# Using Deep Neural Networks for Forecasting Cell Congestion on LTE Networks: A Simple Approach

Pedro Torres[1(✉)], Hugo Marques[1,2], Paulo Marques[1,2], and Jonathan Rodriguez[2]

[1] Instituto Politécnico de Castelo Branco, Castelo Branco, Portugal
{pedrotorres,hugo,paulomarques}@ipcb.pt
[2] Instituto de Telecomunicações, Campus de Santiago, Aveiro, Portugal
jonathan@av.it.pt

**Abstract.** Predicting short-term cellular load in LTE networks is of great importance for mobile operators as it assists in the efficient managing of network resources. Based on predicted behaviours, the network can be intended as a proactive system that enables reconfiguration when needed. Basically, it is the concept of self-organizing networks that ensures the requirements and the quality of service. This paper uses a dataset, provided by a mobile network operator, of collected downlink throughput samples from one cell in an area where cell congestion usually occurs and a Deep Neural Network (DNN) approach to perform short-term cell load forecasting. The results obtained indicate that DNN performs better results when compared to traditional approaches.

**Keywords:** LTE · SON · Machine learning · Deep learning · Forecasting

## 1 Introduction

With the constant demanding for high-speed data applications (e.g., high-quality wireless video streaming, social networking, machine-to-machine communication, IoT, etc.), LTE micro and femto cells, as well as relay nodes, are being relied upon to ensure that the required overall network capacity can be met. This, however, increases the challenges in terms of network planning and specially on management. As the network complexity increases, mobile network operators (MNOs) are required to invest more in network optimization processes and automation. Hence, reducing operational costs can be done automatically through Self-Organizing Networks (SON) strategies, which allow the network to heal and improve itself, based, for example, on forecasted behaviours. SON, therefore, minimizes rollout delays and operational expenditures associated with ongoing LTE deployments.

The fast-technological development rate, we have been assisting on the last years, had led to a strong interdependence of the new emerging technologies, examples are: 5G mobile broadband (5G), IoT, Big Data Analytics (Big Data), Cloud Computing (Cloud) and SDN. For the MNO, new interests are related to knowing more information on their costumers (e.g., known locations, used services and other customer related patterns such as data consumption trends).

P. Marques et al. (Eds.): CROWNCOM 2017, LNICST 228, pp. 276–286, 2018.
https://doi.org/10.1007/978-3-319-76207-4_23

One key task for MNOs is the correct dimensioning of their network capacity. Network load forecasting can assist in guaranteeing network resources are available to consumers, ensuring quality of service is met, therefore, avoiding consumer complaints. Improving the accuracy of load forecasting in a short period of time is a challenging open problem due to the variety of factors that influence data traffic and throughput.

Forecasting has been widely studied in deafferents areas, since the 1970s. Traditional methods introduced linear models for time series forecasting, such as autoregressive (AR), autoregressive with moving average (ARMA) and the derivations of these [1]. Currently, nonlinear forecasting models have generally obtained better accuracy than linear models. These nonlinear models are based on machine learning methods such as neural networks [2] or support vector machines [3]. Neural networks have been used widely in data forecasting due to their ability to approximate complex nonlinear relationships. However, neural network methods have some potential drawbacks such as overfitting of the model, sensitivity to random weight initialization and tendency to convergence to local optima. To address these limitations, recently new approaches, called Deep Neural Networks (DNN) [4], have been proposed. DNN, with additional nonlinear levels, can better represent complex features from their inputs and obtain a more general model with the ability to learn, from the data, these complex features.

One potential use for DNN based forecast methods, that we foresee in this paper, is to predict data access patterns on mobile networks, as MNOs heavily invest in network dimensioning to avoid congestion. Typical actions include: making use of additional spectrum carriers (whenever possible), installing more cell sites and performing traffic offloading onto other networks such as Wi-Fi. In the identification of cell congestion two things need to be considered: throughput and latency. For subscribers, when an access node is near capacity, the rapid increase in latency causes a substantial deterioration of QoE for latency-sensitive applications. One of the most common congestion detection mechanisms is based on real-time measurements of access round trip time (RTT).

