement and the system model used for the handover forecasting problem.

A. System model

We consider an urban area that includes a set of deployed BSs, and we use p to refer to a pair of BSs between which HOs occur. Each pair is associated to a time series H_p^T which represents the evolution of the number of handovers between the pair p over time, up until a timestamp T. We use P to refer to the set of all pairs, such that $p \in P$.

As depicted in Figure 1, we model our network as an undirected graph $G = (P, E)$, where $P = \{p_1, ..., p_N\}$ is the set of nodes denoting the $N = |P|$ pairs of BSs, and E is the set of edges. Considering the scenario in Figure 1, we define the nodes p_1, p_2, p_3 and p_4 as the pairs (BS-A, BS-B), (BS-B, BS-C), (BS-C, BS-D), (BS-D, BS-A) where handovers occur respectively. There is an edge between two nodes, p_i and p_j, if there is at least one base station in common between them, and the weight of the edge is $w_{i,j} = 1$.

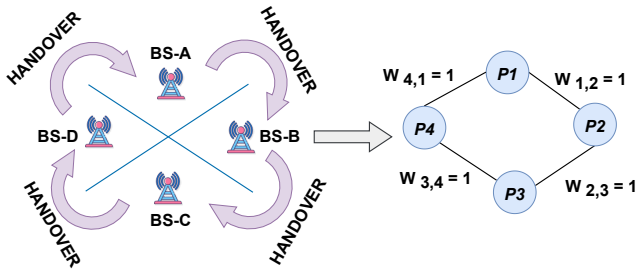


Fig. 1. Graph modeling.

B. Problem statement

Considering the system model, we study the problem of HO forecasting for each pair of BSs. We define $H_p^T = \{h_p^{T-t}, ..., h_p^T\}$ as a time series, representing the evolution of the number of HOs for the pair p between time slot $T-t$ and time slot T. Accordingly, h_p^{T-t} represents the number of HOs for pair p at time slot $T-t$ and h_p^T represents the number of HOs for pair p at time slot T. Given H_p^T, our goal is to forecast h_p^{T+1}, the number of HOs for pair p, in time slot $T+1$.

IV. Framework Design

To address the above-stated problem, we will describe in this section the methodology we used to build our proposed solution, followed by the architecture of the framework.

A. Methodology

The proposed HGC-LSTM model combines GNNs and LSTM to predict the number of handovers between pairs of base stations. Inspired by [17], our architecture includes two main components: the Graph Convolutional Network (GCN) layer and the LSTM layer.

The first component allows to have the nodes representations which are obtained by passing our dataset through the GCN layer. This layer aims to capture the local and global dependencies in the graph, by aggregating information from neighboring nodes through a message passing mechanism. The GCN layer learns representations that encode both the node local features and its structural context within the graph for the forecasting. The message passing is performed as follows:

- **Neighbor's embeddings collection:** the representation of each node is computed based on the connected neighbors of that node.
- **Neighbors aggregation:** the neighbors of each node are aggregated through an average function to have aggregated messages as the output.
- **Node representation update:** the final output is computed by passing the initial node representation and the aggregated messages to a dense neural network layer.

The second component, namely the LSTM layer, will be used solely for the forecasting of the time series associated to each node. In this stage, we are considering the corresponding adjacency matrix $A^{'}(G,t)$ of our graph and the nodes representations $H^{'}(G,t)$, to pass them as an input to the LSTM layer. It is worth mentioning that t represents the time steps in the past considered for the prediction.

B. Framework architecture

In the following, we farther describe each component, starting with the GCN part, and followed by the LSTM part.

1) GCN block: GCN is a fundamental building block in GNNs, enabling aggregation across the nodes in the graph, and as already mentioned, message passing is a key component in GCN. For each node p in the graph, we consider the following steps to compute the nodes representations, as in Figure 2:

- **Initialization**: Each node p is associated to $p_{i \in \{1,...n\}}$ neighbors, where n represents the number of neighbors. The node and its neighbors are assigned features $H_{(p,t)}$ which represent the historical data of length t, that will be considered for the prediction. Basically, for each pair p, we consider t points in the past for the future predictions.
- **Aggregation**: In this step, we compute the aggregated messages p^{agg} of p neighbors.
- **Update**: The final representation $H^{'}_{(p,t)}$ is computed using the aggregated messages p^{agg} and the initial representation of the node p^{init}.

We thus obtain as the output, the nodes representations over time including for each nodes, the information about its neighbors. The update step usually consists in applying an activation function to the final representation of the node. However, for our problem, the activation function will be directly included in the LSTM layer.

