Predicting short-term mobile Internet traffic from Internet activity using recurrent neural networks

Guto Leoni Santos1 | Pierangelo Rosati2 | Theo Lynn2 | Judith Kelner1 | Djamel Sadok1 | Patricia Takako Endo3

1Centro de Informática, Universidade Federal de Pernambuco, Pernambuco, Brazil
2Business School, Dublin City University, Dublin, Ireland
3Universidade de Pernambuco, Pernambuco, Brazil

Correspondence
Guto L. Santos, Centro de Informática, Universidade Federal de Pernambuco, Pernambuco, Brazil.
Email: gls4@cin.ufpe.br

Funding information
Fundação de Amparo a Ciência e Tecnologia de Pernambuco (FACEPE), Grant/Award Number: IBPG-0059-1.03/19

Summary
Mobile network traffic prediction is an important input into network capacity planning and optimization. Existing approaches may lack the speed and computational complexity to account for bursting, non-linear patterns, or other important correlations in time series mobile network data. We compare the performance of two deep learning (DL) architectures, long short-term memory (LSTM) and gated recurrent unit (GRU), and two conventional machine learning (ML) architectures—Random Forest and Decision Tree—for predicting mobile Internet traffic using 2 months of Telecom Italia data for Milan. K-Means clustering was used a priori to group cells based on Internet activity, and the Grid Search method was used to identify the best configurations for each model. The predictive quality of the models was evaluated using root mean squared error and mean absolute error. Both DL algorithms were effective in modeling Internet activity and seasonality, both within days and across 2 months. We find variations in performance across clusters within the city. Overall, the DL models outperformed the conventional ML models, and the LSTM outperformed the GRU in our experiments.

1 | INTRODUCTION

At the beginning of 2020, it was estimated that 67% of the global population (5.2 billion people) subscribed to mobile services.1 According to industry forecasts, we are about to enter into a period of unprecedented mobile data growth driven by the Internet of Things (IoT). According to Cisco,2 by 2023, machine-to-machine (M2M) connections that support a broad range of IoT applications will represent about 50% (14.7 billion) of total global devices and connections. By that time, 45% of all networked devices will be mobile-connected. The combination of this influx of new mobile-connected devices, faster broadband speeds, greater video consumption, and the capabilities of 5G networks is dramatically increasing mobile data traffic. The nature and scale of this growth poses significant challenges to mobile network providers including the management of complexity, scalability, Quality of Service (QoS), Quality of Experience (QoE), and privacy.3 Constrained budgets and intense competition exacerbate these challenges.

This massive surge in demand for mobile broadband requires solutions that satisfy QoS and QoE requirements with minimum service delay and within budget constraints. Capacity planning is the process of adjusting the capacity across the network in response to changing or predicted demands.4 Mobile network traffic modeling and prediction is an important input into multiple capacity planning tasks including network design, performance evaluation, control, and network optimization.5,6 Network traffic is a time series with non-linear and chaotic characteristics and is correlated over both long and short time frames.6 There is a well-established literature that focuses on trends, seasonality, and
anomaly prediction to guide mobile network investments and optimization, both at the network-level and the cell-
level. Much of the existing research focuses on traffic prediction across the network that, while presenting good fore-
cast results, may have unacceptable training and turnaround times, lack computational complexity, and consequently
may not account for characteristics such as bursting, non-linear patterns, or other important correlations.

Addressing this shortcoming requires moving from coarse-grained to fine-grained prediction and the adoption of
novel techniques that can accommodate high dimensionality and provide a satisfactory solution quickly. In the last
decade, we have seen a number of applications that can benefit from short-term or fine mobile network traffic prediction
including opportunistic scheduling, multimedia optimization, and energy efficiency. More recently, the emer-
gence and combination of deep learning (DL) techniques and new data sources are presenting promising results in
mobile traffic prediction.

This paper compares the performance of two recurrent neural networks (RNNs), long short-term memory (LSTM)
and gated recurrent unit (GRU), against two traditional machine learning (ML) techniques, Random Forest, and Deci-
sion Tree, for predicting mobile Internet traffic. In this paper, the terms Internet activity and Internet traffic are used
interchangeably. Our main contribution is the evaluation of a methodology to create clusters of cells based on statistical
traffic measurements. First, cells with similar traffic patterns are grouped together, and then prediction models are cre-
ated to forecast the average traffic for each cluster, since the cells of a given cluster have a similar pattern. By using this
methodology, we can reduce the number of models, which improves efficiency and, as a result, reduces the computa-
tional and economic cost and complexity associated with training thousands of models.

Using 2 months (62 days) of spatio-temporal log data from Telecom Italia for the metropolitan area of Milan, we
select 12 clusters each with its own time series. We find the best configuration of each model using the Grid Search
method and use root mean squared error (RMSE) and mean absolute error (MAE) to evaluate the performance of the
models. Based on the results, we can state that the models satisfactorily learned the pattern of Internet activity and sea-
onality, both within days and across the 2 months. We find variations in performances based on geographic clusters
within the city. Overall, the LSTM performs better than the GRU in most cases. We also compare the MAE results for
our models with extant research using similar data and find both of our DL models significantly outperform previous
work based on Random Forest and Decision Tree models.

The rest of this paper is structured as follows. In Section 2, we provide an overview of RNNs and clustering algo-
rithms. Section 3 presents selected related research. Section 4 details the mobile Internet traffic dataset and data
preprocessing, the configuration of the LSTM and GRU models, and the metrics used to evaluate model performance.
Section 5 presents and discusses our results and analysis. We conclude this article with a brief summary of the main
contributions of the article and propose future directions for research in Section 6.

2 | BACKGROUND

2.1 | DL neural networks

In the last 5 years, deep neural networks, “deep learning,” have increased in prominence in research and practice. DL
is a sub-branch of ML that, at a high level, “enables an algorithm to make predictions, classifications, or decisions based
on data, without being explicitly programmed.” DL addresses the limitations of single-layer neural networks by using
multiple layers to transform their input into higher dimensional representations and then into the output. The emer-
gence of DL is largely driven by increased computational power through heterogeneous processors such as GPUs, the
availability of large datasets for training, and advances in optimization algorithms.