In the literature, there are currently not many works related to forecasting cell congestion, however there are some research studies addressing similar topics. In [6] the authors are motivated by the fact that cellular radio networks are seldom uniformly loaded, hence propose a reactive load-balancing algorithm that adjusts the cell individual offset parameter between the serving cell and all its neighbours by a fixed step. The authors claim that the best step depends on the load conditions in both the serving cell and its neighbours as well as on the serving cell's user distribution. This is the basis for their proposed Q-Learning algorithm, that learns the best step values to apply for different load conditions and demonstrate that the Q-Learning based algorithm performs better than the best fixed $\varphi$ algorithm in virtually all scenarios. In [5] the authors analyse different algorithms to match traffic demands with network resources. The compared techniques are implemented by a modified version of the Q-Learning algorithm, called the reinforcement Q-Learning algorithm, that forecasts load status for every node and that when combined with the SON may improve network performance. In [7] the autoregressive integrated moving average (ARIMA) model and exponential smoothing model are used to predict the throughput in a single cell and whole region in a LTE network.

This paper, on the other hand, describes a deep learning approach to predict cell congestion events based on the downlink average throughput. A cell congestion event was considered when we measured a drop of at least 50% (based on Service Level Agreement values) on the average download speed per user (considering an average of 20 Mbps per user during off-peak hours).

This work was developed in the scope of MUSCLES project (Mobile Ubiquitous Small Cells for Low-cost Energy and Spectrum efficient cloud service delivery) [8]. MUSCLES aims to research, implement and test a platform of LTE mobile networks autonomous management, with capabilities of self-organization, detection and automatic troubleshooting which are still being manually addressed by many MNOs. These capabilities have the potential to substantially reduce operational costs and are to be aligned with the new developments of SON technology proposed by 3GPP standardization organization.

The remaining of the paper is organized as follows: Sect. 2 summarizes the state of the art in what respects forecasting with focus in the classical methods and new deep learning trends; Sect. 3 explains the used methodology and methods; Experimental results are described and presented in Sect. 4 and; the conclusions are given in Sect. 5.

## 2 State of the Art Related to Forecast

In the context of forecasting, a time series is a sequence of periodic observations of a random variable (e.g., monthly energy consumption, the annual number of passengers in an airport, the daily temperature). Time series are important for applied-research because they are often the drivers of decision models. Time series analysis provides tools for selecting a model that best describes the time series and the model is then used to forecast future events in the time series. Modelling the time series is a statistical problem because observed data is used in computational procedures to estimate the coefficients of a supposed model. Models assume that observations vary randomly about an underlying mean value that is a function of time. In this work, the downlink average throughput of a single LTE cell was considered an observation of a time series. Next subsections introduce the time series forecasting traditional methods and the new trends based on deep neural networks.

### 2.1 Classical Framework

The classical forecasting framework uses traditional methods that are exhaustively described in the literature and with good practical applications demonstrated. Typically, these methods consist of defining a parametric model that is supposed to generate data. Based on a given sample, the unknown parameters of the model are estimated and the estimated model is used to make predictions. The following are the most popular classical methods.

An autoregressive (AR) model specifies that the output variable depends linearly on its own previous values and on a stochastic term (an imperfectly predictable term). $\boldsymbol{AR}(\boldsymbol{p})$ model is a linear generative model based on the $\boldsymbol{p}$th order Markov assumption:

$$\forall t, Y_t = \sum_{i=1}^{p} a_i Y_{t-i} + \varepsilon_t \tag{1}$$

where $\varepsilon_t$ are zero mean uncorrelated random variables with variance $\sigma$. $a_1, \ldots, a_p$ are autoregressive coefficients. $Y_t$ is observed stochastic process.

The moving-average (MA) model specifies that the output variable depends linearly on the current and various past values of a stochastic (imperfectly predictable) term. $\boldsymbol{MA}(\boldsymbol{q})$ model is a linear generative model for the noise term based on the $q$ th order Markov assumption:

$$\forall t, Y_t = \varepsilon_t + \sum_{j=1}^{q} b_j \varepsilon_{t-j} \tag{2}$$

where $b_1, \ldots, b_q$ are moving average coefficients.