2) LSTM layer: From the previous component, we take the representation of nodes for each time step, which will be considered as the input for the LSTM layer in order to

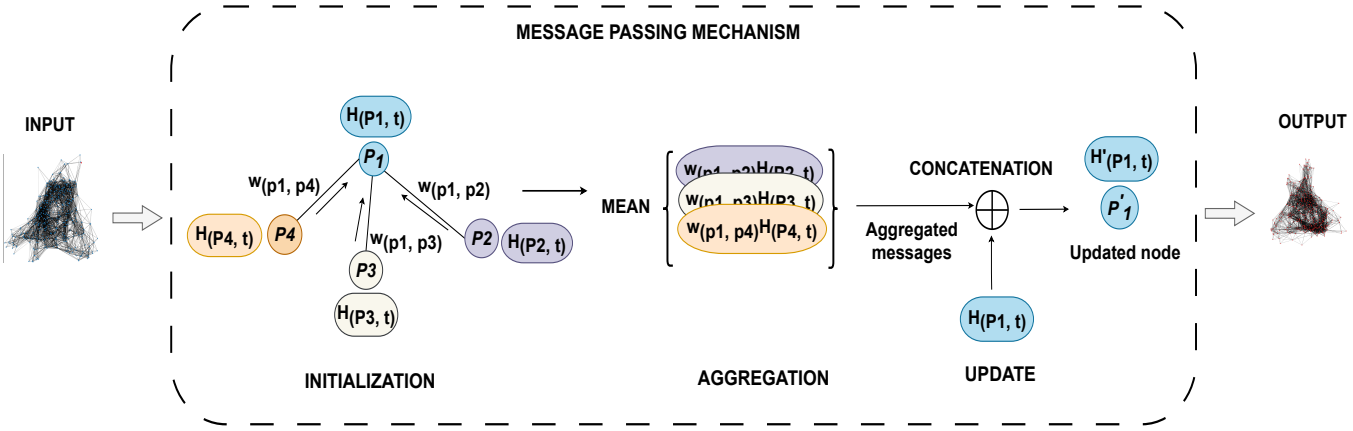


Fig. 2. GCN block.

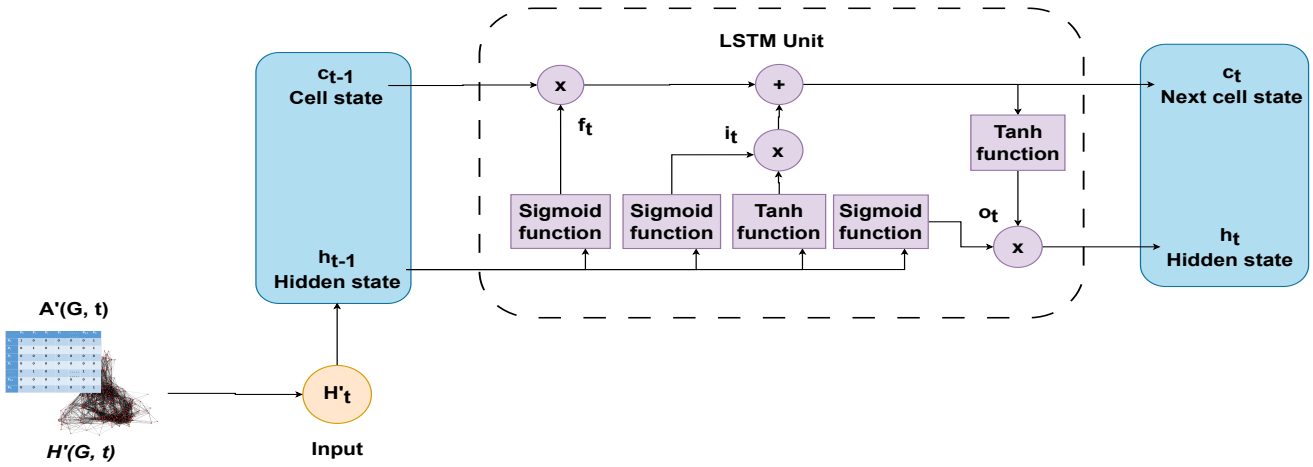


Fig. 3. LSTM block.

process the information over time. So, after applying the graph convolutional layer, we pass the result through a LSTM network as shown in Figure 3, to capture the spatio-temporal dependencies of our data. For the sake of simplicity, we denote as $H_t^{'}$ the representation of each pair p over time. The LSTM network processes the information of the input sequence to make predictions for future time steps and each unit of the network includes the following components:

- **Initialization and input processing:** the hidden state $h_{(t-1)}$ and the cell state $c_{(t-1)}$ are initialized to capture the memory of the LSTM cell in order to propagate information across time steps.
- **Input gate i_t:** this takes the input and the hidden state to past them through a sigmoid activation function to determine if the new information should be added to the current state.
- **Forget gate f_t:** it decides what information to discard by passing the input value and the previous hidden state through a sigmoid function.
- **Cell state update:** it consists in combining the new information from the input gate and the forget gate f_t to determine the new value of the cell state c_t.
- **Output gate o_t:** this determines the output based on the previous hidden state h_{t-1}, the updated cell c_t and the current input $H_t^{'}$.
- **Prediction:** the predicted value is the hidden state h_t, which represents the output of the LSTM network and is obtained by combining the cell state c_t and the output gate o_t.

V. Evaluation

In this section, we present the evaluation of our approach. To do so, we will first present the dataset that we used for the experiments, as well as our experimental settings. Then we will compare the performance of our approach against other methods and discuss the obtained results.

A. Dataset

We evaluate our HGC-LSTM using a real-world dataset collected from 59 4G base stations of the Orange network

in Poitiers, France. The locations of the 59 cellular sites are shown in Figure 4.

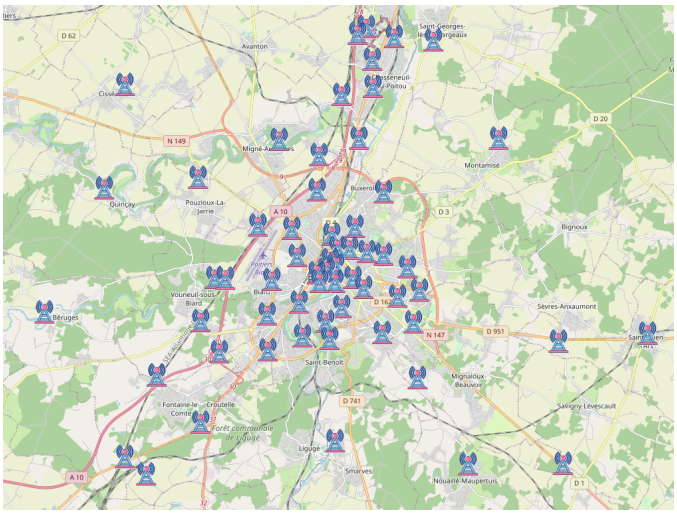


Fig. 4. 4G site locations in Poitiers, France.

The dataset includes the records of the number of HOs between pairs of BSs, with a granularity of 30 minutes, from March 15th, 2019, to April 16th, 2019. The dataset includes in total information about 537 pairs of BSs where HOs occur, and we constructed the graph in Figure 5 based on that.

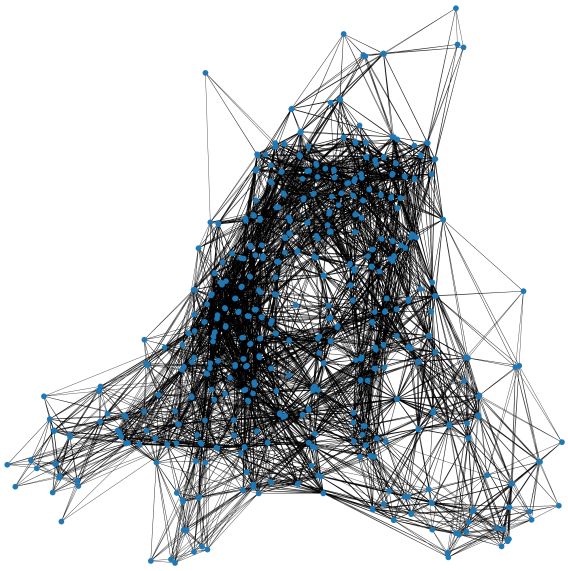


Fig. 5. HGC-LSTM Graph.

B. Benchmarking

We compare the performance of our proposed HGC-LSTM model to two other approaches: AutoRegressive Integrated Moving Average (ARIMA) [18], a classical time series approach, and Multivariate LSTM (MuLSTM), applied in [11], the closest work to ours, where the authors study the problem of HO forecasting between pairs of RRHs over time, in a C-RAN architecture.