In contrast to traditional ML and neural networks, DL can cater for high dimensionality in data thus enabling DL
networks to model highly complex non-linear relationships between variables. As such, it is particularly suitable to
the mobile and wireless networking domain which is characterized by massive volumes of high velocity unlabeled het-
erogeneous data. In addition, DL can significantly reduce operational and capital expenditure by reducing or eliminat-
ing the time and effort required by valuable and scarce human resources in feature extraction and reduce
computational and memory requirements through multi-task learning. However, DL is not without its limitations. It
hides its internal logic to the user thereby sacrificing accuracy for interpretability with practical and ethical conse-
quences. Other limitations include vulnerabilities to adversarial and privacy attacks, computational demands unsuitable to small-form computing in edge networks, and the time taken to find optimal configurations, particularly
for highly parameterized data and multi-step network prediction.
Common DL architectures include multilayer perceptron (MLP), restricted Boltzmann machines (RBM), auto-encoded (AE), convolutional neural networks (CNNs), and RNNs. These can be differentiated by the data structures that they target and their respective tuning parameters. For example, MLP targets feature vectors of fixed length and are tuned by the activation function setting and the number of layers and units. In contrast, CNNs target high-dimensional data with local dependencies and are tuned by the number and width of convolutional kernels or filters. Because both MLP and CNN assume that all inputs are independent of each other, they are not suited to modeling sequential data, where sequential correlations exist between samples. RNNs specifically target sequential data, like time series data flows from mobile networks. As such, we focus on the use of RNNs in this paper.

2.2 | Recurrent neural networks

2.2.1 | Long short-term memory

As discussed earlier, traditional ML, MLPs, and CNNs typically target input vectors with fixed dimensions. The formalization for sequential data is fundamentally different, and thus, MLP and CNN are not suitable for time series data. RNN architectures were specifically designed to model sequential data by producing output via recurrent connections (cells) between hidden units. These recurrent connections are the memory that stores the previous data allowing RNNs to learn the temporal dynamicity of the sequential data. In RNNs, the output of the current timestamp is influenced by the output of the previous timestamps, which is critical when the sequence of events or data is important to determine the outcome of a problem. Despite being designed to model sequential data, early RNNs suffer from long-time dependencies resulting from both vanishing and exploding gradient problems that negatively impacted training using the back-propagation through time (BPTT) algorithm. To overcome this limitation, a variation of traditional RNNs was proposed, LSTM, which introduces the concept of gates to mitigate gradient problems. Figure 1 presents the basic schema of an LSTM unit.

The unit takes as input three values: the input of the current time step \( x_t \), the output from a previous LSTM unit \( h_{t-1} \), and the “memory” of the previous unit \( C_{t-1} \). The output of the LSTM unit are two values: the output and the memory of the current unit, \( h_t \) and \( C_t \) respectively.

**Figure 1** Example of an LSTM unit (adapted from Yan)
The LSTM unit state updates through specific gate operations: write (input gate), read (output gate), or reset (forget gate). These operations consist of component-wise multiplications and apply different functions in the input data.

Initially, the input data, the output from previous LSTM unit, and a bias are the input for a simple one-layer neural network where the activation function is sigmoid. This operation is detailed in Equation (1).

\[
f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}C_{t-1} + b_f)
\]  

where \(W_{xf}, W_{hf}, \text{and } W_{cf}\) are, respectively, the weight matrices of the forget gate for the input \(x\), hidden state \(h\), and the unit state \(c\). \(f_t\) can be defined as a “forget valve,” and its output vector will be applied to the old memory, \(C_{t-1}\) by element-wise multiplication, as illustrated in Figure 1.

The same inputs of \(f_t\) are also passed to a second neural network with one layer but with different weight matrices:

\[
i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}C_{t-1} + b_i)
\]  

where \(i_t\) controls how much the old memory will influence the new memory of the LSTM unit. However, the new memory is generated by another neural network with one layer but a hyperbolic tangent as the activation function:

\[
g_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)
\]  

Then, \(i_t\) and \(g_t\) are element-wise multiplied and summed to the old memory to compose the new unit memory, \(C_t\), as shown in Equation (4). \(C_t\) is one of the output values of the LSTM unit, as shown in Figure 1.

\[
C_t = f_tC_{t-1} + i_tg_t
\]  

Finally, the unit output, \(h_t\), is impacted by the new memory, the previous output \(h_{t-1}\), and the input \(X_t\). \(o_t\) is a neural network similar to \(i_t\) and \(f_t\), with one layer and sigmoid as activation function. It receives as input \(X_t\) and \(h_{t-1}\). As shown in Equation (5), the tangent hyperbolic activation function is applied in \(C_t\) and then element-wise multiplied by \(o_t\). \(h_t\) controls how much new memory should output to the next LSTM unit.

\[
h_t = o_t\tanh(c_t)
\]  

While LSTM addresses gradient problems, critics have noted that the LSTM architecture is ad hoc, has a substantial number of components whose purpose is not immediately apparent, and that it is characterized by long training times.\(^{23,26}\)

Given its ability to model time series data and predictive capacity, we use LSTM in this study.

### 2.2.2 Gated recurrent unit

GRU is a variation of LSTM which only uses two gates, an update gate and a reset gate. Indeed, the update gate in a GRU replaces the input and forget gates used in LSTM and decides what input data will be kept.\(^{27}\) Furthermore and unlike LSTM, GRU exposes its memory content at each step balancing between the previous and new memory content.\(^{28}\)

The GRU activation, \(h_t^j\), is represented in Equation (6). Considering the input data as a time series, at the timestamp \(t\), \(h_t^j\) is the linear interpolation between the previous unit data \((h_{t-1}^j)\) and the current data \((\tilde{h}_t^j)\).

\[
h_t^j = (1 - z_t^j)h_{t-1}^j + z_t^j\tilde{h}_t^j
\]  

where \(z_t^j\) is the output gate and defines what should be forgotten and what should be kept in the GRU unit. The output gate is defined in Equation (7):\(^{28}\)
\[ z_t^j = \sigma(W_zx_t + U_zh_{t-1}) \] (7)

where the current and previous weight matrices are \( W_z \) and \( U_z \), respectively. In a simplified way, this procedure taking a linear sum between the previous hidden states \( h_{t-1} \) and the current input \( x_t \) and applies the sigmoid function \( \sigma \).

The new memory unit is calculated as described in Equation (8):

\[ \hat{h}_t = \text{tanh}(Wx_t + r_t \odot Uh_{t-1}) \] (8)

where \( \odot \) is the element-wise multiplication, \( r_t \) refers to reset gate, and its output can be calculated as defined in Equation (9):

\[ r_t = \sigma(W_rx_t + U_rh_{t-1}) \] (9)

Research suggests that GRUs are easier to generalize with faster training times while achieving comparable performance outcomes.\textsuperscript{23,28–30} As such, we also propose a GRU for comparison against LSTM and ML models in this study.

