Mobile Traffic Prediction from Raw Data Using LSTM Networks

Hoang Duy Trinh, Lorenza Giupponi, Paolo Dini
CTTC/CERCA, Av. Carl Friedrich Gauss, 7, 08860, Castelldefels, Barcelona, Spain
{hdtrinh, lgiupponi, pdini}@cttc.es

***Abstract*—Predictive analysis on mobile network traffic is becoming of fundamental importance for the next generation cellular network. Proactively knowing the user demands, allows the system for an optimal resource allocation. In this paper, we study the mobile traffic of an LTE base station and we design a system for the traffic prediction using Recurrent Neural Networks. The mobile traffic information is gathered from the Physical Downlink Control CHannel (PDCCH) of the LTE using the passive tool presented in [1]. Using this tool we are able to collect all the control information at 1 ms resolution from the base station. This information comprises the resource blocks, the transport block size and the modulation scheme assigned to each user connected to the eNodeB.**

The design of the prediction system includes long short term memory units. With respect to a Multilayer Perceptron Network, or other artificial neurons structures, recurrent networks are advantageous for problems with sequential data (e.g. language modeling) [2]. In our case, we state the problem as a supervised multivariate prediction of the mobile traffic, where the objective is to minimize the prediction error given the information extracted from the PDCCH. We evaluate the one-step prediction and the long-term prediction errors of the proposed methodology, considering different numbers for the duration of the observed values, which determines the memory length of the LSTM network and how much information must be stored for a precise traffic prediction.

I. Introduction

Understanding the dynamic of the traffic demands in a wireless network represents a complex task, due to the massive densification of the mobile devices attached to the network. This is made more challenging by the huge variety of devices available today, and by the different typologies of service that they can offer. Within few years, the fifth-generation (5G) cellular network is promising to enable a plethora of new applications, including M2M communications, autonomous driving and virtual reality applications, which will require a boost in the performance of the network in terms of latency, capacity and context awareness [3].

To meet these strict requirements it is fundamental that the network becomes aware of the traffic demands. The analysis of the traffic and the precise forecast of the user demands are essential for developing an intelligent network. Knowing in advance the user demands makes the network able to promptly manage the resource allocation among the contending users. A smart optimization of the physical resources is crucial to improve the users' quality of experience, but it is also beneficial for the energy efficiency of the overall network.

In recent years, the development of cheaper and more powerful hardware had made possible to unleash the potential of machine learning algorithms, in particular of deep-learning, for a wide range of applications (e.g. objects identification, speech recognition, etc.). Using modern GPUs, it is possible to run complex algorithms on large scale datasets, with minimal efforts [4]. There are numerous applications in which deep learning algorithms show excellent results (e.g. in computer vision), where the amount of data for the learning and for the training tasks are widely available.

From an academic perspective, one major problem related to the cellular networks, is the lack of traffic datasets to be studied. Users traffic data are not always made available by network operators or they can be found but with very limited information [5]. Commonly, the available datasets consist of the aggregated traffic derived from the Call Detail Records (CDRs), where text, voice and data are mixed without additional information on the technology and on which base station the users are attached to [6]. Therefore, no information on the utilization of the physical resources or on the scheduling optimization can be assessed.

Numerous efforts have been devised to understand the dynamic of the cellular networks. The prediction of mobile traffic patterns has been usually studied through time-series analysis methods. Most of the works use techniques such as Auto Regressive Integrated Moving Average (ARIMA) and its different flavours (e.g. SARIMA, ARIMAX, mixed ARIMA), to capture the trends of the temporal evolution of the mobile traffic, [7], [8]. However, one of the known limitations of such techniques is the poor robustness to the rapid fluctuations of the time-series, since the prediction tends to over reproduce the average of the past observed values [9]. Additionally, these methods work with homogeneous time-series, where the input and the prediction are within the same set of values.