ARIMA is a popular statistical univariate model used to analyze and forecast single time series data, while focusing on modeling the behavior and patterns of a single variable over time. It combines autoregressive (AR), differencing (I), and moving average (MA) components to analyze and predict time series data. The AR component captures the linear relationship between an observation and lagged observations, the I component removes trends through differencing, and the MA component models the dependency on residual errors [19], [20]. The model is defined by three parameters: a, d, and q. Parameter a represents the autoregressive order, d represents the differencing order, and q represents the moving average order [21]. These parameters are determined based on the characteristics of the time series data and can be estimated using statistical techniques. Since ARIMA is not designed for multivariate forecasting, we built a personalized model for each pair of BSs with HOs to be predicted, knowing that each model used in this approach has its own parameters, which are different based on the chosen pair.

MuLSTM on the other hand is a neural network used to process and predict a feature (variable) which depends on multiple other features. This model is designed to capture dependencies and patterns across multiple variables in a time series dataset [22]. The LSTM architecture provides the ability to capture long-term dependencies and handle sequential data effectively, as it is designed to remember information for long periods of time [23]. A LSTM cell consists of three components also known as gates: input gate, forget gate and output gate. We implemented and applied this algorithm to our problem, considering one model for each pair of BSs. Indeed, to forecast the time series associated to a specific pair of BSs, we used the historical data of that pair and its neighbors considering the graph that we defined earlier.

We rely on the Root Mean Square Error (RMSE) of the predicted values with respect to the ground truth values in the dataset, which is a widely used metric to assess the performances of the above-mentioned algorithms. We divide the dataset and use 70% for training and 20% for testing in the case of ARIMA and MuLSTM, whereas we have 70%, 10%, 20% for training, validation and test sets respectively for HGC-LSTM. The predictions are performed on the test set for each algorithm. The best model would be the one leading to the lowest RMSE values.

C. Experimental settings

In this section, we are presenting the different configurations that we considered when running the experiments.

Our framework variables are described in Table I. This table includes the parameters for the GCN components and the LSTM network. We highlight that we considered one model per pair of BSs for ARIMA, with different parameters for each one. For MuLSTM, we also considered one model per pair,

and considered the same batch size as HGC-LSTM and 50 LSTM units.

Variable	Description	Value
N	Number of nodes (pairs)	537
E	Number of edges	21098
$batch$	Batch size (number of time series samples in each batch)	64
$inputs$	Number of input features	1
$ouputs$	Number of output features	1
t	Number of time steps considered for the prediction	4
$epochs$	Number of epochs	300
$units$	Number of LSTM units	64

TABLE I
FRAMEWORK VARIABLES.

Our HGC-LSTM model is built using the TensorFlow-GPU open source machine learning library, and we mainly used PySpark to preprocess the data, and the PyCharm Integrated Development Environment (IDE) for coding. All the algorithms are trained on the Canada national high-performance compute (HPC) infrastructure called Compute Canada, which is a national platform of advanced research computing resources. The device configuration used in all the experiments is Intel Xeon Silver 4110 Skylake 2.10 GHz and Intel Xeon Gold 6238 Cascade Lake 2.10 GHz as the CPUs, the graphical processor is Tesla T4, and the memory is 3.20 GB.

D. Results

For this section, we will first compare the performances of our algorithm with the chosen baselines and after that, we will conduct an analysis of the relevant outcomes of HGC-LSTM on its own.

1) RMSEs analysis: We show, in Figure 6, the global average of the RMSEs for the three models computed for the 537 pairs of BSs. As we can see, HGC-LSTM has the lowest average RMSE, which means that its overall performance is better than ARIMA and MuLSTM.

In order to have a better visualization of the forecasting performance achieved by our framework and the baselines, in Figure 7, we arbitrarily chose four pairs to plot the predicted number of HOs on the test data, for HGC-LSTM, ARIMA and MuLSTM. The figure also suggests that HGC-LSTM achieves better performance than the two other methods.

Considering the previously selected pairs, we can see a more quantitative comparison through the individual RMSEs depicted in Figure 8, where the results of HGC-LSTM are not too far overall from MuLSTM in terms of performance on these examples. However, we noticed that MuLSTM and ARIMA perform poorly at predicting time series where the seasonal pattern does not include many points. The ability of HGC-LSTM to capture more complex patterns, due to the GNN part which allows the model to do so, is the reason why it performs better overall.