### 2.3 Clustering

Clustering is an unsupervised ML technique that breaks down a dataset into disjointed groups (clusters) of observation with similar characteristics.\textsuperscript{31} There is a variety of clustering techniques;\textsuperscript{32} \( K \)-means clustering is the most widely used technique. The \( K \)-means algorithm is a partitioning method which groups unlabeled data into a predefined \( k \) number of clusters based on the Euclidean distance between different data vectors.\textsuperscript{33}

The \( K \)-means algorithm starts by randomly selecting a \( k \) number of centroids; it then calculates the distance between a data vector and each centroid and assigns the vector to the cluster with the closest centroid as outlined in Equation (10).\textsuperscript{33}

\[ S_i^{(t)} = \left\{ x_p : \| x_p - \mu_i^{(t)} \|^2 \leq \| x_p - \mu_j^{(t)} \|^2 \ \forall j, 1 \leq j \leq k \right\} \] (10)

where \( S \) is the distance between the element and the centroid, \( i \) is the cluster number, \( t \) is the number of iterations, \( x_p \) is point value, \( \mu_i^{(t)} \) is the value of centroid, and \( j \) is the measure of dissimilarity.

Every time a new data vector is assigned to a cluster, the \( K \)-means algorithm calculates a new centroid based on the average of distances of all data points in each cluster as presented in Equation (11). This iterative process is completed when all data vectors have been allocated to different clusters and the ultimate centroids are identified.

\[ \mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j \] (11)

We are seeking to predict mobile Internet traffic across a city comprising multiple cells. Traffic behavior may vary across different cells, and the behavior of one cell is not necessarily similar to its neighboring cell. As such, we propose a methodology based on cell clustering to predict the Internet traffic using \( K \)-means clustering.

### 3 RELATED WORKS

Predicting traffic for the next day, hour, or even the next minute can be used to optimize the available system resources, for example by reducing the energy consumption, applying opportunistic scheduling, or preventing problems in the infrastructure.\textsuperscript{8}
Zhang et al. proposed a CNN model that was able to capture the spatial dependency and two temporal dependencies, closeness and period. Prediction matched the ground truth trend well, and the peaks of both incoming and outgoing traffic was effectively captured and predicted compared to three existing algorithms—historical average value, autoregressive integrated moving average (ARIMA), and LSTM. Huang et al. propose a multi-task learning architecture with three different types of DL models including LSTM, three-dimensional CNN, and a combination of CNN and RNN (CNN-RNN) to model spatial and temporal aspects of the traffic. They also compare the performance against ARIMA and non-DL methods. The CNN-RNN model was found to be reliable for all tasks with 70% to 80% forecasting accuracy. Chen et al. propose a two-phase framework to dynamically find optimal remote radio head (RRH) clustering and baseband unit (BBU) mapping schemes under different contexts. A multivariate LSTM was used to learn the temporal dependency and spatial correlation among base station traffic patterns and to make accurate traffic forecasts for future time periods. The prediction output was used to create RRH clusters and map them to BBU pools to maximize the average BBU capacity utility and minimize the overall deployment cost. Results suggested that the proposed method increased the average capacity utility and reduced the overall deployment cost outperforms baseline methods, that is, distance-constrained complementarity-aware (DCCA) ARIMA, DCCA windowed artificial neural networks (DCCA-WANN), and multivariate distance-constrained LSTM. Wang et al. propose a hybrid DL model for spatio-temporal prediction comprising of an AE-based model for spatial modeling, and LSTM for temporal modeling. Results suggest that the proposed hybrid model significantly improves prediction accuracy compared to ARIMA and support vector regression (SVR). In Alawe et al. an LSTM and a deep neural network model were compared for predicting traffic load on a 5G network to inform a scalability mechanism. Performance was evaluated using simulation and suggests that the forecast-based scalability mechanism outperformed threshold-based solutions.

Unlike the other Telecom Italia studies cited above, Zhang and Patras focus on reliable long-term mobile data traffic forecasting. They propose an ensemble system that leverages convolutional LSTM and 3D-ConvNets structures to model long-term trends and short-term variations of the mobile traffic volume, respectively. Results suggest that the proposed system provided highly accurate long term (10-h long) traffic predictions, while operating with short observation intervals (2 h), irrespective of the time of day. The ensemble system outperformed baseline methods including Holt Winters (HW), ARIMA, MLP, and support vector machine (SVM).

Wang et al. propose a novel decomposition of in-cell and inter-cell urban data traffic and apply a graph-based neural network (GNN) using LSTM to accurately predict mobile traffic. They compare their proposed GNN with a number of baseline methods including NAIVE, ARIMA, LSTM, HW, and variations of their GNN. Results suggest that the proposed DL variant of their GNN consistently and significantly outperformed all the baselines in both MAE and mean absolute relative error (MARE). Feng et al. propose an LSTM-based end-to-end model (DeepTP) to forecast traffic demands from spatial-dependent and long-period cellular traffic. DeepTP outperforms ARIMA, SVR, and GRU (although to a lesser extent) based on MRSE. While outperforming other models, DeepTP was much slower than other models including GRU. Qiu et al. also use LSTM combined with unified multi-task learning frameworks to explore spatio-temporal correlations among base stations to improve traffic prediction. They evaluate their proposal against online SVR, non-parametric regression, the adaptive Kalman filters, and an AE approach. The proposed LSTM approach outperforms other methods using MSE as a performance metric.

Even though a number of studies have already tried to address the challenges posed by mobile traffic prediction, our work proposes a different methodology. While other studies have clustered cells based on their geographical location, in this study, we group them based on traffic similarity in order to mitigate traffic variation and therefore improve traffic prediction. In addition, we test and compare two different DL approaches and associated configurations to identify the most effective model for traffic prediction. We also compare these DL models with Random Forest and Decision Tree, which are traditional ML models used for time series forecast.

4 | MATERIAL AND METHODS

4.1 | Dataset

The metropolitan area of Milan is located in northern Italy and consists of nine different municipalities (see Figure 2). Milan is the largest metropolitan area in Italy and one of the 10 most populous in the European Union.