In this paper, we exploit the exceptional abilities of the Long-Short Term Memory (LSTM) units to present a multistep predictive algorithm for the mobile data traffic. LSTM units are the elementary parts to form Recurrent Neural Networks (RNN), which are a particular extension of the FeedForward Neural Networks. In particular, RNNs show outstanding results for time-domain problems and sequential data, and they have been heavily adopted for text prediction and machine translation problems [2]. These neural network structures perfectly fit our traffic prediction problem, where the collected dataset is multivariate and presents heterogeneous, non-linear information about the users communications. Moreover, the time-domain characteristic of the mobile traffic can be assisted by the LSTM properties, which are able to capture the temporal trends of the data.

Results in literature show that LSTM networks outperform other machine learning approaches for time-series analysis in the traffic prediction. LSTM structures have been proposed in [9]. Here, the datasets consist of the spatio-temporal distribution of the mobile traffic in different base stations. Spatial correlation has been used to evidence similarities between neighbouring base stations. Even though LSTM has been applied for the traffic prediction, the input data consider only one metric (e.g. the traffic, spatially distributed), and only a one-step prediction is considered.

At the time of writing, to the best of our knowledge, we are the first to present a complete methodology for the data collection and for the design of a multistep predictive network, which exploit the ability of the LSTM to enhance the prediction accuracy. In particular, we use directly the raw data obtained from the traffic control channel, and only an aggregative operation is performed to predict the traffic for multiple steps onwards. This is critical in terms of speed of prediction, which can be done with very little preprocessing of the data, with a limited number of observations. Moreover, working with the raw data, the forecast of the traffic can be accomplished with a high time resolution, since we can capture the traffic variation, which is large, even in a few milliseconds time window.

The paper is organized as follows: in Section II we describe how we collect the data from the LTE control channel and how we aggregate the dataset. In Section III we present the architecture for the mobile traffic prediction, which includes LSTM units. Section IV and V are devoted to the numerical results and to conclusions, respectively.

II. The LTE Scheduling Traffic Dataset

A. LTE Control Channel

Using the methodology as done in [1] and [10], we can collect the LTE scheduling information of the users connected to a certain eNodeB. The advantage of this tool is the richness of information and the temporal granularity of the data. We can decode the Downlink Control Information (DCI) messages from the PDCCH, which contain the modulation and coding scheme (MCS), the number of resource blocks and the transport block size assigned to the users at each millisecond. This information is available for both the downlink and the uplink directions. Anonymity of data is ensured since the users are identified by a limited number of Radio Network Temporary Identifiers (RNTI), which are refreshed after 10.15 seconds of inactivity. Only scheduling control information are available: the PDCCH is unencrypted, while downlink and uplink actual data are transmitted over encrypted physical channels to secure the users privacy.

B. Dataset Aggregation

The collected dataset consists of one-month of scheduling information that we gathered by monitoring different eNodeBs located in the city of Barcelona, Spain. Let $\mathcal{D} = \{D_{c_1}, D_{c_2}, ..\}$ be the dataset, where D_{c_k} is the set of measurements for the monitored cell k. Given a D_{c_k}, for each connected user at the time t, temporary identified by a RNTI r, we decode the DCI message containing the resource blocks, the transport block size and other scheduling information for the uplink and the downlink directions. We store this information in a measure $\mathbf{s}_r(t)$, where t corresponds univocally to an LTE subframe number, or TTI, which is 1 ms long.

We calculate the aggregate cell traffic measurements for a given timeslot T, which is the sum of the traffic generated by all the RNTIs connected during the timeslot T, $R(T)$

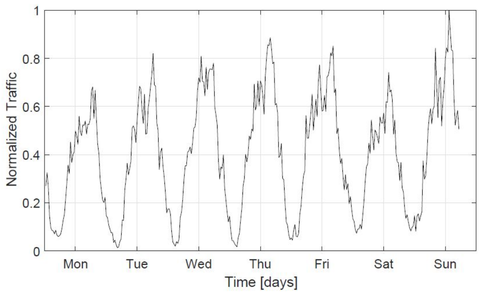
$$\mathbf{S}(T) = \sum_{r(t)\in R(T)} \sum_{t\in T} \mathbf{s}_r(t) \tag{1}$$

Thus, $\mathbf{S}(T)$ is the vector that contains the number of resource blocks allocated in the uplink and in the downlink directions, the number of the messages sent in both the directions and the sum of the total transport block sizes for a given timeslot T. Moreover, for each timeslot T, we include the number of the attached users to the eNodeB in the timeslot T. In our case we consider T, as the number of TTIs for which we aggregate the traffic.