Combined $\boldsymbol{AR}(\boldsymbol{p})$ and $\boldsymbol{MA}(\boldsymbol{q})$ models can be obtained through the autoregressive-moving-average ($\boldsymbol{ARMA}(\boldsymbol{p}, \boldsymbol{q})$) model:

$$\forall t, Y_t = \sum_{i=1}^{p} a_i Y_{t-i} + \varepsilon_t + \sum_{j=1}^{q} b_j \varepsilon_{t-j} \tag{3}$$

ARMA model is a weakly stationary process. When data show evidence of non-stationarity, an initial differencing step (corresponding to the “integrated” part of the model) can be applied one or more times to eliminate the non-stationarity and we are in the presence of an autoregressive integrated moving average (ARIMA) model.

Other extensions can be extensively used, such as models with seasonal components (SARIMA), models with side information (ARIMAX), models with long-memory (ARFIMA), multi-variate time series models (VAR), models with time-varying coefficients and other non-linear models.

There are different methods for estimating model parameters, such as Maximum likelihood estimation, method of moments or Conditional and unconditional least square estimation.

## 2.2 Deep Neural Networks

Another branch of time series forecasting consists in using machine learning techniques, such as support vector machines or artificial neural networks (ANN) [9]. The most common type of ANNs is a multilayer perceptron (MLP) that forecasts a profile using previous data. A deep neural network (DNN) [10] is an ANN with more layers than the typical three layers of MLP. The deep structure increases the feature abstraction capability of neural networks. Deep learning algorithms use multiple-layer architectures or deep architectures to extract inherent features in data from the lowest level to the highest level, and they can discover huge amounts of structure in the data. Figures 1 and 2, illustrate typical architectures of the ANN and DNN.

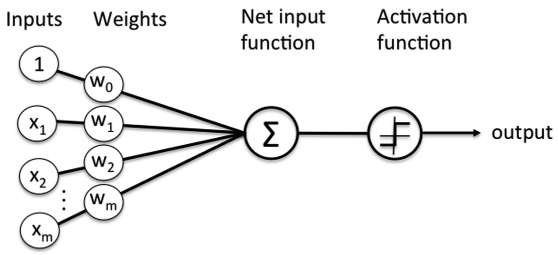

**Fig. 1.** Traditional neural network architecture, perceptron node

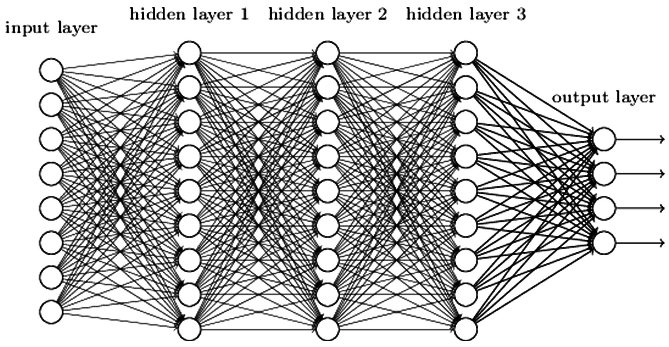

**Fig. 2.** Deep Neural Network architecture

## 3 Methodology and Methods

### 3.1 Methodology

The methodology used in this work consists of analysing the historical throughput data coming from one cell to train the forecasting methods to learn trends and predict future events, considering also the seasonality. Our dataset consists of hourly samples over a period of a month. The dataset was broken into 2 parts for training (75% of the dataset size) and validation and testing (25%). A Recurrent Neural Network (RNN) [11] using the Long Short-Term Memory (LSTM) [12] architecture was implemented using TensorFlow [13] to train and create our model.

**A summary on the used methodology:**

- Use of a dataset of historic measurements (four weeks), coming from 1 cell;
- The measurements were collected through an automated process and averages were computed on each hour;
- The measurements were used as the input for forecasting future network behaviour (time series analysis);

- The forecasting approach is to train by three weeks and computed for the 4th week;
- Forecasting is computed and validated, by comparing the obtained results with the real measurements, obtained for the 4th week;
- Results are ready to be exploited by the MNOs SON strategies.

## 3.2 Method

The deep learning method used consists in a RNN with a LSRM architecture. The LSTM network is a type of recurrent neural network used in deep learning because very large architectures can be successfully trained.