2) Execution Time: From the perspective of the computation complexity, as we can see in Table II, HGC-LSTM

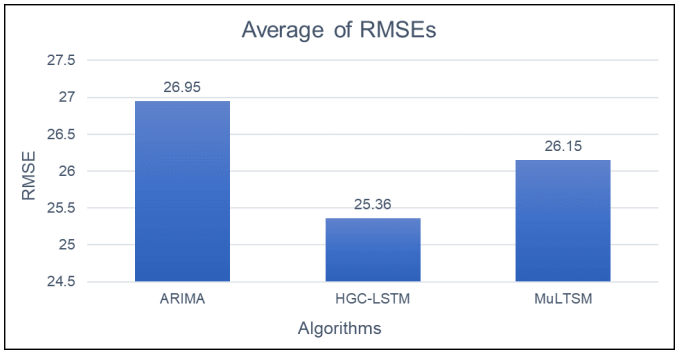


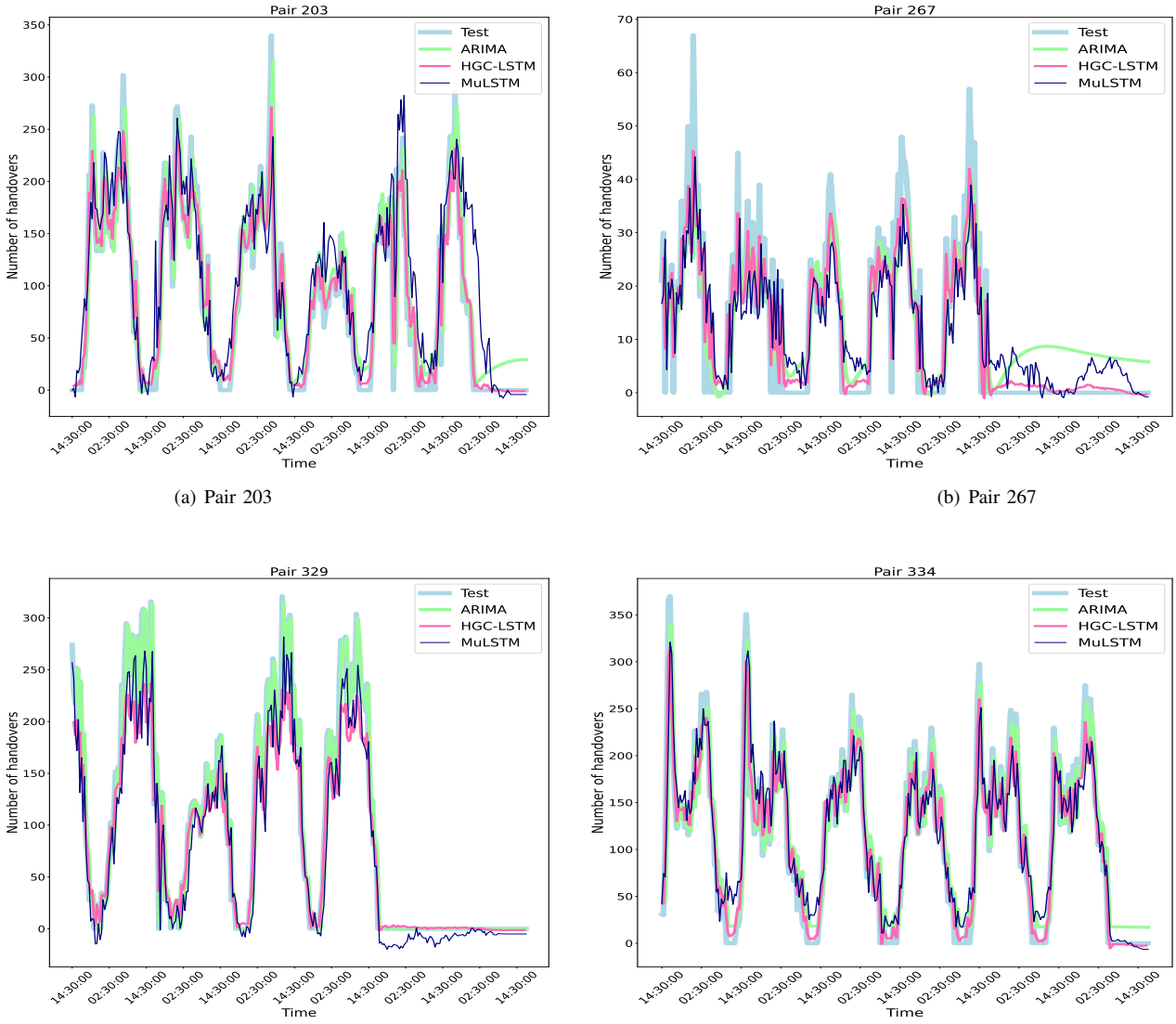
Fig. 6. Average RMSEs of ARIMA, HGC-LSTM and MuLSTM.