III. A Long-Short Term Memory Network

Recurrent neural networks are a generalization of feedforward neural networks, that have been devised for handling temporal and predictive problems. LSTM are a particular kind of RNN, that have been introduced in [11]. They have been explicitly designed to avoid the long-term dependency issue, which is the cause of the vanishing-gradient problem in normal RNNs [12].

The capability of learning long-term dependencies is due to the structure of the LSTM units, which incorporates gates that regulate the learning process. In a standard LSTM unit (see Fig.2), the basic operations are accomplished by the input gate $\mathbf{i}_t$, the forget gate



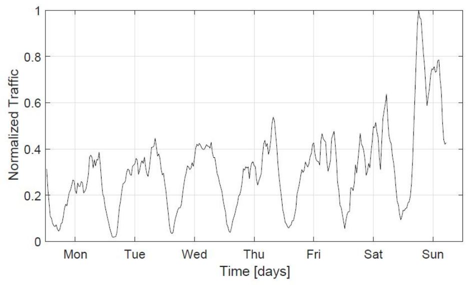


Fig. 1: Normalized weekly traffic signature of two monitored LTE eNodeBs.

$\mathbf{f}_t$ and the output gate $\mathbf{o}_t$. Moreover, the cell state $\mathbf{c}_t$ represents the memory of the unit and it is updated with the information to be kept (or to be forgotten), provided by the input gate (or forget gate).

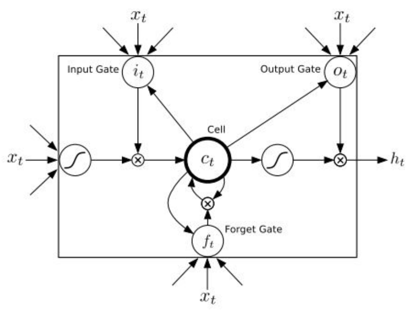


Fig. 2: Standard LSTM unit.

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_i)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_f)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_o)$$

$$\tilde{\mathbf{c}}_t = \phi(\mathbf{W}_c \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_c)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \phi(\mathbf{c}_t)$$

In the previous equations, $\sigma(\cdot)$ is the sigmoid function and ϕ is hyperbolic tangent function (tanh). $\mathbf{W}$ and $\mathbf{b}$ are respectively the weight matrix and the bias of the gates $\mathbf{i}$, $\mathbf{f}$, $\mathbf{o}$ or of the cell state $\mathbf{c}$. The subscript t is the time index and $\odot$ is the element-wise multiplication.

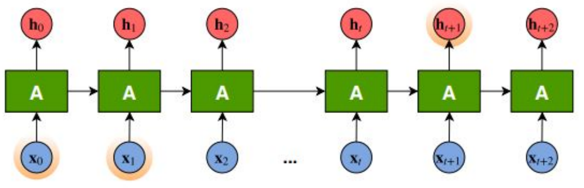


Fig. 3: Single-layer LSTM network.

The LSTM unit combines the output of the previous unit $\mathbf{h}_{t-1}$ with the current input $\mathbf{x}_t$ using the input, the output and the forget gates to update the memory of the cell. The variables $\mathbf{i}_t$ and $\mathbf{f}_t$ represent respectively the information that need to be kept or to be forgotten from the past and the current input. The cell state $\mathbf{c}_t$ is updated by summing the previous cell state $\mathbf{c}_{t-1}$ and the candidate cell state $\tilde{\mathbf{c}}_t$, weighted respectively with $\mathbf{f}_t$ and $\mathbf{i}_t$. Finally, we obtain the output $\mathbf{h}_t$ applying the tanh function to $\mathbf{c}_t$ and multiplying it by $\mathbf{o}_t$. Then, the current output $\mathbf{h}_t$ is passed to next unit and combined with the input at the next time index $t + 1$.