In a traditional RNN, during the gradient back-propagation phase, the gradient signal can end up being multiplied many times (as many as the number of timesteps) by the weight matrix associated with the connections between the neurons of the recurrent hidden layer. This means that, the magnitude of weights in the transition matrix can have a strong impact on the learning process.

If the weights in this matrix are small (or, more formally, if the leading eigenvalue of the weight matrix is smaller than 1.0), it can lead to a situation called vanishing gradients, where the gradient signal gets so small that learning either becomes very slow or stops working altogether. It can also make more difficult the task of learning long-term dependencies in the data. Conversely, if the weights in this matrix are large (or, again, more formally, if the leading eigenvalue of the weight matrix is larger than 1.0), it can lead to a situation where the gradient signal is so large that it can cause learning to diverge. This is often referred to as exploding gradients.

These issues are the main motivation behind the LSTM model which introduces a new structure called a memory cell (see Fig. 3 below).

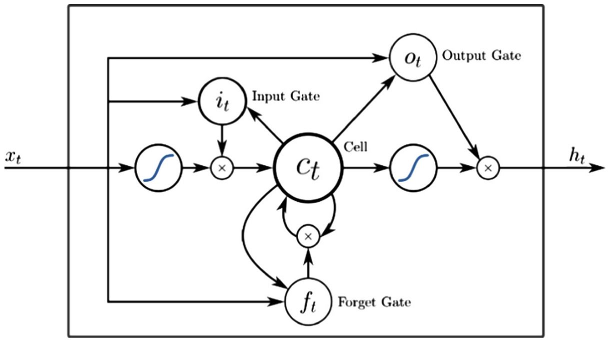

**Fig. 3.** LSTM memory cell, with input, output and forget gates

A memory cell is composed of four main elements: an input gate, a neuron with a self-recurrent connection (a connection to itself), a forget gate and an output gate. The self-recurrent connection has a weight of 1.0 and ensures that, barring any outside interference, the state of a memory cell can remain constant from one timestep to another. The gates serve to modulate the interactions between the memory cell itself and its environment. The input gate can allow incoming signal to alter the state of the memory cell or block it. On the other hand, the output gate can allow the state of the memory cell to influence other neurons or prevent it. Finally, the forget gate can modulate the memory cell's self-recurrent connection, allowing the cell to remember or forget its previous state, as needed.

In LSTM, the update of a layer of memory cells, at every timestep $t$, can be described as follows:

1. Compute the values for $i_t$, the input gate, and $\tilde{C}_t$ the candidate value for the states of the memory cells at time $t$:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$

$$\tilde{C}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{5}$$

2. Compute the value for $\boldsymbol{f_t}$, the activation of the memory cells forgets the gates at time $t$:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{6}$$

3. Given the value of the input gate activation $i_t$, the forget gate activation $f_t$ and the candidate state value $\tilde{C}_t$, we can compute $C_t$ the memory cells' new state at time $t$:

$$C_t = i_t * \tilde{C}_t + f_t * C_{t-1}) \tag{7}$$

4. With the new state of the memory cells, we can compute the value of their output gates and, subsequently, their outputs:

$$O_t = \sigma(W_O x_t + W_O h_{t-1} + V_O C_t + b_O) \tag{8}$$

$$h_t = O_t * \tanh(C_t) \tag{9}$$

where $x_t$ is the input of the memory cell layer at time $t$. $W_i$, $W_f$, $W_C$, $W_O$, $U_i$, $U_f$, $U_C$, $U_O$ and $V_O$ are weight matrices. $b_i$, $b_f$, $b_c$ and $b_O$ are bias vectors.

The model is composed of a single LSTM layer followed by an average pooling and a logistic regression layer as illustrated in Fig. 4 below. Thus, from an input sequence $x_0, x_1, \ldots, x_n$, the memory cells in the LSTM layer will produce a representation sequence $h_0, h_1, \ldots, h_n$. This representation sequence is then averaged over all timesteps resulting in representation $\boldsymbol{h}$. Finally, this representation is fed to a logistic regression layer whose target is the class label associated with the input sequence.