In this study, we use the Telecom Italia dataset for Milan from the Big Data Challenge. The dataset is organized into 10,000 cells (100 × 100) comprising over 10 million user activity logs, each related to a particular cell.
The dataset has log data for 2 months (62 days) from November 1, 2013 to January 1, 2014. Although this dataset was collected between 2013 and 2014, it still proves to be quite valuable for researchers exploring mobile traffic prediction, and it has been used in a number of recently published articles (see, e.g., other studies). The Telecom Italia dataset adopted in this study in fact is one of the few telecommunication datasets that are publicly available in contrast to the large number of datasets that are typically accessible to a restricted number of researchers under non-disclosure agreements (NDAs), or by third parties that have a contractual relationship with telecommunication providers.

The log activity is structured as call detail records (CDRs) on the following activities: (i) incoming and outgoing voice calls, (ii) short message service (SMS) messages, and (iii) Internet activity. A CDR is generated every time a user starts or finishes a voice call, sends or receives an SMS, and starts or terminates an Internet session (the data are recorded if the connection takes more than 15 min, or more than 5 MB is transferred during the session). In this paper, we specifically focus on predicting Internet traffic.

As the dataset has periods with no Internet traffic (e.g., a few minutes at night where there are no records), we aggregate all CDRs for Internet traffic into 30-min periods. Consequently, we have 48 records per day related to Internet traffic. We use a sliding window strategy with a window size of four time periods; thus, we use the previous 2 h to predict the Internet activity of the subsequent 30 min.

To create the training and testing datasets, the original dataset is divided in two parts: the first 80% of the time series for training and the last 20% for testing. We also normalized these datasets to the [0,1] range to facilitate the training of DL models, since their parameters are very small, close to zero.

4.2 Traffic prediction pipeline

Analyzing the traffic of the cells present in the Telecom Italia dataset, we noted that there are different traffic patterns. For example, Figure 3A,B shows the traffic of the Cells 1 and 1000, respectively. The traffic of these cells is quite different. Cell 1 has some peaks at the beginning of the time series, with the biggest peak at middle of November 2013; in December, the traffic is more stable. On the other hand, Cell 1000 traffic has regular peaks across November and December. At the end of December, Cell 1000 traffic drops; this does not happen in any other period of the time series, nor in Cell 1 traffic.

The traffic of these cells reinforces the complexity and dynamism of the scenario under study; different parts of Milan present different traffic across the periods. This poses a challenge when using just one model to predict the traffic for all cells, since DL models learn the pattern from the input data. Therefore, if a DL model trained with the Telecom
Italia dataset learns the pattern of Cell 1 traffic, it may present a high error to predict the traffic of the Cell 1000, due to the different patterns. On the other hand, to create a prediction model for each cell present in the dataset will result in 10,000 models, which can be complex to manage, since these models must be trained and retrained if the traffic pattern of the cells changes.

In order to deal with these issues, we create clusters of cells based on their traffic prediction patterns. Figure 4 illustrates the pipeline used to create the cells' clusters to make the traffic prediction. Initially, we calculate the traffic of each cell present in the Telecom Italia dataset. Afterwards, we create clusters of cells based on the traffic statistics, as we will explain in Section 4.3. We group all cells with similar traffic patterns into a cluster. Once the clusters of cells are created, we calculate the mean traffic of all cells for each cluster, composing $N$ time series, where $N$ is the number of clusters. These time series (mean traffic of each cluster) are used to train DL models for traffic prediction. Instead of training one model for each cell, which results in a high number of models to train (e.g., for the Milan region, approximately 10,000 DL models would need to be trained and managed), we create $N$ DL models, one for each cluster. Since the traffic for cells in a given cluster have a similar pattern, the model of the respective cluster can be used to predict traffic of the cells that compose that cluster. Cell clusters are discussed in more detail in the next subsection.

Since the DL models of each cluster are trained with the mean traffic of the respective cluster, we can use these models to predict the mean traffic for each cluster, the last step in our pipeline. To evaluate the performance of the DL models, we use RMSE and MAE as metrics as detailed later in Section 4.5.

### 4.3 Clustering the cells

The dataset is composed of traffic data for different cells for Milan. As discussed previously, we propose a methodology based on cell clustering using Internet activity as a statistical metric to propose DL models to predict
Internet traffic. We calculate the total number of Internet activities considering six periods in each day, as described in Table 1.

Each cell can be represented by a vector containing six values based on the average Internet activity for each period of the day. Based on these values, we create clusters of cells using the $K$-means algorithm, which has been widely used in a number of different research domains such as document classification, recommendation systems based on user interests, classification based on user purchase behavior and so forth. The $K$-means algorithm has many advantages compared to other clustering techniques including ease of implementation and fast convergence, even for Big Data.

To automatically estimate the optimal number of clusters ($k$) of cells, we applied the elbow method. This method varies the number of clusters within a range to find the optimal $k$ based on the sum of square error. We varied $k$ from 1 to 50 as shown in Figure 5. As the number of cluster increases, the sum of squared distance tends towards zero with the elbow of curve being the optimal value. In our case, we select $k = 12$ since it is at the end of the elbow and the beginning of the stabilization of the sum of the squared distances. Based on the results presented in Figure 5, using more than 12 clusters would only increase the complexity of the algorithm with no significant gains in terms of performance.

The 12 clusters have a similar Internet activity pattern across different time periods within each day, regardless of the cell location. Figure 6 shows the 12 clusters overlaid on a map of the Milan Metropolitan Area. To some extent, Cluster 1 represents the external areas of Milan; Cluster 10 represents the border of the municipalities (excluding the municipality 1 in the center), while Cluster 6 is at the center of such municipalities. Other clusters (2, 3, 4, 5, 7, 8, 9, 11, and 12) are regions within the city.

We calculate the mean Internet traffic for each of the 12 clusters using the 30-min aggregated traffic of all cells included in each cluster divided by the number of cells of the cluster. Thus, for each cluster, we have a time series related to their mean cell’s traffic. This time series is used to train and test the proposed DL models.

| Period          | Time (in hour) |
|-----------------|----------------|
| Late night      | 00:00–04:00    |
| Early morning   | 04:00–08:00    |
| Morning         | 08:00–12:00    |
| Afternoon       | 12:00–16:00    |
| Evening         | 16:00–20:00    |
| Night           | 20:00–00:00    |

*In Northern Italy, unlike other countries where lunch is 12:30–14:00, the working day often includes a break from 12:00–13:30 or 14:30–16:00. For the purposes of this study, we have aggregated this as a 4-h block. Researchers may need to modify this for other countries.
In this paper, we propose two different RNNs that are widely used in the DL literature for regression problems, LSTM and GRU. To find the best configuration of the models, we apply a technique called Grid Search. This technique performs an exhaustive search in a subset of the previously defined parameters and provides the near optimal parameter combination within the given range. To apply the Grid Search, we vary the number of hidden layers and their units for both LSTM and GRU (see the parameters and levels in Table 2).