outperforms ARIMA and MuLSTM with an execution time of 775.97 seconds. We also noticed that ARIMA performs poorly from this point of view because its execution time is high. This can be explained by the fact that ARIMA itself requires building one model per pair of BS, with its own set of parameters that need to be automatically defined. Defining these parameters comes at the cost of an important additional computational overhead. Considering that we have more than 500 time series, the whole process of finding the parameters, training the model, and conducting the forecasting, had to be done more than 500 times. This causes ARIMA to take much longer execution time than the others. As for MuLSTM, although we have one model per pair of BSs as well, it does not require a high execution time. This can be explained by the fact that in our configuration settings, to run all the algorithms, we used the Tesla T4 GPU which is optimized for training workloads, as it comes with dedicated tensor cores for deep learning operations, leading to faster computations when we manipulate neural networks [24], [25], as we do in MuLSTM and HGC-LSTM.

Algorithms	Execution time (seconds)
$ARIMA$	312422
$HGC-LSTM$	775.97
$MuLSTM$	6276

TABLE II
FRAMEWORK VARIABLES

3) LSTM units: With regard to the analysis of HGC-LSTM itself, we initially investigated the influence of the number of LSTM units on the quality of the prediction. As we can see in Figure 9, there is a remarkable decrease of the average RMSE as we go from 4 to 8 units, followed by a slower decrease as we go to larger number of units, despite a local peak that is observed for 200 units. Moreover, we can notice that 500 units gives the best results in terms of RMSE (24.97) and 4 units gives the worst results (26.37). We can conclude that the number of units of the LSTM model has a significant impact on the performance of our HGC-LSTM framework, and it should be properly evaluated for different datasets.



(a) Pair 203

(b) Pair 267

(c) Pair 329

(d) Pair 334

Fig. 7. Forecasting for the pairs 203, 267, 329, and 334 with HGC-LSTM, ARIMA and MuLSTM.

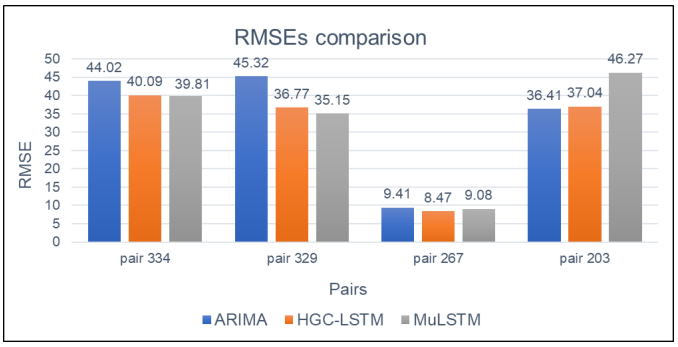


Fig. 8. RMSE values per pair for ARIMA, HGC-LSTM and MuLSTM.

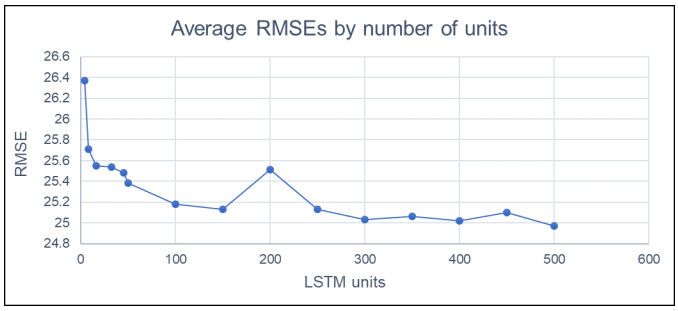


Fig. 9. Average RMSEs by number of units for HGC-LSTM.