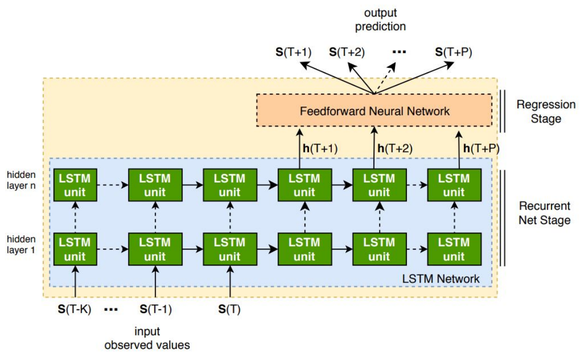


Fig. 4: Proposed architecture for the mobile traffic prediction.

Multiple LSTM units are concatenated to form one layer of the LSTM network. Each unit computes the operations on one time index and transfer the output to the next LSTM unit. The number of concatenated cells indicates the number of observations of the data that are considered before making the prediction. In our case, the input $\mathbf{x}_t$ is the eNodeB traffic vector $\mathbf{S}(T)$, and the number of observations is the number of selected timeslots T.

The proposed architecture for the mobile traffic prediction is depicted in Fig. 4. In our design, we consider multiple layers of basic LSTM units to form a stacked LSTM network. The intuition is that the deep LSTM network is able to learn the temporal dependencies of the aggregate mobile traffic: the LSTM unit of each layer extract a fixed number of features which are passed to the next layer. The depth of the network (e.g. the number of layers) is to increment the accuracy of the prediction, which is done by the last fully connected layer. For the one-step prediction we use a many-to-one architecture, which means that the network observes the mobile traffic for a fixed number of timeslots until T and, then try to predict the traffic in the next time slot $T+1$. In the multi-step prediction, we delay the prediction for a chosen number of timesteps, similarly to what is done for language modeling problems when they try to predict a sequence of words. Last, the output of the LSTM network is passed to a fully connected neural network, which performs the actual prediction. The motivation for this last layer is for the regression problem we are trying to solve: from an implementation perspective, this feedforward layer applies the softmax activation function, which is needed during the training phase to optimize the weights of the network neurons.

IV. Numerical Results

A. Evaluation Setup

We use the set of mobile traffic data from two different eNodeBs, that we collected during one month, to evaluate the performance of the proposed architecture. For each eNodeB, we calculate the aggregate cell traffic, as described in Section II-B.

We choose the Normalized Root Mean Square Error (NRMSE) as the metric to measure the accuracy of the prediction algorithm, which is given as

$$\text{NRMSE} = \frac{1}{\bar{x}} \sqrt{\frac{\sum_{t=1}^{N} (\tilde{x_t} - x_t)^2}{N}} \tag{2}$$

where N is the total number of points, $\tilde{x_t}$ and x_t are the predicted value and its correspondent observation at the time t and $\bar{x_t}$ is their mean. This same metric is used to compare the accuracy of the proposed architecture with the one obtained using other predictive algorithms.

The implementation of the mobile traffic prediction algorithm is done in Python, using Keras and Tensorflow, as backend. The chosen hyperparameters are reported in Table 1. The number of hidden layers is fixed to 5: this is one of the hyperparameters that need to be selected and can affect the tradeoff between the prediction accuracy and the time needed to train the network. A higher number of layers may increase the precision of the prediction, but we want to focus on the the relationship between the number of past observed values and the precision of the multi-step prediction, which determines the quantity of information needed to be memorized and utilized by the network. For the same reason, we fix the number of epochs to 100. Three weeks of data are used to train and to validate the architecture. Next results are related to the last week. We use the Adam optimization [13], to update the network weights iteratively based on the training data.