The first layer of the model is a fixed recurrent layer (the same as the hidden layers) where the number of units equals the input data length. The last layer is a fully connected layer with one neuron that gives the prediction value. Table 3 shows the fixed parameters (empirically chosen) to train the DL models.

### Table 2: Grid Search parameters and levels

| Parameters       | Levels                                         |
|------------------|------------------------------------------------|
| Number of layers | 1 to 4, Step 1                                  |
| Number of units  | 50 to 150, Step 50                              |

### Table 3: Parameters used to train the deep learning (DL) models

| Parameter                     | Value                           |
|-------------------------------|---------------------------------|
| Activation function of recurrent layers | Sigmoid                        |
| Activation function of last layer         | Hard sigmoid                    |
| Number of epochs              | 50                              |
| Optimizer                     | ADAM                            |
| Learning rate                 | 0.001                           |
| Batch size                    | 32                              |
| Loss function                 | Mean squared error              |
| Number of runs                | 30                              |

## 4.4 DL model configuration

In this paper, we propose two different RNNs that are widely used in the DL literature for regression problems, LSTM and GRU. To find the best configuration of the models, we apply a technique called Grid Search. This technique performs an exhaustive search in a subset of the previously defined parameters and provides the near optimal parameter combination within the given range. To apply the Grid Search, we vary the number of hidden layers and their units for both LSTM and GRU (see the parameters and levels in Table 2).

The first layer of the model is a fixed recurrent layer (the same as the hidden layers) where the number of units equals the input data length. The last layer is a fully connected layer with one neuron that gives the prediction value. Table 3 shows the fixed parameters (empirically chosen) to train the DL models.
Due to the random characteristics that exist in training (e.g., initialization of weights and selection of batches), we perform the experiments 30 times and calculate the average RMSE and the average MAE.

### 4.5 Metrics for evaluating DL models

To assess the performance of the models, we use two metrics: RMSE and MAE. The RMSE metric is calculated as per Equation (12):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2}$$  \hspace{1cm} (12)

where $N$ is the number of points from the traffic series, $f_i$ is the model prediction at timestamp $i$, and $y_i$ is the real value at timestamp $i$.\(^{48}\) We use RMSE because it measures the deviation between the true value and the value predicted by the model and is widely used in extant literature for evaluating traffic prediction models.\(^{34,49-51}\)

We also used MAE to evaluate the DL models. In contrast to RMSE, MAE assigns the same weight to all errors.\(^{52}\) MAE can be calculated as per Equation (13):

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|f_i - y_i|$$  \hspace{1cm} (13)

where $N$ is the length of time series, $f_i$ is the prediction, and $y_i$ is the actual value of timestamp $i$. Similar to the RMSE, MAE is widely used in the literature to evaluate traffic prediction models.\(^{53-55}\)

### 5 RESULTS

We use only the RMSE metric to assess the different DL architectures performance and find the best configuration for each one. In Section 5.2, we use both RMSE and MAE to compare the best configurations of DL models between them and against ML models. Figures 7 and 8 present the Grid Search RMSE results for each cluster for each of the LSTM and GRU models, respectively. For the LSTM models (Figure 7), for the majority of the clusters, the best configuration has one layer and 250 units. The only exception is Cluster 12, where the best overall average RMSE is achieved using a configuration with one layer and 150 units (LSTM-12-1L-150U) and one layer with 200 units (LSTM-12-1L-200U). Cluster 1 achieves the worst average RMSE result (0.084), while Cluster 12 achieves the best average RMSE result (0.068).

For GRU (Figure 8), the configurations with one layer present the best average RMSE; however, the performance of different clusters varies based on the number of units. From Clusters 2 to 9 (GRU-2-1L-200U, GRU-3-1L-200U, GRU-4-1L-200U, GRU-5-1L-200U, GRU-6-1L-200U, GRU-7-1L-200U, GRU-8-1L-200U, and GRU-9-1L-200U), the configuration with the lowest average RMSE is one layer with 200 units. For Clusters 1 and 10, the configuration that provides the best average RMSE is one layer with 250 units (GRU-1-1L-250U and GRU-10-1L-250U). In Clusters 11 and 12, two configurations have the best average RMSE, all with only one layer. For Cluster 11, the best configurations have 150 units (GRU-11-1L-150U) and 250 (GRU-11-1L-250U) units while for Cluster 12 the best performing configuration has 150 units (GRU-12-1L-150U) and 200 units (GRU-12-1L-250U). Similar to LSTM, the clusters with the worst and best average RMSE are Clusters 1 (0.0091) and 12 (0.0045), respectively.

The complexity of the model is directly related to the number of layers and units, that is, the more layers and units, the more complex the model becomes. Model complexity has to be adjusted according to the data. Figures 7 and 8 illustrate that very complex models (those with many layers and units) resulted in poorer performance, due to model overfitting. Simpler models with less complexity also performed poorly, most likely due to model underfitting.

In general, the DL models with one hidden layer obtain better average RMSE results than models with more layers, while those with 150 units or more result in better average RMSE results. Fine-tuning the models by increasing the number of units rather than layers results in better performance. Adding hidden layers results in performance degradation.
5.1 Statistical analysis

Some configurations achieved by the Grid Search obtained very similar average RMSE. To explore this further, we use Kruskal–Wallis non-parametric analysis to compare independent samples to check whether they are similar or not, based on the mean ranks of these samples.

In our LSTM results, Cluster 12 (LSTM-12-1L-150U and LSTM-12-1L-200U) has two configurations with the same average RMSE. Figure 9 presents the box plot of the RMSE of these Cluster 12 configurations. While the best Cluster 12 configurations have the same average RMSE (0.068), they have different RMSE distributions. We can note that the LSTM-12-1L-200U has a lower dispersion and a lower median than LSTM-12-1L-150U. LSTM-12-1L-200U has an outlier below the minimum RMSE value, and LSTM-12-1L-150U has an outlier above the maximum RMSE value. This analysis suggests that LSTM-12-1L-200U is the best configuration since it has the lowest dispersion and the lowest median.