4) Peaks analysis: We also investigated the performance of HGC-LSTM on the prediction of peak values of HOs in our data. This metric is particularly important from the point of view of a mobile operator, as peak moments are the most

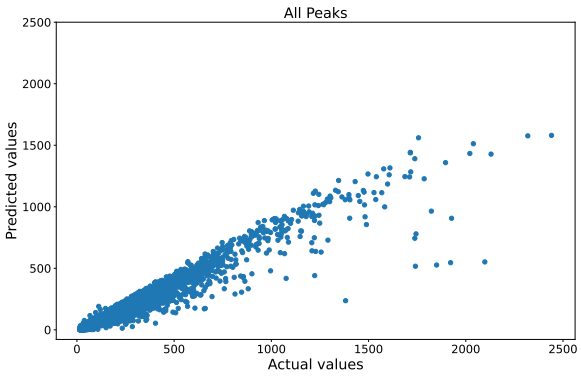


Fig. 10. Peaks distribution.

important to manage properly. We defined a peak as a point in the time series where the k values before and after that point in the series are lower than that point. We considered $k = 4$ for this analysis, and Figure 10 depicts the distribution of the predicted and actual values of the peaks. We can observe that, in the very large majority of cases, the predicted values are smaller than the actual values. In addition, for higher peaks, HGC-LSTM has a hard time performing accurate predictions.

We also computed the error percentage on all the peaks of the network using the following formula:

$$error(\%) = ((prediction - real)/real) \cdot 100,$$

where $prediction$ is the predicted value of the peak and $real$ is the real value of the peak. By computing the global error (mean of all the errors) we have 37.55%, which means that globally the model fails in the prediction at this percentage when the network experiences a very high HO rate, reflecting a high level of mobility.

Furthermore, we analyzed the probability distribution of the error according to the different peaks by using the Cumulative Distribution Function (CDF). The CDF is defined by :

$$F(x) = P(X \leq x),$$

where X is a random variable, x represents our $error$ value, and the CDF $F(x)$ calculates the probability that the random variable X is less or equal to x. Considering the absolute values of the error percentage, like in Figure 11(a), we notice that the probability to have an error percentage less than 50% is higher for all the data points than just the peaks. This means that the percentage error can be much higher on peaks than on other data points.

On the other hand, we show in Figure 11(b) the CDF of the error, with both positive and negative errors. We can notice that almost all errors on the peaks present a negative value, meaning that the HGC-LSTM approach is almost always underestimating the values at the peaks. The percentage of negative errors drops on the other hand to 60%, as we consider all data points. Accordingly, despite the fact that we underestimate the values in the majority of cases, we still overestimate them in 40% of the cases.

VI. Conclusion

This paper addresses the problem of the forecasting of the number of HOs between pairs of BSs, over time. To this end, we propose HGC-LSTM, a GNN-based framework combined with LSTM to capture the spatio-temporal dependencies and complex patterns within the data. Our approach was built and evaluated with a real-world HO dataset on all the 4G base stations of the city of Poitiers. We compared our results to two baselines, ARIMA and MuLSTM, which are well-known algorithms in time series forecasting, and the experiments show that our model outperforms these state-of-art methods in terms of global RMSEs. We also demonstrated that HGC-LSTM is more efficient considering the execution time. Furthermore, we showed that increasing the number of LSTM units improves the accuracy of the forecasting to a certain extent. Since this contribution is directed to networking applications, we also investigated the forecasting of peaks of HOs in the day, where we have a lot of mobility. Despite the fact that the solution succeeds in predicting complex patterns, we noticed that the accuracy is diminished when it comes to predicting very high values. As future works, we are planning to leverage the outcomes of this contribution for optimization applications in RAN architectures to enable better resource allocation.

References

[1] Ericsson, "Ericsson Mobility Report November 2022," no. November, 2022.

[2] M. S. Mollel, A. I. Abubakar, M. Ozturk, S. F. Kaijage, M. Kisangiri, S. Hussain, M. A. Imran, and Q. H. Abbasi, "A Survey of Machine Learning Applications to Handover Management in 5G and beyond," *IEEE Access*, vol. 9, pp. 45 770–45 802, 2021.

[3] M. Ozturk, "Cognitive networking for next generation of cellular communication systems," Ph.D. dissertation, University of Glasgow, 05 2020.

[4] A. Goldsmith, *Wireless Communications*, 1st ed. Cambridge, UK: Cambridge University Press, 2005.

[5] L. L. Vy, L. P. Tung, and B. S. P. Lin, "Big data and machine learning driven handover management and forecasting," *2017 IEEE Conference on Standards for Communications and Networking, CSCN 2017*, pp. 214–219, 2017.