B. Results Analysis

Next, we present the results of multi-step prediction, that is when the output is delayed for a fixed number of timeslots and the prediction is performed for later time instants. We show how the accuracy decreases when we try to predict the traffic data in future timesteps. Furthermore, we analyze the effect of the number of observations that the LSTM network can see, and the duration of the timeslots T: these are design parameters that need to be estimated, since they determine the memory length of the LSTM network and how much traffic information is needed to be stored for an accurate prediction.

In Fig. 5, we show the results of the mobile traffic prediction for two cells: since they are located into two different areas, the monitored eNodeBs present two distinct traffic profiles in terms of profile and traffic magnitude. We can see that the prediction is precise for the whole week, despite the oscillating behaviour of the traffic. In this case, the prediction is one-step ahead, that means that we use a fixed number of past values $(K = 10)$ to predict the traffic for the next timeslot.

In Fig.6 and in Fig.7 we evaluate the prediction error with respect to past observed values. It is relevant also to consider different values for the timeslot duration of T, which affects the calculation of the aggregated traffic $\mathbf{S}(T)$ from the raw LTE traces. Figures are related to the first eNodeB (results are comparable for eNodeB 2). In Fig. 7, we see that the NRMSE is larger for a higher duration of T and, as expected, the error decreases with a larger number of observations. To emphasize the effect of the number of past observations, we plot the increasing accuracy (with respect to observing only one past value) for different values of T. We can observe that the major increase in percentage is given for larger values of T. For 10 past observed timesteps the accuracy can increase more than 40%.

TABLE I: Training Hyperparameters

Initial Learning Rate	0.001
Num. of Epochs	100
LSTM Hidden States	64
LSTM Hidden Layers	5
Feedforward Hidden Layers	1
Optimization Algorithm	Adam
Loss Function	MAE

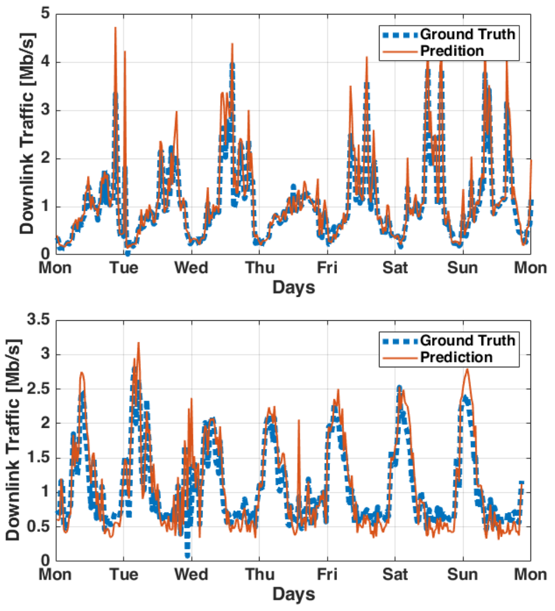


Fig. 5: Prediction of the weekly mobile traffic for two different eNodeBs.

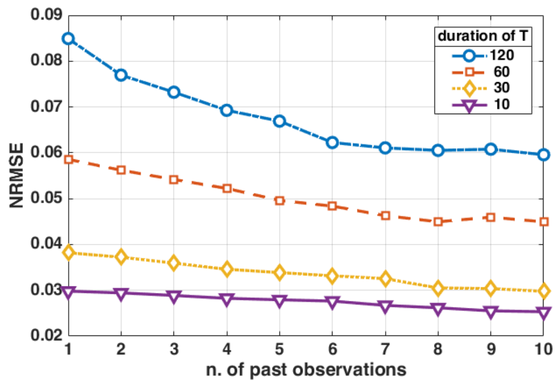


Fig. 6: Prediction error versus number of past observed values.