For the GRU models, Clusters 1–3 and 5–9 each had at least one statistically similar best configuration (Figure 10). For Clusters 1–3 and 5–7, the configurations with 250 units (GRU-1-1L-250, GRU-2-1L-250, GRU-3-1L-250, GRU-5-1L-250, GRU-6-1L-250, and GRU-7-1L-250) presented a higher dispersion than the configurations with 200 units (GRU-1-1L-200, GRU-2-1L-200, GRU-3-1L-200, GRU-5-1L-200, GRU-6-1L-200, and GRU-7-1L-200); GRU-1-1L-200 presented a higher median than GRU-1-1L-250. For Cluster 8, three models are statistically similar, those with one layer and 150, 200, and 250 units. Again, the configuration with 200 units presented lower dispersion and a lower median than the other configurations. Finally, both configurations of Cluster 9 (GRU-9-1L-200 and GRU-1-1L-250) had very similar distributions, with similar dispersion and a similar median. In general, for these clusters with statistically
similar configurations, the configuration with one layer and 200 units presented a lower dispersion and lower median; this is considered the best performing configuration for the GRU models.

### 5.2 Comparison of LSTM and GRU models

Table 4 presents the results for the RMSE and MAE metrics, comparing the LSTM and GRU models against two traditional ML models as baseline: Random Forest and Decision Tree. For each cluster, we created the DL and ML models,
then trained and evaluated with the same training and testing datasets 30 times and calculated the average RMSE and MAE.

The DL models outperform the ML models for all clusters based on the RMSE and the MAE results. For the Cluster 1, the LSTM has a reduction of 50.32% in average RMSE and a reduction of 51.99% in average MAE when compared with Random Forest. Considering the Decision Tree, the LSTM presents a reduction of 62.67% in the average RMSE and a reduction of 62.96% in the average MAE. In Cluster 1, comparing the GRU and Random Forest, the GRU has a reduction of 43.78% in the average RMSE and 45.66% in the average MAE. In the same cluster, the GRU presents a reduction of 57.64% and 58.08%, for the average RMSE and MAE respectively, when compared with Decision Tree.

The DL models are clearly superior to the conventional ML models for traffic prediction tasks; the ML models cannot achieve the same level of prediction error of DL models based on the RMSE and the MAE.

**FIGURE 10** Boxplot of the RMSE of the best configurations of GRU models

**TABLE 4** Comparison of the LSTM, GRU, Random Forest, and Decision Tree models

| Cluster | LSTM | GRU | Random Forest | Decision Tree |
|---------|------|-----|---------------|---------------|
|         | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| 1       | 0.0842 | 0.0653 | 0.0953 | 0.0739 | 0.1695 | 0.1360 | 0.2251 | 0.1763 |
| 2       | 0.0809 | 0.0639 | 0.0917 | 0.0716 | 0.1664 | 0.1330 | 0.2200 | 0.1677 |
| 3       | 0.0803 | 0.0634 | 0.0922 | 0.0720 | 0.1645 | 0.1316 | 0.2202 | 0.1670 |
| 4       | 0.0796 | 0.0629 | 0.0896 | 0.0701 | 0.1699 | 0.1348 | 0.2052 | 0.1630 |
| 5       | 0.0800 | 0.0632 | 0.0909 | 0.0709 | 0.1602 | 0.1276 | 0.2007 | 0.1567 |
| 6       | 0.0806 | 0.0636 | 0.0928 | 0.0724 | 0.1665 | 0.1331 | 0.2198 | 0.1671 |
| 7       | 0.0807 | 0.0638 | 0.0920 | 0.0718 | 0.1675 | 0.1339 | 0.2185 | 0.1654 |
| 8       | 0.0808 | 0.0639 | 0.0929 | 0.0725 | 0.1684 | 0.1347 | 0.2208 | 0.1680 |
| 9       | 0.0805 | 0.0636 | 0.0917 | 0.0716 | 0.1680 | 0.1345 | 0.2219 | 0.1699 |
| 10      | 0.0683 | 0.0544 | 0.0715 | 0.0564 | 0.0957 | 0.0740 | 0.1084 | 0.0824 |
| 11      | 0.0712 | 0.0564 | 0.0719 | 0.0565 | 0.0842 | 0.0627 | 0.1072 | 0.0788 |
| 12      | 0.0677 | 0.0546 | 0.0671 | 0.0539 | 0.0784 | 0.0580 | 0.1035 | 0.0772 |

Abbreviations: GRU, gated recurrent unit; LSTM, long short-term memory; MAE, mean absolute error; RMSE, root mean squared error.
ML model results suggest that the Random Forest model outperforms the Decision Tree model for all clusters. For Cluster 1, the Random Forest model presents an average RMSE of 0.1695, while the Decision Tree model presented an average RMSE of 0.2251, a difference of 24.70%. For Cluster 1, the Random Forest and Decision Tree present an average MAE of 0.1360 and 0.1763, respectively, a difference of 29.63%.

We compared the best configurations for the LSTM and GRU models for each cluster using the Kruskal–Wallis test. The best configurations of LSTM and GRU based on the number of layers and RNN units were selected using the lowest average RMSE values (see Figures 7 and 8). We consider 30 RMSE values obtained from experiments. For all clusters, the RMSE distributions of the LSTM and GRU models are statistically different. From Table 4, one can note that the LSTM models obtain better average RMSE and MAE results except for Cluster 12, where the GRU slightly outperforms the LSTM.

Cluster 1 presents the worst results for RMSE and MAE. LSTM presents an average RMSE of 0.0842, while the GRU presents 0.0953, a difference of 13.18%. In the MAE results for Cluster 1, the LSTM presents 0.0653, while the GRU presents 0.0739, a difference of 13.17%. In contrast, Cluster 12 has the best average RMSE, 0.0677 and 0.0671 for LSTM and GRU, respectively, with a difference of 0.89%. For the same cluster, the LSTM and GRU present an average MAE of 0.0546 and 0.0539, respectively, with a difference of 1.28%.

To help understand the difference between the performance of the models across the clusters, we visualize the actual Internet activity (green line), the LSTM model Internet traffic predictions (blue line), and the GRU model internet predictions (orange line) (see Figure 11). For visualization and sensemaking purposes, we omit the ML model predictions.