[6] P. Bellavista, C.-M. Chen, and H. Hassanein, "Editorial: Special issue on service delivery management in broadband networks," *J. Netw. Comput. Appl.*, vol. 35, no. 5, p. 1375–1376, sep 2012. [Online]. Available: https://doi.org/10.1016/j.jnca.2012.05.004

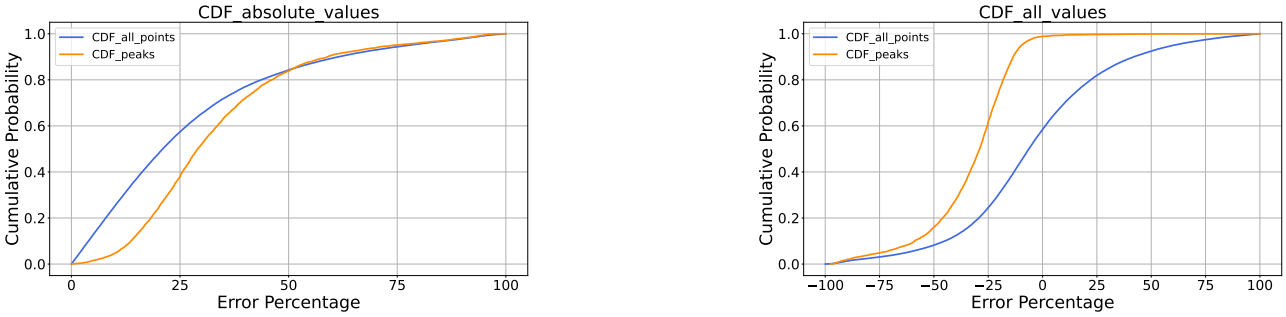
[7] D. Xenakis, N. Passas, L. Merakos, and C. Verikoukis, "Mobility management for femtocells in lte-advanced: Key aspects and survey of handover decision algorithms," *IEEE Communications surveys & tutorials*, vol. 16, no. 1, pp. 64–91, 2013.

[8] K. Kitagawa, T. Komine, T. Yamamoto, and S. Konishi, "A handover optimization algorithm with mobility robustness for lte systems," in *2011 IEEE 22nd international symposium on personal, indoor and mobile radio communications*. IEEE, 2011, pp. 1647–1651.

[9] N. Montavont and T. Noel, "Handover management for mobile nodes in ipv6 networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 38–43, 2002.

[10] H. Abdah, J. P. Barraca, and R. L. Aguiar, "Handover prediction integrated with service migration in 5g systems," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–7.

[11] L. Chen, T.-M.-T. Nguyen, D. Yang, M. Nogueira, C. Wang, and D. Zhang, "Data-Driven C-RAN Optimization Exploiting Traffic and Mobility Dynamics of Mobile Users," *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1773–1788, 2021.



(a) Absolute values

(b) All values

Fig. 11. Probability distribution of the error percentage

[12] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile traffic prediction from raw data using lstm networks," in *IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2018.

[13] H. Qu, Y. Zhang, J. Zhao, G. Ren, and W. Wang, "A hybrid handover forecasting mechanism based on fuzzy forecasting model in cellular networks," *China Communications*, vol. 15, no. 6, pp. 84–97, 2018.

[14] Z. B. , "Efficiency of Handover Prediction Based on Handover History," *Journal of Convergence Information Technology*, vol. 4, no. 4, pp. 41–47, 2009.

[15] Y. Kim, E. A. Hakim, J. Haraldson, H. Eriksson, J. M. B. da Silva, and C. Fischione, "Dynamic clustering in federated learning," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6.

[16] Y. Fang, S. Ergüt, and P. Patras, "Sdgnet: A handover-aware spatiotemporal graph neural network for mobile traffic forecasting," *IEEE Communications Letters*, vol. 26, no. 3, pp. 582–586, 2022.

[17] Z. He, C. Zhao, and Y. Huang, "Multivariate Time Series Deep Spatiotemporal Forecasting with Graph Neural Network," *Applied Sciences*, vol. 12, no. 11, p. 5731, 2022.

[18] M. Elsaraiti, G. Ali, H. Musbah, A. Merabet, and T. Little, "Time series analysis of electricity consumption forecasting using arima model," in *2021 IEEE Green Technologies Conference (GreenTech)*, 2021, pp. 259–262.

[19] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. Wiley, 2015.

[20] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*. OTexts, 2018.

[21] J. D. Hamilton, *Time Series Analysis*. Princeton University Press, 1994.

[22] X. Li, W. Zhang, and Z. Xu, "A multivariate time series lstm model for short-term load forecasting," in *2017 3rd International Conference on Systems and Informatics (ICSAI)*, 2017, pp. 852–856.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] "Nvidia tesla t4," NVIDIA Website, accessed 2023. [Online]. Available: https://www.nvidia.com/en-gb/data-center/tesla-t4/

[25] "Tensorflow," Website, accessed 2023. [Online]. Available: https://www.tensorflow.org/