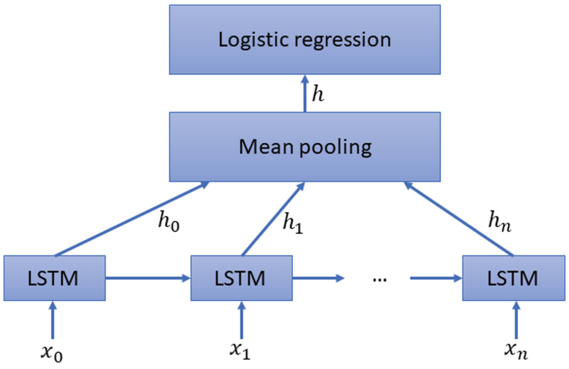

**Fig. 4.** Concept for the DNN based LSTM forecast model used in this paper. It is composed of a single LSTM layer followed by mean pooling over time and logistic regression

## 4 Experimental Results

This section presents the forecasting results obtained with a Deep Learning approach and the comparison with two classical methods, ARIMAX model and Naïve persistence model [14]. The goal is to forecast one entire week of the cell average downlink (DL) throughput, with a resolution of one hour. The input dataset for our model was provided by a MNO, originating from 1 cell in a dense urban area, where congestion problems typically occur. Three weeks of historical collected measurements have been used for training the prediction models and one week was used to compare the observed throughput with the forecast values.

Figure 5 depicts the forecasting results obtained from the deep learning method. The dataset is split, the data is separated into training datasets, where 75% of the dataset size was used to train the model, leaving the remaining 25% for validating the results.

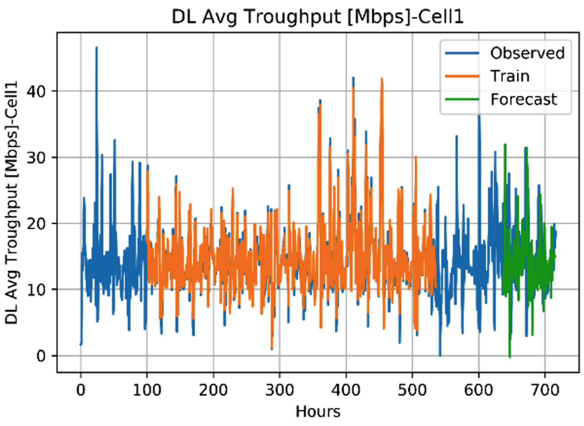

**Fig. 5.** Downlink average throughput observation for cell 1 (blue line) with the training (orange line) and forecast (green line) results. (Color figure online)

**Table 1.** Forecasting errors (MSE) expressed in Mbps

| Method | MSE [Mbps] |
|---|---|
| ARIMAX model | 7.16 |
| Naïve persistence model | 6.90 |
| Deep learning | 1.01 |

To measure the model accuracy, the Mean Squared Error (MSE) is computed and compared with 2 classical methods. The results presented in Table 1, show that the deep learning method is more accurate than the other methods, for this dataset. However, the computational cost associated to this method is higher in comparison to the classical methods.

Based on the more detailed forecasting results, in Fig. 6, it is possible predicted situations of cell congestion and proactively actuate in the network through SON functions that can change key network configuration parameters. For example, for this cell the average download speed, per user, on an LTE network on off-peak hours is around 20 Mbps. Cell congestion was considered when we observed a 50% reduction of this speed (identified in Fig. 6 by the asterisk symbol). Furthermore, measured values of 10 Mbps or lower, have been consistently observed on short periods (30 min to 1 h) during commonly identified peak-hours (e.g., 7:00 a.m. to 9:00 p.m.). In such cases, the network configuration parameters, for example a network carrier activation (if available), could be activated 1 h before the predicted event. The dots in Fig. 6 represent this decision point where something must be changed in the network to avoid the identified problems (asterisks) in the network.