Both models learned the pattern of Internet activity data, capturing its seasonality. For Clusters 1–9, the models' predictions are slightly lower than the ground truth data in some periods, with the GRU prediction values lower than the LSTM predictions, consistent with the GRU models' higher average RMSE and MAE results. For Clusters 10–12, the predictions of both models are closer to the ground truth data in comparison to the other clusters, with the LSTM model closer to the ground truth data in more periods than the GRU models. It is important to notice that Cluster 10 is situated at the outskirt of the metropolitan area of Milan while Clusters 11 and 12 are closer to the center of the city.

The periods where the predictions of the DL models (for all clusters) are more distant from the ground truth data are in the Christmas period (December 24–26). During this period, the predictions of both models are much lower than the ground truth data. This can be explained easily by the seasonal, although predictable, traffic at that time. This could be addressed by augmenting the overall DL scheme with historical statistics that summarize prior knowledge of predictable long-term trends as per Zhang and Patras.37

Based on our statistical tests, the LSTM models outperform the GRU models in all but one cluster. In that case, Cluster 12, the superior performance of the GRU model is relatively small. As shown in Figure 11, both RNNs capture the data pattern; however, the values predicted by the LSTM models are closer to the ground truth Internet activity data. This is not entirely surprising. As discussed in Section 2.2.2, typically LSTM is expected to outperform its less complex variant, GRU. We attribute the lower performance of the GRU to the lower complexity of the GRU units in comparison to the LSTM. Figure 11 shows that the predictions of the GRU model follow a similar pattern as the LSTM, but with lower values. These lower values increase prediction error, resulting in a higher RMSE and MAE. Notwithstanding this, the GRU model took less time to train, and thus, where a model needs to be retrained for multi-step traffic prediction, the benefits of efficiency gains (e.g., faster decision making, and computational and economic cost savings) may outweigh the differences in accuracy.

As discussed earlier, Chen et al16 also used the Telecom Italia dataset for Milan and an LSTM to predict base station traffic. While they used a clustering strategy at a group base stations level, it was post hoc, that is, they forecast the traffic patterns using the LSTM and then cluster complementary base stations to BBU based on the traffic patterns. As such, the results presented by Chen et al16 are not entirely comparable with this study since they considered a different clustering strategy and a different time interval in their experiments. In fact, the different clustering approaches may result in different traffic patterns resulting in a direct impact on the model performance. Finding related works for direct comparison in the context of traffic prediction is a complex task as different studies may vary dataset, evaluation metrics, transformations, and data preprocessing. Notwithstanding the differences, Chen et al16 is the closest to study identified to that presented in this article, and consequently, we use it as a basis for comparison. All average MAE values obtained for LSTM models were lower than the best result presented in Chen et al,16 which was 0.074. For the GRU models, Cluster 1 obtained the same MAE as Chen et al16 (0.0739) while the models for the other clusters performed better. Cluster 12 achieved the best MAE. The LSTM presented an improvement of 26.22%, and the GRU model presented an improvement of 27.16% when compared to the reported results in Chen et al.16
Figure 11  Comparison of ground truth Internet activity and LSTM and GRU Internet traffic predictions
6 | CONCLUSION

Global total mobile data traffic is projected to grow 4X to reach 160EB per month in 2025; 5G networks will carry nearly half of the world’s mobile data traffic by that time.\(^5\) This massive surge in demand for mobile broadband requires solutions that satisfy QoS and QoE requirements with minimum service delay and within budget constraints. Mobile network traffic prediction will be an essential input into infrastructure planning as well as dynamic and proactive network resource optimization. Extant approaches may have unacceptable accuracy, training times, turnaround times, lack computational complexity and may not therefore account for characteristics such as bursting, non-linear patterns, or other important correlations to meet the QoS and QoE requirements of increasingly demanding end users.\(^6,9\) These issues can result in mis-timed resource allocation as well as overutilization and underutilization. DL has the potential to address these shortcomings.

In this work, we proposed and compared the performance of two RNNs, LSTM, and GRU, to predict mobile Internet traffic in a large metropolitan area, Milan. We proposed a novel a priori clustering methodology to group cells using K-means clustering and used the Grid Search method to identify the best configurations for each RNN. We compared RNN performance using RMSE and MAE and testing against ground truth data for Milan. Both RNNs were effective in modeling Internet activity and seasonality, both within days and across 2 months however were sub-optimal in predicting anomalies, for example, Christmas. In this case, this could have been addressed by augmenting the training with historic trend data as per Zhang and Patras.\(^37\) We also find variations by in clusters across the city. While the LSTM outperformed the GRU, the GRU had faster training times which may be relevant for multi-step prediction scenarios. We compared our results with Random Forest and Decision Tree, common ML model techniques used for time series prediction. Both LSTM and GRU models outperformed the ML models for all clusters of cells. We also compared our proposed RNN models against the results in Chen et al\(^{16}\) using MAE. Notwithstanding the validity issues in such a comparison, results suggest our models present significantly better performance.

In future work, we plan to compare additional ML and DL architectures including ensemble approaches and augmenting the model with longer term historical trend data. Furthermore, we will extend the dataset with more heterogeneous data sources including SMS and voice call log data, among others, as well as other areas, for example, Trentino, available in the Telecom Italia dataset. Improved mobile network prediction can be applied to a wide range of network planning and optimization use cases to optimize utilization, reduce cost, and meet QoS. In future works, we will explore the efficacy of these models in a variety of use cases particularly where the faster training times of GRUs may provide advantages over LSTM, such as multi-step prediction and faster optimization time scales.

ACKNOWLEDGEMENTS

The authors would like to thank the Fundação de Amparo a Ciência e Tecnologia de Pernambuco (FACEPE) for funding this work through Grant IBPG-0059-1.03/19 and the Irish Institute of Digital Business (IIDB) at Dublin City University.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

ORCID

Guto Leoni Santos https://orcid.org/0000-0002-0257-4214
Pierangelo Rosati https://orcid.org/0000-0002-6070-0426
Theo Lynn https://orcid.org/0000-0001-9284-7580
Judith Kelner https://orcid.org/0000-0002-2673-5887
Djamel Sadok https://orcid.org/0000-0001-5378-4732
Patricia Takako Endo https://orcid.org/0000-0002-9163-5583