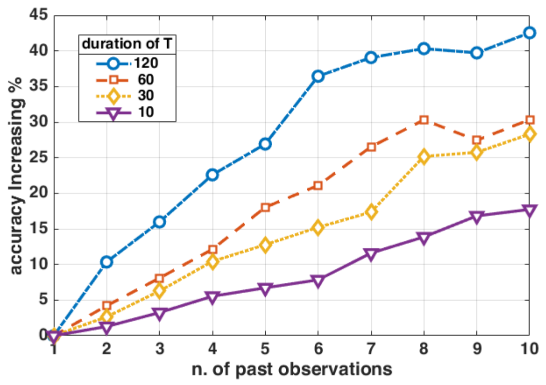


Fig. 7: Percentage of accuracy gaining versus number of past observed values.

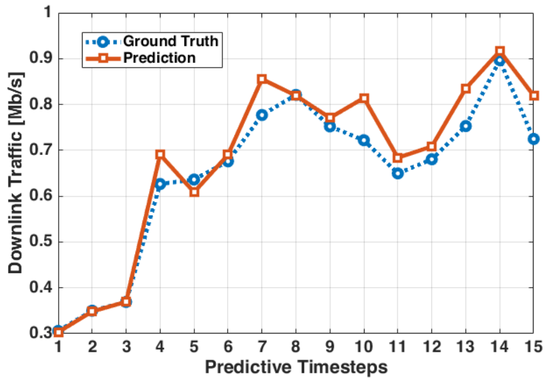


Fig. 8: Lookup for the predicted mobile traffic vs. ground-truth measurements for 15 steps ahead ($T = 10$).

In Fig.8 we show the prediction for 15 timesteps ahead. We fix $K = 10$ and $T = 10$ TTIs. At first, the prediction almost corresponds to the measured values, while after some steps the prediction error is more dominant. In Fig. 9, we plot the increasing error with respect to future prediction steps. As expected, longer prediction causes an increment in the accuracy of the algorithm. The error increases with different trends and it is around 40% when we predict for 15 steps ahead. This is similar to what happens in the problem of language synthesis for the prediction of long sentences: for further words prediction, the number of candidate words is larger, therefore there is more uncertainty in the correct choice. Conversely to the number of past observations, changing the duration of the timeslot T, does not give useful insight on the prediction error: for longer periods, the variability of the mobile traffic is larger, leading to an oscillating and randomic error for future predictions.

Finally, we compare the proposed architecture with two time-series prediction methods: an ARIMA model is a well-established technique for the time-series analysis, and it is defined by 3 parameters (p, d, q) that determine the auto-regression, the differentiation and the moving average, respectively. Here, we use a $(10, 1, 5)$ model. Also, note that we use only one variable for the prediction (i.e. the aggregate traffic), instead of multiple information obtained by the raw data. The other traffic prediction is obtained using a deep FeedForward Neural Network (FFNN), where we replace the LSTM neural network with a network of fully connected neurons. For a fair comparison, we use the same number of hidden layers. Figure 10 shows the traffic prediction on the same time window using these two techniques. The accuracy using the ARIMA model is lower, since the prediction tends to be closer to the average value of the traffic. On the other hand, the FFNN is able to follow the periodic trend and the traffic oscillations, but it still lack of a high precision. Also, we compare the average error for the three prediction methods on the two traffic profiles: as