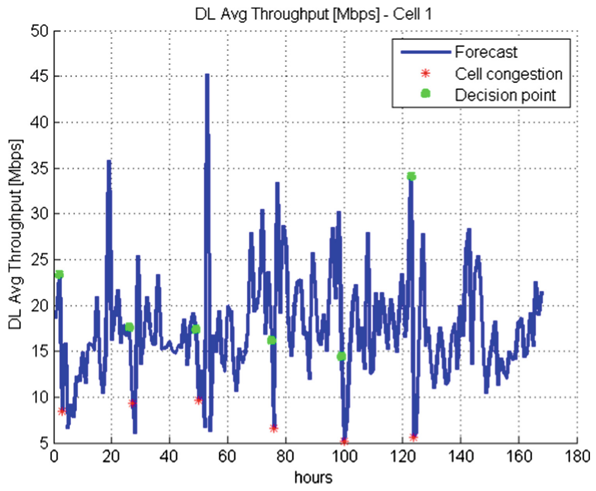

**Fig. 6.** Identification of cellular congestion (asterisks) and decision points (dots) to actuate on the network in order to avoid congestion. In the provided example, the actuation event occurs 1 h before the predicted congestion event.

It is important to note that the cellular congestion does not only depend on the download speed, it is however a very important key performance indicator (KPI) to be considered. The same method can be applied to forecast other network KPIs and the correlation between the different KPIs further improves the problem detection algorithm.

## 5 Conclusions

This work describes a deep neural network data analytics methodology and model, capable of forecasting the average downlink throughput of one LTE cell based on historic measurements. The obtained results have shown that, in comparison with other traditional forecasting methods, if an appropriate dataset of samples is provided, the proposed model is able to forecast with high accuracy the cell downlink throughput. We were able to predict a cell congestion event up to 30 h in advance which provides SON strategies enough time to react (e.g., by shifting coverage and capacity to areas in need), before subscribers have been impacted by dropped calls or reduced data speeds and therefore making MNOs happy by anticipating network problems and avoiding customer complaints. The current model is still prone to further improvements by refining the deep neural network algorithm. Furthermore, the authors are currently implementing the algorithm in an LTE system-level simulator to quantify network performance improvements based on cell congestion prediction and SON strategies.

**Acknowledgments.** This work is funded by the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project Nr. 17787] (POCI-01-0247-FEDER-MUSCLES).

## References

1. Deb, C., Zhang, F., Yang, J., Lee, S.E., Shah, K.W.: A review on time series forecasting techniques for building energy consumption. Renew. Sustain. Energy Rev. **74**, 902–924 (2017)
2. Zhang, G., Patuwo, B.E., Hu, M.Y.: Forecasting with artificial neural networks: the state of the art. Int. J. Forecast. **14**(1), 35–62 (1998)
3. Sapankevych, N.I., Sankar, R.: Time series prediction using support vector machines: a survey. IEEE Comput. Intell. Mag. **4**(2), 24–38 (2009)
4. Dalto, M., Matusko, J., Vasak, M.: Deep neural networks for ultra-short-term wind forecasting. In: Proceedings of the 2015 IEEE International Conference on Industrial Technology (ICIT), Seville, Spain, 17–19 March 2015, pp. 1657–1663 (2015)
5. Xu, J., Tang, L., Chen, Q., Yi, L.: Study on based reinforcement Q-Learning for mobile load balancing techniques in LTE-A HetNets. In: 2014 IEEE 17th International Conference on Computational Science and Engineering, Chengdu, pp. 1766–1771 (2014)
6. Mwanje, S.S., Mitschele-Thiel, A.: A Q-Learning strategy for LTE mobility load balancing. In: 2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), London, pp. 2154–2158 (2013)

7. Dong, X., Fan, W., Gu, J.: Predicting LTE throughput using traffic time series. ZTE Commun. **4** (2015)
8. MUSCLES project. https://www.celticplus.eu/project-muscles/
9. Moreno, J.J.M., Poll, A.P., Gracia, P.M.: Artificial neural networks applied to forecasting time series. Psicothema **23**(2), 322–329 (2011)
10. Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B.: Recent Advances in Convolutional Neural Networks, (2015) arXiv:1512.07108
11. Dorffner, G.: Neural networks for time series processing. Neural Netw. World (1996)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
13. TensorFlow. https://www.tensorflow.org/
14. Torres, P., Marques, P., Marques, H., Dionísio, R., Alves, T., Pereira, L., Ribeiro, J.: Data analytics for forecasting cell congestion on LTE networks. In: IEEE/IFIP Workshop on Mobile Network Measurement (MNM 2017), Dublin, June 2017