REFERENCES

1. GSMA. The mobile economy 2020. Accessed: April, 2020; 2020.
2. Cisco. Cisco annual internet report. Accessed: April, 2020; 2020.
3. Aguzzi S, Bradshaw D, Canning M, et al. Definition of a research and innovation policy leveraging cloud computing and IoT combination. Final Report, Eur Comm SMART. 2013;37:2013.
35. Wang J, Tang J, Xu Z, Wang Y, Xue G, Zhang X, Yang D. Spatiotemporal modeling and prediction in cellular networks: a big data enabled deep learning approach. In: IEEE INFOCOM 2017 - IEEE Conference on Computer Communications; 2017; Atlanta, GA, USA: 1-9.
36. Alawe I, Ksentini A, Hadjadj-Aoul Y, Bertin P. Improving traffic forecasting for 5G core network scalability: a machine learning approach. IEEE Netw. 2018;32(6):42-49.
37. Zhang C, Patras P. Long-term mobile traffic forecasting using deep spatio-temporal neural networks. In: Proceedings of the eighteenth acm international symposium on mobile ad hoc networking and computing. Los Angeles, USA: 2018:231-240.
38. Feng J, Chen X, Gao R, Zeng M, Li Y. DeepTP: an end-to-end neural network for mobile cellular traffic prediction. IEEE Netw. 2018; 32(6):108-115.
39. Eurostat. Population on 1 January by age groups and sex—cities and greater cities; 2020.
40. Barlacchi G, De Nadai M, Larcher R, et al. A multi-source dataset of urban life in the city of Milan and the province of Trentino. Sci Data. 2015;2:150055.
41. Hussain B, Du Q, Zhang S, Imran A, Imran MA. Mobile edge computing-based data-driven deep learning framework for anomaly detection. IEEE Access. 2019;7:137,656-137,667.
42. Mededovic E, Douros VG, Mähönen P. Node centrality metrics for hotspots analysis in telecom big data. In: IEEE Infocom 2019-IEEE conference on computer communications workshops (INFOCOM WKSHPS) IEEE. Paris, France; 2019:417-422.
43. Zhang D, Liu L, Xie C, Yang B, Liu Q. Citywide cellular traffic prediction based on a hybrid spatiotemporal network. Algorithms. 2020; 13(1):20.
44. Amin F, Choi GS. Hotspots analysis using cyber-physical-social system for a smart city. IEEE Access. 2020;8:122,197-122,209.
45. Yuan C, Yang H. Research on k-value selection method of k-means clustering algorithm. J-Multidiscip Sci J. 2019;2(2):226-235.
46. Syakur MA, Khotimah BK, Rochman EMS, Satoto BD. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. IOP Conference Series: Materials Science and Engineering: IOP Publishing, Surabaya, Indonesia; 2018: 012017.
47. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980; 2014.
48. Wang W, Lu Y. Analysis of the mean absolute error (MAE) and the root mean square error (RMSE) in assessing rounding model. IOP Conference Series: Materials science and engineering IOP Publishing; 2018:012049.
49. Zeng Q, Sun Q, Chen G, Duan H, Li C, Song G. Traffic prediction of wireless cellular networks based on deep transfer learning and cross-domain data. IEEE Access. 2020;8:172,387-172,397.
50. Kuber T, Seskar I, Mandayam N. Traffic prediction by augmenting cellular data with non-cellular attributes.; 2021:1-6.
51. Shen W, Zhang H, Guo S, Zhang C. Time-wise attention aided convolutional neural network for data-driven cellular traffic prediction. IEEE Wirel Commun Lett. 2021;10:1747-1751.
52. Chai T, Draxler RR. Root mean square error (RMSE) or mean absolute error (MAE)?—arguments against avoiding RMSE in the literature. Geosci Model Dev. 2014;7(3):1247-1250.
53. Zang Y, Ni F, Feng Z, Cui S, Ding Z. Wavelet transform processing for cellular traffic prediction in machine learning networks. In: 2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP) IEEE; 2015; Chengdu, China: 458-462.
54. Troia S, Alvizu R, Zhou Y, Maier G, Pattavina A. Deep learning-based traffic prediction for network optimization. In: 2018 20th International Conference on Transparent Optical Networks (ICTON) IEEE; 2018; Bucharest, Romania:1-4.
55. Cao S, Liu W. LSTM network based traffic flow prediction for cellular networks. International conference on simulation tools and techniques. Cham: Springer; 2019:643-653.
56. Elliott AC, Hynan LS. A SAS® macro implementation of a multiple comparison post hoc test for a Kruskal–Wallis analysis. Comput Methods Prog Biomed. 2011;102(7):75-80.
57. Ericsson. Ericsson mobility report: 5G uptake even faster than expected. Accessed: April, 2020; 2019.

AUTHORS BIOGRAPHIES

Guto L. Santos obtained his BSc in Information System from Universidade de Pernambuco and his MSc degree in Computer Science from Universidade Federal de Pernambuco (UFPE). Currently, he is a PhD student of Computer Science at UFPE and has taken part in the GPRT since 2016. He is taking part in GPRT, and his research interests include cloud computing, performance evaluation, Internet of Things, fog computing, and artificial intelligence, machine learning, and deep learning.

Pierangelo Rosati is an assistant professor in Business Analytics at DCU Business School, Ireland, and a deputy director of the Irish Institute of Digital Business (IIDB). Dr Rosati specializes in data analytics to support business decisions.
Theo Lynn is a full professor of Digital Business at Dublin City University. Professor Lynn specializes in the role of digital technologies in transforming business processes with a specific focus on cloud computing, social media, and data science.

Judith Kelner received her PhD from the Computing Laboratory at the University of Kent at Canterbury, UK, in 1993. She has been a full professor at Universidade Federal de Pernambuco (UFPE), since 1979. Currently, she leads the GRVM team as well as coordinates a number of research projects in the areas of multimedia systems, design of virtual and augmented reality applications, and smart communication devices.

Djamel Sadok received his PhD of Computer Science at the University of Kent at Canterbury, UK in 1990. He has been a member of staff at Universidade Federal de Pernambuco (UFPE) since 1993. His research interests include communication systems, access networks, security, cloud computing, and traffic classification. Currently, he leads the GPRT team as well coordinates a number of research projects.

Patricia T. Endo received her PhD of Computer Science from the Universidade Federal de Pernambuco (UFPE), Brazil in 2014. She has been a professor at Universidade de Pernambuco (UPE), Brazil since 2009 and a researcher at Research in Networks and Telecommunication Group (GPRT) since 2009. Her current research interests are cloud and fog computing, and resource management.

**How to cite this article:** Santos GL, Rosati P, Lynn T, Kelner J, Sadok D, Endo PT. Predicting short-term mobile Internet traffic from Internet activity using recurrent neural networks. *Int J Network Mgmt.* 2022;32(3): e2191. doi:10.1002/nem.2191