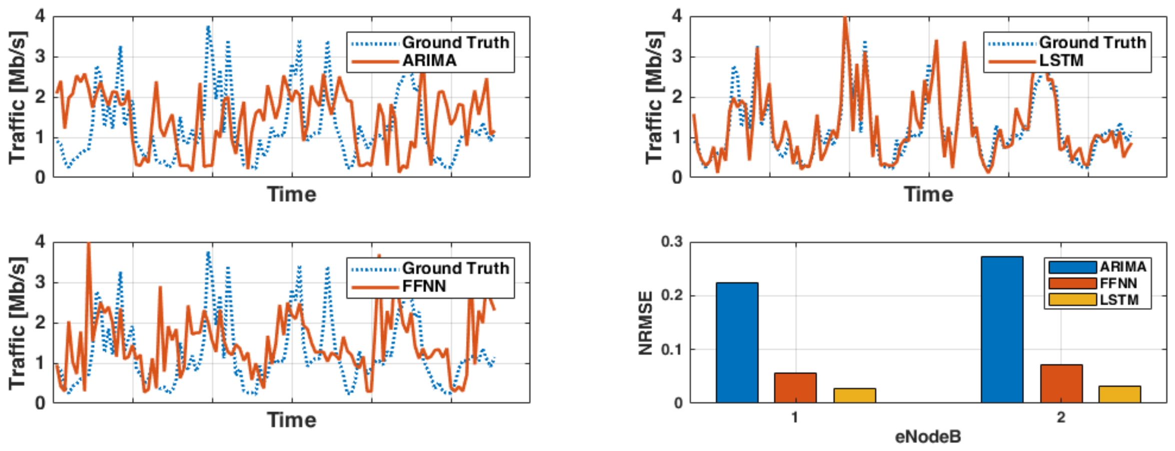


Fig. 10: Traffic prediction obtained with different model and errors.

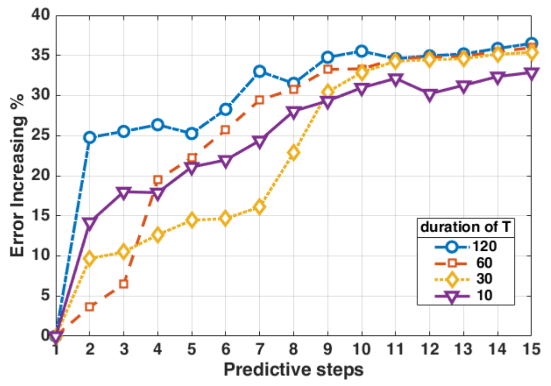


Fig. 9: Percentage of error increasing versus number of predicting timesteps.

expected, thanks to the LSTM properties, the proposed algorithm captures the temporal characteristics of the mobile traffic, and it provides superior accuracy with respect to the FeedForward Neural Network or to the classic ARIMA model.

V. Conclusion

In this work, we study the effectiveness of recurrent neural networks applied to the prediction of the mobile traffic. The choice of using LSTM network is imposed by the dataset characteristics, since we use multivariate traffic information that derive directly from the DCI of the LTE control channel. The LSTM units succeed in capturing the temporal correlation of the traffic even for distant timeslots. Applying the prediction of the traffic using raw aggregate data from the physical channel, is fundamental in time-critical applications and avoids the need for additional resources to process the traffic data.

Acknowledgement

The research leading to this work has received funding from the Spanish Ministry of Economy and Competitiveness under grant TEC2017-88373-R (5G REFINE) and from the European Union Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675891 (SCAVENGE).

References

[1] N. Bui and J. Widmer, “Owl: a reliable online watcher for lte control channel measurements,” *ACM All Things Cellular*, 2016.

[2] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder–decoder for statistical machine translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

[3] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, “What will 5g be?,” *IEEE Journal on selected areas in communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[4] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, “Theano: A cpu and gpu math compiler in python,” in *Proc. 9th Python in Science Conf*, vol. 1, 2010.

[5] V. D. Blondel, M. Esch, C. Chan, F. Clérot, P. Deville, E. Huens, F. Morlot, Z. Smoreda, and C. Ziemlicki, “Data for development: the d4d challenge on mobile phone data,” *arXiv preprint arXiv:1210.0137*, 2012.

[6] T. Italia, “Big data challenge 2015,” *aris.me/contents/teaching /data-mining-2015 /project/BigDataChallengeData.html*, 2015.

[7] Y. Shu, M. Yu, O. YANG, J. Liu, and H. Feng, “Wireless traffic modeling and prediction using seasonal arima models,” vol. E88B, 01 2003.

[8] G. P. Zhang, “Time series forecasting using a hybrid arima and neural network model,” *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[9] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang, “Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach,” in *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*, pp. 1–9, IEEE, 2017.

[10] H. D. Trinh, N. Bui, J. Widmer, L. Giupponi, and P. Dini, “Analysis and modeling of mobile traffic using real traces,” *PIMRC*, 2017.

[11] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual prediction with lstm,” 1999.

[13] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.